

---

ComponentOne

# **DataSource for Entity Framework**

Copyright © 1987-2015 GrapeCity, Inc. All rights reserved.

**ComponentOne, a division of GrapeCity**

201 South Highland Avenue, Third Floor  
Pittsburgh, PA 15206 USA

**Website:** <http://www.componentone.com>  
**Sales:** [sales@componentone.com](mailto:sales@componentone.com)  
**Telephone:** 1.800.858.2739 or 1.412.681.4343 (Pittsburgh, PA USA Office)

**Trademarks**

The ComponentOne product name is a trademark and ComponentOne is a registered trademark of GrapeCity, Inc. All other trademarks used herein are the properties of their respective owners.

**Warranty**

ComponentOne warrants that the media on which the software is delivered is free from defects in material and workmanship, assuming normal use, for a period of 90 days from the date of purchase. If a defect occurs during this time, you may return the defective media to ComponentOne, along with a dated proof of purchase, and ComponentOne will replace it at no charge. After 90 days, you can obtain a replacement for the defective media by sending it and a check for \$25 (to cover postage and handling) to ComponentOne.

Except for the express warranty of the original media on which the software is delivered is set forth here, ComponentOne makes no other warranties, express or implied. Every attempt has been made to ensure that the information contained in this manual is correct as of the time it was written. ComponentOne is not responsible for any errors or omissions. ComponentOne's liability is limited to the amount you paid for the product. ComponentOne is not liable for any special, consequential, or other damages for any reason.

**Copying and Distribution**

While you are welcome to make backup copies of the software for your own use and protection, you are not permitted to make copies for the use of anyone else. We put a lot of time and effort into creating this product, and we appreciate your support in seeing that it is used by licensed users only.

# Table of Contents

DataSource for Entity Framework Overview .....	62
Help with WPF Edition .....	62
Key Features.....	62
C1DataSource Introduction .....	64
Unified Data Context .....	65
More Powerful Data Binding .....	65
Virtual Data Access .....	66
WPF Quick Start .....	66
Step 1 of 4: Adding a Data Source .....	66
Step 2 of 4: Connecting to the DataSource.....	69
Step 3 of 4: Adding a Grid .....	70
Step 4 of 4: Running the Project .....	70
DataSource for Entity Framework in WPF .....	71
Simple Binding .....	71
Server-Side Filtering.....	74
The Power of Client Data Cache .....	76
Master-Detail Binding .....	78
Large DataSets: Paging.....	79
Large Datasets: Virtual Mode .....	83
Automatic Lookup Columns in Grids .....	85
Customizing View.....	87
Working with Data Sources in Code .....	90
Live Views.....	97
Simplifying MVVM .....	97
Using Entity Framework DataSource in MVVM with other MVVM Frameworks.....	102
Samples.....	103
Getting Started Samples .....	104
How To Samples.....	104
Indexing Samples .....	104
Live View Samples.....	105
Programming Guide.....	105
Query Operators Supported in Live Views.....	106

Query Expressions Supported in Live Views .....	107
View Maintenance Mode.....	107
Updatable Views .....	108
Live View Performance .....	110
LiveLinq Query Performance: Logical Optimization.....	110
LiveLinq Query Performance: Tuning Indexing Performance .....	111
Live Views How To: Create Views Based on Other Views and Create Indexes on Views ..	112
Live Views How To: Use Live Views to Create Non-GUI Applications as a Set of Declarative Rules Instead of Prodedural Code .....	114
LiveLinq .....	116
LiveLinq Overview .....	116
LiveLinq in Silverlight .....	117
How to use LiveLinq .....	117
How to Query Collections with LiveLinq .....	117
Using the Built-in Collection Class IndexedCollection<T>(LiveLinq to Objects) .....	118
Using ADO.NET Data Collections (LiveLinq to DataSet) .....	120
Using XML Data (LiveLinq to XML) .....	121
Using Bindable Collection Classes (LiveLinq to Objects).....	122
LiveLinq to Objects: IndexedCollection (T) and other Collection Classes .....	123
How to Create Indexes.....	123
How to use Indexes Programmatically .....	125
How to Create a Live View .....	126
How to Bind GUI Controls to a Live View.....	127
How to Use Live Views in Non-GUI Code.....	127
Programming Guide.....	128
Working with Entities in Code .....	128
Using ClientView in Code .....	131
Client-Side Transactions.....	131
Virtual Mode .....	134
Unmanaged Virtual Mode Limitations.....	135
Other Virtual Mode Limitations .....	136
API Reference.....	137
C1.Data.Entity.4 Assembly .....	137
Namespaces .....	137



C1.Data Namespace .....	137
Overview .....	137
Classes .....	138
ClientCacheBase .....	138
Overview .....	139
Members .....	140
Methods .....	141
BulkChanges Method .....	142
CleanupCache Method .....	144
Clear Method .....	144
CreateScope Method .....	145
CreateTransaction Method .....	146
Refresh Method .....	147
RejectChanges Method .....	147
SaveChanges Method .....	148
ClientScope .....	149
Overview .....	150
Members .....	151
ClientScope Constructor .....	152
Methods .....	152
AddRef Method .....	153
AddRef(Object) Method .....	154
AddRef(Type) Method .....	155
Dispose Method .....	156
Release Method .....	156
Release(Object) Method .....	157
Release(Type) Method .....	158
Properties .....	159
ClientCache Property .....	159
ClientView<T> .....	160
Overview .....	161
Members .....	162
Methods .....	167
AsFiltered Method .....	171

AsFilteredBound<TKey> Method .....	172
CancelLoad Method .....	173
Include Method.....	173
Load Method.....	174
Paging Method.....	175
Paging<TKey>(Expression<Func<T,TKey>>,Int32) Method .....	175
Paging<TKey>(Expression<Func<T,TKey>>,Boolean,Int32) Method .....	177
ProgressiveLoading Method .....	178
ProgressiveLoading<TKey>(Expression<Func<T,TKey>>,Int32) Method ..	179
ProgressiveLoading<TKey>(Expression<Func<T,TKey>>,Boolean,Int32) Method .....	181
Refresh Method .....	183
Properties.....	183
AutoLoad Property.....	185
IsLoaded Property .....	186
IsLoading Property .....	186
Scope Property.....	187
Events.....	187
Loaded Event.....	188
ClientViewLoadedEventArgs .....	189
Overview .....	190
Members.....	191
Methods.....	192
MarkErrorAsHandled Method .....	192
Properties.....	193
Error Property .....	193
HasError Property .....	194
IsErrorHandled Property .....	195
Items Property .....	195
TotalItemCount Property .....	196
DataExtensions.....	196
Overview .....	197
Members.....	198
Methods.....	199

ExecuteIn<T> Method.....	199
FilteredView<T> .....	200
Overview .....	202
Members.....	203
Methods.....	209
BindFilterKey Method .....	212
BindFilterKey(Binding) Method .....	213
BindFilterKey(Object,String) Method.....	214
Properties.....	215
FilterKey Property .....	217
FilterKeyType Property .....	218
Operator Property.....	218
Fields .....	219
Unfiltered Field .....	219
PageChangingEventArgs .....	220
Overview .....	221
Members.....	221
PageChangingEventArgs Constructor .....	222
Properties.....	223
NewPageIndex Property .....	223
PagingView<T>.....	224
Overview .....	225
Members.....	227
Properties.....	232
LoadSize Property .....	234
PageCount Property.....	235
PageIndex Property.....	236
PageSize Property .....	236
TotalItemCount Property .....	237
ProgressiveView<T> .....	238
Overview .....	239
Members.....	240
Properties.....	245
LoadSize Property .....	247

SavedChangesEventArgs .....	248
Overview .....	249
Members .....	250
Methods .....	250
MarkErrorAsHandled Method .....	251
Properties .....	252
Error Property .....	252
HasError Property .....	253
IsErrorHandled Property .....	253
C1.Data.DataSource Namespace .....	254
Overview .....	254
Classes .....	256
BaseControlHandler .....	256
Overview .....	257
Members .....	258
Methods .....	260
Apply Method .....	261
Properties .....	262
AutoLookup Property .....	263
SupportsVirtualMode Property .....	263
VirtualMode Property .....	264
Fields .....	265
AutoLookupProperty Field .....	266
VirtualModeProperty Field .....	266
ClientCollectionView .....	267
Overview .....	268
Members .....	269
Methods .....	272
Add Method .....	273
AsLive<T> Method .....	274
Contains Method .....	275
IndexOf Method .....	276
MoveCurrentTo Method .....	277
MoveCurrentToFirst Method .....	278

MoveCurrentToLast Method .....	278
MoveCurrentToNext Method .....	279
MoveCurrentToPosition Method .....	280
MoveCurrentToPrevious Method .....	281
MoveToFirstPage Method .....	281
MoveToLastPage Method .....	282
MoveToNextPage Method .....	283
MoveToPage Method .....	283
MoveToPreviousPage Method .....	284
Remove Method .....	285
RemoveAt Method .....	286
Properties .....	287
CanAdd Property .....	288
CanChangePage Property .....	289
CanRemove Property .....	289
CollectionViewFactory Property .....	290
Count Property .....	291
CurrentItem Property .....	291
CurrentPosition Property .....	292
IsEmpty Property .....	293
IsPageChanging Property .....	293
Item Property .....	294
ItemProperties Property .....	295
PageCount Property .....	296
PageIndex Property .....	296
PageSize Property .....	297
TotalItemCount Property .....	297
Events .....	298
CurrentChanged Event .....	299
CurrentChanging Event .....	299
PageChanged Event .....	300
PageChanging Event .....	301
PropertyChanged Event .....	302
ClientViewSource .....	302

Overview .....	303
Members.....	304
ClientViewSource Constructor .....	309
Methods.....	309
DeferLoad Method.....	311
Load Method.....	311
LoadRange Method.....	312
Refresh Method .....	313
Properties.....	314
AutoLoad Property.....	316
BaseView Property.....	317
CacheTimeout Property .....	318
CurrentClientView Property.....	319
DataView Property.....	320
FilterDescriptors Property.....	321
FilterOperator Property .....	321
GroupDescriptors Property.....	322
Include Property.....	323
IsLoadingData Property.....	323
LoadCommand Property.....	324
LoadDelay Property.....	325
LoadSize Property .....	325
MoveToFirstOnLoad Property.....	326
Name Property.....	327
NameOverride Property.....	327
PageSize Property .....	328
SortDescriptors Property .....	329
VirtualMode Property .....	330
Events.....	331
LoadedData Event.....	331
PropertyChanged Event .....	332
ClientViewSourceException .....	333
Overview .....	334
Members.....	335

ClientViewSourceException Constructor .....	336
ClientViewSourceException Constructor().....	338
ClientViewSourceException Constructor(String) .....	338
ClientViewSourceException Constructor(String,Exception) .....	339
ClientViewSourceException Constructor(SerializationInfo,StreamingContext) .....	340
DependencyObjectCollection<T> .....	341
Overview .....	342
Members .....	344
DependencyObjectCollection<T> Constructor .....	346
Methods.....	347
InsertItem Method.....	349
SetItem Method .....	350
FilterDescriptor .....	351
Overview .....	352
Members.....	352
FilterDescriptor Constructor .....	355
FilterDescriptor Constructor(String,FilterOperator,Object) .....	356
FilterDescriptor Constructor().....	357
Properties.....	357
IgnoredValue Property.....	358
IsCaseSensitive Property.....	359
Operator Property.....	360
PropertyPath Property .....	360
Value Property .....	361
Fields .....	362
DefaultIgnoredValue Field .....	362
IgnoredValueProperty Field .....	363
IsCaseSensitiveProperty Field .....	364
OperatorProperty Field .....	364
PropertyPathProperty Field .....	365
ValueProperty Field .....	365
FilterDescriptorCollection .....	366
Overview .....	367

Members .....	368
FilterDescriptorCollection Constructor .....	371
GroupDescriptor .....	372
Overview .....	373
Members .....	374
GroupDescriptor Constructor .....	376
GroupDescriptor Constructor() .....	376
GroupDescriptor Constructor(String) .....	377
Properties .....	378
PropertyPath Property .....	378
Fields .....	379
PropertyPathProperty Field .....	379
GroupDescriptorCollection .....	380
Overview .....	381
Members .....	382
GroupDescriptorCollection Constructor .....	385
SortDescriptor .....	386
Overview .....	387
Members .....	387
SortDescriptor Constructor .....	389
SortDescriptor Constructor() .....	390
SortDescriptor Constructor(String,ListSortDirection) .....	391
Properties .....	392
Direction Property .....	392
PropertyPath Property .....	393
Fields .....	394
DirectionProperty Field .....	394
PropertyPathProperty Field .....	395
SortDescriptorCollection .....	395
Overview .....	396
Members .....	397
SortDescriptorCollection Constructor .....	401
Enumerations .....	401
FilterDescriptorLogicalOperator .....	401



FilterOperator .....	402
VirtualModeKind .....	404
C1.Data.Entities Namespace .....	405
Overview .....	405
Classes .....	405
EntityClientCache .....	405
Overview .....	406
Members .....	407
EntityClientCache Constructor .....	409
EntityClientCache Constructor(DbContext) .....	410
EntityClientCache Constructor(ObjectContext) .....	411
Methods .....	412
AcceptChanges Method .....	413
CreateScope Method .....	414
GetDefault Method .....	414
RegisterContext Method .....	415
RegisterContext(ObjectContext,Type) Method .....	416
RegisterContext(ObjectContext) Method .....	418
RegisterContext(DbContext,Type) Method .....	419
RegisterContext(DbContext) Method .....	420
Properties .....	422
DbContext Property .....	422
ObjectContext Property .....	423
EntityClientScope .....	423
Overview .....	424
Members .....	425
EntityClientScope Constructor .....	426
Methods .....	427
GetItems Method .....	428
GetItems<T>() Method .....	429
GetItems<T>(String) Method .....	429
Properties .....	431
ClientCache Property .....	431
EntityFrameworkExtensions .....	432

Overview .....	432
Members .....	433
Methods .....	434
AsCollectionView<T> Method .....	434
AsLive Method .....	436
AsLive<T>(EntityCollection<T>) Method .....	436
AsLive<T>(ICollection<T>,EntityClientScope) Method .....	438
EntityViewSource .....	439
Overview .....	440
Members .....	441
EntityViewSource Constructor .....	446
Properties .....	447
EntitySetName Property .....	450
Name Property .....	450
C1.Data.Transactions Namespace .....	451
Overview .....	451
Classes .....	452
ClientTransaction .....	452
Overview .....	453
Members .....	453
ClientTransaction Constructor .....	455
Methods .....	456
Commit Method .....	457
Dispose Method .....	457
Rollback Method .....	458
Scope Method .....	459
ScopeDataContext Method .....	460
Properties .....	461
HasChanges Property .....	461
State Property .....	462
Events .....	463
PropertyChanged Event .....	463
C1.Data.Util Namespace .....	464
Overview .....	464

Classes.....	464
DesignTime.....	464
Overview .....	465
Members.....	466
DesignTime Constructor .....	467
Methods.....	467
AboutBox Method.....	467
C1.LiveLinq.4 Assembly.....	468
Namespaces .....	468
C1.LiveLinq Namespace .....	468
Overview .....	468
Classes.....	470
CompiledQuery .....	470
Overview .....	471
Members.....	472
Methods.....	472
Compile Method .....	473
Compile<T1,T2,T3,T4,TResult>(Expression<Func<T1,T2,T3,T4,IIndexedSource<TResult>>>) Method.....	474
Compile<T1,T2,T3,TResult>(Expression<Func<T1,T2,T3,IIndexedSource<TResult>>>) Method.....	476
Compile<T1,T2,TResult>(Expression<Func<T1,T2,IIndexedSource<TResult>>>) Method .....	477
Compile<T,TResult>(Expression<Func<T,IIndexedSource<TResult>>>) Method .....	479
DeletedStateIsAvailableAttribute .....	480
Overview .....	482
Members.....	483
DeletedStateIsAvailableAttribute Constructor .....	484
Properties.....	485
IsAvailable Property .....	485
Hints .....	486
Overview .....	487
Members.....	488
Methods.....	489

Indexed Method.....	489
Indexed<T>(T,IndexingHintAction) Method .....	490
Indexed<T>(T) Method .....	491
Properties.....	493
DefaultAction Property .....	493
IndexedQueryExtensions .....	494
Overview .....	495
Members.....	496
Methods.....	498
AsIndexed<T> Method.....	499
GroupBy Method .....	501
GroupBy<TSource,TKey,TElement,TResult>(IIndexedSource<TSource>,Expr ession<Func<TSource,TKey>>,Expression<Func<TSource,TElement>>,Expre ssion<Func<TKey,IEnumerable<TElement>,TResult>>) Method .....	506
GroupBy<TSource,TKey,TElement>(IIndexedSource<TSource>,Expression< Func<TSource,TKey>>,Expression<Func<TSource,TElement>>) Method	508
GroupBy<TSource,TKey,TResult>(IIndexedSource<TSource>,Expression<Fu nc<TSource,TKey>>,Expression<Func<TKey,IEnumerable<TSource>,TResult >>) Method .....	510
GroupBy<TSource,TKey>(IIndexedSource<TSource>,Expression<Func<TSou rce,TKey>>) Method.....	512
GroupJoin<TOuter,TInner,TKey,TResult> Method .....	513
Join Method .....	516
Join<TOuter,TInner,TKey,TResult>(IIndexedSource<TOuter>,IEnumerable< TInner>,Expression<Func<TOuter,TKey>>,Expression<Func<TInner,TKey>>, Expression<Func<TOuter,TInner,TResult>>) Method .....	517
Join<TOuter,TInner,TKey,TResult>(IEnumerable<TOuter>,IIndexedSource< TInner>,Expression<Func<TOuter,TKey>>,Expression<Func<TInner,TKey>>, Expression<Func<TOuter,TInner,TResult>>) Method .....	520
Join<TOuter,TInner,TKey,TResult>(IIndexedSource<TOuter>,IIndexedSourc e<TInner>,Expression<Func<TOuter,TKey>>,Expression<Func<TInner,TKey >>,Expression<Func<TOuter,TInner,TResult>>) Method.....	522
OrderBy<T,TKey> Method .....	525
OrderByDescending<T,TKey> Method.....	526
Select<TSource,TResult> Method.....	527
SelectMany Method.....	529

SelectMany<TSource,TCollection,TResult>(IIndexedSource<TSource>,Expression<Func<TSource,IEnumerable<TCollection>>>,Expression<Func<TSource,TCollection,TResult>>) Method.....	531
SelectMany<TSource,TResult>(IIndexedSource<TSource>,Expression<Func<TSource,IEnumerable<TResult>>>) Method.....	533
ToIndexed Method .....	535
ToIndexed<T>(IObservableSource<T>) Method .....	536
ToIndexed<T>(IBindingList) Method.....	537
ToIndexed<T>(BindingList<T>) Method.....	539
Where Method.....	540
Where<T>(IIndexedSource<T>,Expression<Func<T,Boolean>>) Method	541
Where<T>(IIndexedSource<T>,Expression<Func<T,Boolean>>,Boolean) Method .....	542
LiveViewExtensions.....	544
Overview .....	545
Members.....	545
Methods.....	546
AsLive Method .....	547
AsLive<T>(IObservableSource<T>) Method.....	548
AsLive<T>(IObservableSource<T>,ViewOrder) Method .....	550
AsLive<T>(IBindingList) Method .....	551
AsLive<T>(IBindingList,ViewOrder) Method.....	553
AsLive<T>(BindingList<T>) Method .....	554
AsLive<T>(BindingList<T>,ViewOrder) Method .....	555
AsNonUpdatable<T> Method .....	557
AsUpdatable<T> Method .....	558
LiveAggregate Method.....	559
LiveAggregate<TSource>(View<TSource>,Expression<Func<TSource,TSource,TSource>>,Expression<Func<TSource,TSource,TSource>>,Expression<Func<TSource,TSource,Boolean>>) Method .....	563
LiveAggregate<TSource,TAccumulate>(View<TSource>,TAccumulate,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,Boolean>>) Method .....	565
LiveAggregate<TSource,TAccumulate,TResult>(View<TSource>,TAccumulate,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func	

<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,Boolean>>,Expression<Func<TAccumulate,TResult>>) Method .....	568
LiveAverage Method .....	571
LiveAverage(View<Int32>) Method .....	577
LiveAverage(View<Nullable<Int32>>) Method.....	578
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method .....	579
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method .....	581
LiveAverage(View<Int64>) Method .....	582
LiveAverage(View<Nullable<Int64>>) Method.....	583
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method .....	585
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>) Method .....	586
LiveAverage(View<Decimal>) Method .....	588
LiveAverage(View<Nullable<Decimal>>) Method .....	589
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>>) Method .....	590
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method .....	592
LiveAverage(View<Double>) Method .....	594
LiveAverage(View<Nullable<Double>>) Method.....	595
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method .....	596
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>) Method.....	598
LiveAverage(View<Single>) Method .....	599
LiveAverage(View<Nullable<Single>>) Method.....	601
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method .....	602
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method.....	603
LiveCount Method .....	605
LiveCount<T>(View<T>) Method .....	606
LiveCount<T>(View<T>,Expression<Func<T,Boolean>>) Method .....	607
LiveMax Method .....	608

LiveMax(View<Double>) Method .....	615
LiveMax(View<Nullable<Double>>) Method .....	616
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method .....	617
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Do uble>>>) Method .....	619
LiveMax(View<Single>) Method .....	621
LiveMax(View<Nullable<Single>>) Method .....	622
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method .....	623
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Sin gle>>>) Method .....	625
LiveMax<TSource,TResult>(View<TSource>,Expression<Func<TSource,TRes ult>>) Method .....	626
LiveMax<TSource>(View<TSource>) Method .....	628
LiveMax(View<Int32>) Method .....	629
LiveMax(View<Nullable<Int32>>) Method .....	631
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method .....	632
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int 32>>>) Method .....	633
LiveMax(View<Decimal>) Method .....	635
LiveMax(View<Nullable<Decimal>>) Method .....	636
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method .....	638
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<De cimal>>>) Method .....	639
LiveMax(View<Int64>) Method .....	641
LiveMax(View<Nullable<Int64>>) Method .....	642
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method .....	643
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int 64>>>) Method .....	645
LiveMin Method .....	646
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Do uble>>>) Method .....	653
LiveMin(View<Single>) Method .....	654

LiveMin(View<Nullable<Single>>) Method .....	655
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method .....	657
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Sin gle>>>) Method.....	658
LiveMin<TSource,TResult>(View<TSource>,Expression<Func<TSource,TRes ult>>) Method .....	660
LiveMin<TSource>(View<TSource>) Method .....	662
LiveMin(View<Int32>) Method .....	663
LiveMin(View<Nullable<Int32>>) Method.....	664
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method .....	665
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int 32>>>) Method .....	667
LiveMin(View<Decimal>) Method .....	669
LiveMin(View<Nullable<Decimal>>) Method .....	670
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method .....	671
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<De cimal>>>) Method.....	673
LiveMin(View<Int64>) Method .....	674
LiveMin(View<Nullable<Int64>>) Method.....	675
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method .....	677
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int 64>>>) Method .....	678
LiveMin(View<Double>) Method.....	680
LiveMin(View<Nullable<Double>>) Method .....	681
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method .....	682
LiveSum Method .....	684
LiveSum(View<Int32>) Method .....	689
LiveSum(View<Nullable<Int32>>) Method.....	691
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method .....	692
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int 32>>>) Method .....	693



LiveSum(View<Decimal>) Method.....	695
LiveSum(View<Nullable<Decimal>>) Method .....	696
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method .....	697
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<De cimal>>>) Method.....	699
LiveSum(View<Int64>) Method .....	701
LiveSum(View<Nullable<Int64>>) Method.....	702
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method .....	703
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int 64>>>) Method .....	705
LiveSum(View<Double>) Method .....	706
LiveSum(View<Nullable<Double>>) Method.....	707
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method .....	709
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Do uble>>>) Method .....	710
LiveSum(View<Single>) Method .....	712
LiveSum(View<Nullable<Single>>) Method.....	713
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method .....	714
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Sin gle>>>) Method.....	716
Ordering<T> .....	717
Overview .....	718
Members.....	720
Methods.....	720
GetEnumerator Method .....	721
ThenBy<TKey> Method.....	721
ThenByDescending<TKey> Method .....	723
QueryOptimizationException.....	724
Overview .....	725
Members.....	725
QueryOptimizationException Constructor .....	727
QueryOptimizationException Constructor() .....	728

QueryOptimizationException Constructor(String).....	729
QueryOptimizationException Constructor(String,Exception).....	730
QueryOptimizationException Constructor(SerializationInfo,StreamingContext) .....	731
SourceChangeEventArgs<T> .....	732
Overview .....	733
Members.....	734
SourceChangeEventArgs<T> Constructor .....	735
Methods.....	736
Equals Method .....	737
Equals(Object) Method .....	737
Equals(SourceChangeEventArgs<T>) Method .....	738
GetHashCode Method .....	739
ToString Method .....	740
Properties.....	740
ChangeType Property .....	741
Item Property.....	742
Ordinal Property .....	742
Enumerations.....	743
IndexingHintAction .....	743
Order .....	744
SourceChangeType .....	745
TransactionState .....	746
Interfaces .....	747
IObservableSource<T> .....	747
Overview .....	749
Members.....	750
Methods.....	751
EnableItemOrdinals Method.....	751
Properties.....	752
CreateNew Property .....	753
IsDeletedStateAvailable Property .....	753
SupportsItemOrdinals Property .....	754
Events.....	755

Changed Event .....	755
ITransaction .....	756
Overview .....	757
Members.....	757
Methods.....	758
Commit Method.....	759
Rollback Method .....	759
Scope Method.....	760
Properties.....	761
HasChanges Property.....	761
State Property .....	762
C1.LiveLinq.AdoNet Namespace .....	763
Overview .....	763
Classes.....	763
AdoNetExtensions.....	763
Overview .....	764
Members.....	765
Methods.....	766
AsIndexed Method.....	767
AsIndexed(DataTable) Method .....	768
AsIndexed<TRow>(DataTable) Method .....	769
AsIndexed<TRow>(TypedTableBase<TRow>) Method .....	770
AsLive Method .....	772
AsLive(DataTable) Method .....	772
AsLive<TRow>(DataTable) Method .....	773
AsLive<TRow>(TypedTableBase<TRow>) Method .....	775
BeginUpdate Method.....	776
EndUpdate Method .....	777
IndexedField Method.....	778
IndexedField<T>(DataRow,Int32) Method .....	779
IndexedField<T>(DataRow,DataColumn) Method.....	780
IndexedField<T>(DataRow,String) Method .....	782
IndexedField<T>(DataRow,Int32,IndexingHintAction) Method .....	784
IndexedField<T>(DataRow,DataColumn,IndexingHintAction) Method....	785

IndexedField<T>(DataRow,String,IndexingHintAction) Method .....	787
IndexedDataTable<TRow> .....	789
Overview .....	791
Members .....	792
Methods .....	793
BeginUpdate Method .....	794
EndUpdate Method .....	795
GetEnumerator Method .....	795
Properties .....	796
Count Property .....	797
Indexes Property .....	797
Item Property .....	798
Table Property .....	799
C1.LiveLinq.Collections Namespace .....	800
Overview .....	800
Classes .....	800
IndexableObject .....	800
Overview .....	801
Members .....	802
IndexableObject Constructor .....	803
Methods .....	804
OnPropertyChanged Method .....	804
Events .....	805
PropertyChanged Event .....	805
IndexedCollection<T> .....	806
Overview .....	808
Members .....	809
IndexedCollection<T> Constructor .....	812
IndexedCollection<T> Constructor() .....	813
IndexedCollection<T> Constructor(IList<T>) .....	813
IndexedCollection<T> Constructor(IList<T>,PropertyChangeListener<T>) ..	814
Methods .....	816
AddRange Method .....	817
BeginUpdate Method .....	818

ClearItems Method .....	819
EndUpdate Method .....	820
InsertItem Method.....	820
OnChanged Method.....	821
RemoveItem Method.....	822
SetItem Method .....	823
Properties.....	824
CreateNew Property .....	825
Indexes Property .....	825
Events.....	826
Changed Event .....	826
C1.LiveLinq.Indexing Namespace.....	827
Overview .....	827
Classes.....	828
Index<T>.....	828
Overview .....	830
Members.....	831
Methods.....	834
ContainsKey Method.....	835
Find Method.....	836
FindBetween Method .....	837
FindGreater Method .....	839
FindKeys Method .....	840
FindLess Method .....	841
FindStartingWith Method .....	843
GroupJoin Method .....	844
GroupJoin<T2,TResult>(IIndexScanner<T2>,Func<T,IEnumerable<T2>,TResult>) Method .....	846
GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<IEnumerable<T>,T2,TResult>) Method.....	847
GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<T,IEnumerable<T2>,TResult>) Method .....	849
Join Method .....	851
Join<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<T,T2,TResult>,JoinOperator) Method .....	852

Join<T2,TResult>(IIndexScanner<T2>,Func<T,T2,TResult>,JoinOperator) Method .....	854
Properties.....	856
ItemCount Property .....	857
KeyCount Property.....	858
Index<T,TKey>.....	859
Overview .....	860
Members.....	862
Methods.....	865
All Method .....	866
ContainsKey Method.....	867
ContainsKey(TKey) Method .....	868
Find Method.....	869
Find(TKey) Method .....	869
FindBetween Method .....	870
FindBetween(TKey,Boolean,TKey,Boolean,Func<TKey,Boolean>,Order) Method .....	871
FindGreater Method .....	873
FindGreater(TKey,Boolean,Func<TKey,Boolean>,Order) Method .....	874
FindKeys Method .....	875
FindKeys(IEnumerable<TKey>,Order) Method .....	876
FindLess Method.....	877
FindLess(TKey,Boolean,Func<TKey,Boolean>,Order) Method .....	878
FindSingle Method .....	879
GroupJoin Method.....	880
GroupJoin<T2,TResult>(IIndexScanner<T2,TKey>,Func<T,IEnumerable<T2>, TResult>) Method .....	883
GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<IEnumerable<T>, T2,TResult>) Method.....	885
GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<T,IEnumerab le<T2>,TResult>) Method.....	887
Join Method .....	888
Join<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<T,T2,TResult>,Joi nOperator) Method .....	890
Join<T2,TResult>(IIndexScanner<T2,TKey>,Func<T,T2,TResult>,JoinOperat or) Method .....	892

Keys Method .....	894
Properties.....	895
KeySelector Property .....	896
IndexCollection<T> .....	897
Overview .....	898
Members .....	899
IndexCollection<T> Constructor .....	901
Methods .....	902
Add Method .....	903
Add<TKey>(Expression<Func<T,TKey>>) Method .....	904
Add<TKey>(Expression<Func<T,TKey>>,Boolean) Method .....	905
Clear Method .....	907
Find Method.....	907
Find(LambdaExpression) Method .....	908
Find<TKey>(Expression<Func<T,TKey>>) Method.....	909
GetEnumerator Method .....	910
Remove Method .....	910
Remove(LambdaExpression) Method.....	911
Remove(Index<T>) Method .....	912
Properties.....	913
Count Property.....	914
Item Property .....	914
IndexDefinition<T>.....	915
Overview .....	917
Members.....	918
Properties.....	919
Algorithm Property .....	920
KeysIsUnique Property .....	920
KeySelector Property .....	921
KeyType Property.....	922
Locale Property .....	923
Root Property.....	923
Subindexes Property .....	924
IndexingAlgorithm.....	925

Overview .....	926
Members.....	927
Methods.....	927
CreateIndex<T,TKey> Method .....	928
Fields .....	930
RedBlackTree Field.....	931
IndexingException.....	931
Overview .....	932
Members.....	933
IndexingException Constructor.....	935
IndexingException Constructor() .....	936
IndexingException Constructor(String).....	936
IndexingException Constructor(String,Exception) .....	937
IndexingException Constructor(SerializationInfo,StreamingContext) .....	938
ScannerCollection<T> .....	939
Overview .....	940
Members.....	942
Methods.....	943
Add Method .....	943
Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean,IndexingAlgorith m,CultureInfo) Method .....	945
Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean,CultureInfo) Method .....	947
Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean) Method.....	949
Add<TKey>(Expression<Func<T,TKey>>,Boolean) Method .....	951
Add<TKey>(Expression<Func<T,TKey>>) Method .....	953
Add(LambdaExpression,Boolean,Boolean,IndexingAlgorithm,CultureInfo) Method .....	954
Clear Method .....	956
Contains Method .....	957
Contains(LambdaExpression) Method.....	957
Contains<TKey>(Expression<Func<T,TKey>>) Method .....	958
Find Method.....	959
Find(LambdaExpression) Method .....	960
Find<TKey>(Expression<Func<T,TKey>>) Method.....	961



GetEnumerator Method .....	962
Remove Method .....	963
Remove(LambdaExpression) Method .....	963
Remove<TKey>(Expression<Func<T,TKey>>) Method .....	964
Properties.....	965
Count Property.....	966
Subindex<T>.....	967
Overview .....	968
Members.....	969
Properties.....	970
Parent Property.....	971
Subindex<T,TKey> .....	972
Overview .....	973
Members.....	975
Properties.....	976
KeySelector Property .....	977
SubindexCollection<T>.....	978
Overview .....	979
Members.....	980
Methods.....	981
Add Method .....	982
Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean,IndexingAlgorith m,CultureInfo) Method .....	984
Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean,CultureInfo) Method .....	986
Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean) Method .....	988
Add<TKey>(Expression<Func<T,TKey>>,Boolean) Method .....	990
Add<TKey>(Expression<Func<T,TKey>>) Method .....	992
Add(LambdaExpression,Boolean,Boolean,IndexingAlgorithm,CultureInfo) Method .....	993
Clear Method .....	995
Contains Method .....	996
Contains(LambdaExpression) Method .....	996
Contains<TKey>(Expression<Func<T,TKey>>) Method .....	997
Find Method.....	999

Find<TKey>(Expression<Func<T,TKey>>) Method.....	999
Find(LambdaExpression) Method .....	1000
GetEnumerator Method .....	1001
Remove Method .....	1002
Remove(Subindex<T>) Method .....	1002
Remove(LambdaExpression) Method.....	1003
Properties.....	1004
Count Property.....	1005
Item Property .....	1006
Interfaces .....	1006
IIndexedSource<T> .....	1006
Overview .....	1008
Members.....	1009
Properties.....	1009
Indexes Property .....	1010
C1.LiveLinq.Indexing.Search Namespace.....	1010
Overview .....	1010
Classes.....	1012
GroupingQuery<T> .....	1012
Overview .....	1013
Members.....	1014
Methods.....	1015
GetEnumerator Method .....	1015
GroupingQuery<TKey,T>.....	1016
Overview .....	1017
Members.....	1019
Methods.....	1019
GetEnumerator Method .....	1020
IndexedGroup<T> .....	1020
Overview .....	1021
Members.....	1023
Properties.....	1024
Key Property.....	1024
KeyType Property.....	1025

IndexedGroup<TKey,T>.....	1026
Overview .....	1027
Members.....	1028
Properties.....	1029
Key Property.....	1030
KeyType Property.....	1030
IndexQuery<T>.....	1031
Overview .....	1032
Members.....	1033
Methods.....	1034
GetEnumerator Method .....	1035
GroupByUntypedKey Method.....	1035
Subindex Method.....	1036
Subindex<TKey>(Subindex<T,TKey>) Method .....	1036
Subindex(Subindex<T>) Method.....	1038
Properties.....	1039
Indexes Property .....	1039
IndexQuery<T,TKey> .....	1040
Overview .....	1041
Members.....	1042
Methods.....	1043
GroupByKey Method .....	1044
GroupByUntypedKey Method.....	1045
Enumerations.....	1046
JoinOperator .....	1046
Interfaces .....	1047
IIndexScanner<T>.....	1047
Overview .....	1048
Members.....	1049
Methods.....	1050
All Method .....	1051
ContainsKey Method.....	1052
Find Method.....	1053
FindBetween Method .....	1054

FindGreater Method .....	1056
FindKeys Method .....	1057
FindLess Method .....	1058
FindStartingWith Method .....	1060
GroupJoin Method .....	1061
GroupJoin<T2,TResult>(IIndexScanner<T2>,Func<T,IEnumerable<T2>,TResult>) Method .....	1063
GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<IEnumerable<T>,T2,TResult>) Method .....	1065
GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<T,IEnumerable<T2>,TResult>) Method .....	1066
Join Method .....	1068
Join<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<T,T2,TResult>,JoinOperator) Method .....	1070
Join<T2,TResult>(IIndexScanner<T2>,Func<T,T2,TResult>,JoinOperator) Method .....	1071
Properties.....	1073
Definition Property .....	1074
KeyCount Property.....	1074
ParentScanner Property.....	1075
IIndexScanner<T,TKey> .....	1076
Overview .....	1077
Members.....	1078
Methods.....	1079
All Method .....	1080
ContainsKey Method.....	1081
Find Method.....	1082
FindBetween Method .....	1083
FindGreater Method .....	1085
FindKeys Method .....	1086
FindLess Method .....	1088
GroupJoin Method .....	1089
GroupJoin<T2,TResult>(IIndexScanner<T2,TKey>,Func<T,IEnumerable<T2>,TResult>) Method .....	1091
GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<IEnumerable<T>,T2,TResult>) Method.....	1093

GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<T,IEnumer able<T2>,TResult>) Method.....	1094
Join Method .....	1096
Join<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<T,T2,TResult>,Joi nOperator) Method .....	1098
Join<T2,TResult>(IIndexScanner<T2,TKey>,Func<T,T2,TResult>,JoinOperat or) Method .....	1099
Keys Method .....	1101
C1.LiveLinq.Listeners Namespace .....	1102
Overview .....	1102
Classes.....	1102
PropertyChangeListener<T> .....	1102
Overview .....	1104
Members.....	1105
Methods.....	1106
Clear Method .....	1107
CreateDefault Method.....	1107
GetListeningProperties Method .....	1108
StartListening Method .....	1109
StopListening Method.....	1110
Events.....	1111
PropertyChanged Event .....	1111
C1.LiveLinq.LiveViews Namespace .....	1112
Overview .....	1112
Classes.....	1114
AggregationView<TSource,TResult>.....	1114
Overview .....	1115
Members.....	1116
Properties.....	1120
Value Property .....	1122
GroupView<TKey,TElement> .....	1122
Overview .....	1124
Members.....	1126
Methods.....	1130
Maintain Method .....	1132

PurgeEmptyGroups Method .....	1133
Rebuild Method .....	1133
ToString Method .....	1134
Properties.....	1134
DeferredMaintenance Property .....	1136
Key Property.....	1137
MaintenanceMode Property .....	1137
Parent Property.....	1138
OrderedView<T> .....	1138
Overview .....	1140
Members.....	1141
Methods.....	1145
ThenBy<TKey> Method.....	1148
ThenByDescending<TKey> Method .....	1149
PropertyIsNotVirtualException .....	1150
Overview .....	1151
Members.....	1152
PropertyIsNotVirtualException Constructor .....	1153
Properties.....	1155
Context Property.....	1155
Message Property .....	1156
Property Property .....	1157
ResultType Property.....	1157
View .....	1158
Overview .....	1159
Members.....	1160
Methods.....	1162
AllowInResult Method .....	1163
AsCollectionViewFactory Method .....	1163
AsDynamic Method.....	1164
DeferMaintenance Method .....	1165
Maintain Method .....	1166
PurgeEmptyGroups Method .....	1167
Rebuild Method .....	1167

SetTransaction Method .....	1168
Properties.....	1169
Count Property.....	1171
CurrentItem Property .....	1171
DataBindingMode Property .....	1172
DeferredMaintenance Property .....	1172
IsReadOnly Property .....	1173
MaintenanceMode Property .....	1174
MoveToFirstOnReset Property .....	1175
Order Property.....	1176
Rows Property.....	1176
Transaction Property .....	1177
View<T>.....	1178
Overview .....	1179
Members.....	1181
Methods.....	1185
AttachAggregationView Method .....	1187
AttachAggregationView<TResult>(Object,Func<View,AggregationView<T,TResult>>) Method .....	1187
AttachAggregationView<TResult>(Func<View,AggregationView<T,TResult>>) Method .....	1188
AttachView Method.....	1189
AttachView<TResult>(Func<View,View<TResult>>) Method.....	1190
AttachView<TResult>(Object,Func<View,View<TResult>>) Method .....	1191
Concat Method .....	1192
Contains Method .....	1192
GetEnumerator Method .....	1193
GroupBy Method .....	1194
GroupBy<TKey>(Expression<Func<T,TKey>>) Method .....	1197
GroupBy<TKey,TElement>(Expression<Func<T,TKey>>,Expression<Func<T,TElement>>) Method .....	1198
GroupBy<TKey,TElement,TResult>(Expression<Func<T,TKey>>,Expression<Func<T,TElement>>,Expression<Func<TKey,IEnumerable<TElement>,TResult>>) Method .....	1200
GroupJoin<TInner,TKey,TResult> Method.....	1201

IndexOf Method .....	1203
Join<TInner,TKey,TResult> Method .....	1204
OrderBy<TKey>(Expression<Func<T,TKey>>) Method .....	1206
OrderByDescending<TKey> Method.....	1207
Select<TResult> Method .....	1208
SelectMany Method.....	1209
SelectMany<TCollection,TResult>(Expression<Func<T,IObservableSource<TCollection>>>,Expression<Func<T,TCollection,TResult>>) Method .....	1212
SelectMany<TResult>(Expression<Func<T,IObservableSource<TResult>>>) Method .....	1213
ToString Method .....	1214
Union Method.....	1215
Where Method.....	1216
Properties.....	1217
Indexes Property .....	1218
Item Property .....	1219
Events.....	1220
Changed Event .....	1220
ViewRow .....	1221
Overview .....	1222
Members.....	1223
Methods.....	1225
BeginEdit Method .....	1225
CancelEdit Method.....	1227
Delete Method .....	1227
EndEdit Method .....	1228
OnPropertyChanged Method .....	1229
ToString Method .....	1230
Properties.....	1230
Item Property .....	1231
Item(Int32) Property .....	1232
Item(String) Property .....	1233
RowState Property .....	1234
Tag Property.....	1235
Value Property .....	1235



View Property .....	1236
Events.....	1237
PropertyChanged Event .....	1237
ViewRowAddingEventArgs.....	1238
Overview .....	1239
Members .....	1240
Properties.....	1240
Row Property .....	1241
ViewRowCollection .....	1241
Overview .....	1243
Members .....	1244
Methods.....	1247
Clear Method .....	1248
Contains Method .....	1248
CreateRow Method.....	1249
DeferRefresh Method .....	1250
GetEnumerator Method .....	1251
GetItemProperties Method .....	1251
IndexOf Method.....	1253
Remove Method .....	1253
RemoveAt Method.....	1254
Properties.....	1255
AllowClear Property .....	1257
AllowEdit Property .....	1257
AllowNew Property .....	1258
AllowRemove Property .....	1259
Count Property.....	1260
GroupDescriptions Property .....	1260
Groups Property.....	1261
Indexes Property .....	1262
Item Property .....	1262
Properties Property.....	1263
SortDescriptions Property.....	1264
Events.....	1265

Changed Event .....	1265
ViewRowAdding Event .....	1266
ViewRowPropertyInfo .....	1267
Overview .....	1268
Members .....	1269
Properties .....	1269
ImmediateUpdate Property .....	1270
PropertyName Property .....	1270
Enumerations .....	1271
DataBindingMode .....	1271
ViewMaintenanceMode .....	1272
ViewOrder .....	1274
ViewRowState .....	1275
C1.LiveLinq.LiveViews.Xml Namespace .....	1277
Overview .....	1277
Classes .....	1277
XHint .....	1277
Overview .....	1278
Members .....	1279
Operators .....	1280
Explicit Type Conversion Operator .....	1280
XmlExtensions .....	1281
Overview .....	1282
Members .....	1283
Methods .....	1285
AsLive Method .....	1286
AsLive(XDocument) Method .....	1286
AsLive(XElement) Method .....	1288
Attributes Method .....	1289
Attributes(View<XElement>) Method .....	1290
Attributes(View<XElement>,XName) Method .....	1291
BeginUpdate Method .....	1292
DescendantNodes<T> Method .....	1293
DescendantNodesAndSelf Method .....	1294

Descendants Method.....	1295
Descendants<T>(View<T>) Method.....	1296
Descendants<T>(View<T>,XName) Method .....	1297
DescendantsAndSelf Method .....	1299
DescendantsAndSelf(View<XElement>) Method .....	1300
DescendantsAndSelf(View<XElement>,XName) Method .....	1301
Elements Method.....	1302
Elements<T>(View<T>) Method .....	1303
Elements<T>(View<T>,XName) Method.....	1304
EndUpdate Method .....	1305
IndexedAttribute Method.....	1306
IndexedAttribute(XElement,XName) Method .....	1307
IndexedAttribute(XAttribute) Method .....	1309
IndexedAttribute(XAttribute,IndexingHintAction) Method .....	1310
IndexedAttribute(XElement,XName,IndexingHintAction) Method .....	1312
IndexedElement Method .....	1313
IndexedElement(XContainer,XName) Method .....	1314
IndexedElement(XElement) Method .....	1316
IndexedElement(XElement,IndexingHintAction) Method .....	1317
IndexedElement(XContainer,XName,IndexingHintAction) Method .....	1319
Nodes<T> Method .....	1320
Root Method.....	1322
C1.LiveLinq.Metadata Namespace .....	1323
Overview .....	1323
C1.WPF.LiveLinq Namespace.....	1323
Overview .....	1323
Classes.....	1323
WpfExtensions .....	1323
Overview .....	1324
Members.....	1325
Methods.....	1326
AsLive Method .....	1326
AsLive<T>(INotifyCollectionChanged) Method.....	1327
AsLive<T>(INotifyCollectionChanged,ViewOrder) Method .....	1329

AsLive<T>(ObservableCollection<T>) Method.....	1331
AsLive<T>(ObservableCollection<T>,ViewOrder) Method .....	1332
AsLive<T>(ReadOnlyObservableCollection<T>) Method .....	1333
AsLive<T>(ReadOnlyObservableCollection<T>,ViewOrder) Method .....	1335
ToIndexed Method .....	1336
ToIndexed<T>(INotifyCollectionChanged) Method .....	1337
ToIndexed<T>(ObservableCollection<T>) Method.....	1338
C1.Silverlight.Data.Entity.5 Assembly .....	1340
Namespaces .....	1340
C1.Data Namespace .....	1340
Overview .....	1340
Classes.....	1341
ClientCacheBase.....	1341
Overview .....	1342
Members.....	1343
Methods.....	1344
BulkChanges Method.....	1345
CleanupCache Method.....	1347
Clear Method .....	1348
CreateScope Method .....	1348
CreateTransaction Method.....	1349
Refresh Method .....	1350
RejectChanges Method.....	1351
SaveChanges Method .....	1351
ClientScope .....	1352
Overview .....	1353
Members.....	1353
ClientScope Constructor .....	1354
Methods.....	1355
AddRef Method.....	1356
AddRef(Object) Method.....	1356
AddRef(Type) Method .....	1357
Dispose Method.....	1358
Release Method .....	1358

Release(Object) Method .....	1359
Release(Type) Method .....	1360
Properties.....	1360
ClientCache Property .....	1361
ClientView<T> .....	1361
Overview .....	1362
Members.....	1364
Methods.....	1368
AsFiltered Method .....	1371
AsFilteredBound<TKey> Method .....	1372
CancelLoad Method .....	1373
Load Method.....	1374
Paging Method.....	1374
Paging<TKey>(Expression<Func<T,TKey>>,Int32) Method .....	1375
Paging<TKey>(Expression<Func<T,TKey>>,Boolean,Int32) Method .....	1376
ProgressiveLoading Method .....	1378
ProgressiveLoading<TKey>(Expression<Func<T,TKey>>,Int32) Method .....	1379
ProgressiveLoading<TKey>(Expression<Func<T,TKey>>,Boolean,Int32) Method .....	1380
Refresh Method .....	1382
Properties.....	1383
AutoLoad Property.....	1384
IsLoaded Property .....	1385
IsLoading Property .....	1385
Scope Property.....	1386
Events.....	1387
Loaded Event.....	1387
ClientViewLoadedEventArgs .....	1388
Overview .....	1389
Members.....	1390
Methods.....	1391
MarkErrorAsHandled Method .....	1391
Properties.....	1392
Error Property .....	1393

HasError Property .....	1393
IsErrorHandled Property .....	1394
Items Property .....	1395
TotalItemCount Property .....	1395
ValidationErrors Property .....	1396
DataExtensions.....	1396
Overview .....	1397
Members.....	1398
Methods.....	1399
ExecuteIn Method.....	1399
ExecuteIn<T>(IEnumerable<T>,ClientScope) Method.....	1400
ExecuteIn<T>(EntityQuery<T>,RiaClientScope) Method .....	1400
FilteredView<T> .....	1402
Overview .....	1403
Members.....	1404
Methods.....	1409
BindFilterKey Method .....	1412
BindFilterKey(Binding) Method .....	1413
BindFilterKey(Object,String) Method.....	1413
Properties.....	1414
FilterKey Property .....	1416
FilterKeyType Property .....	1416
Operator Property.....	1417
Fields .....	1418
Unfiltered Field .....	1418
PagingView<T>.....	1419
Overview .....	1420
Members.....	1421
Properties.....	1426
LoadSize Property .....	1428
PageCount Property.....	1429
PageIndex Property.....	1429
PageSize Property .....	1430
TotalItemCount Property .....	1431

ProgressiveView<T> .....	1431
Overview .....	1432
Members .....	1433
Properties .....	1438
LoadSize Property .....	1440
SavedChangesEventArgs .....	1440
Overview .....	1441
Members .....	1442
Methods .....	1443
MarkErrorAsHandled Method .....	1444
Properties .....	1444
Error Property .....	1445
HasError Property .....	1445
IsErrorHandled Property .....	1446
C1.Data.DataSource Namespace .....	1447
Overview .....	1447
Classes .....	1448
BaseControlHandler .....	1448
Overview .....	1449
Members .....	1450
Methods .....	1451
Apply Method .....	1452
Properties .....	1453
AutoLookup Property .....	1454
SupportsVirtualMode Property .....	1454
VirtualMode Property .....	1455
Fields .....	1456
AutoLookupProperty Field .....	1456
VirtualModeProperty Field .....	1457
ClientCollectionView .....	1458
Overview .....	1459
Members .....	1459
Methods .....	1463
Add Method .....	1464

AsLive<T> Method .....	1465
Contains Method .....	1466
IndexOf Method.....	1467
MoveCurrentTo Method.....	1468
MoveCurrentToFirst Method.....	1469
MoveCurrentToLast Method .....	1470
MoveCurrentToNext Method .....	1470
MoveCurrentToPosition Method.....	1471
MoveCurrentToPrevious Method.....	1472
MoveToFirstPage Method .....	1472
MoveToLastPage Method.....	1473
MoveToNextPage Method.....	1474
MoveToPage Method .....	1474
MoveToPreviousPage Method .....	1475
Remove Method .....	1476
RemoveAt Method.....	1477
Properties.....	1478
CanAdd Property.....	1479
CanChangePage Property .....	1480
CanRemove Property .....	1480
CollectionViewFactory Property .....	1481
Count Property.....	1482
CurrentItem Property .....	1482
CurrentPosition Property.....	1483
IsEmpty Property .....	1484
IsPageChanging Property .....	1484
Item Property.....	1485
PageCount Property.....	1486
PageIndex Property.....	1487
PageSize Property .....	1487
TotalItemCount Property .....	1488
Events.....	1488
CurrentChanged Event.....	1489
CurrentChanging Event .....	1490



PageChanged Event .....	1490
PageChanging Event.....	1491
PropertyChanged Event .....	1492
ClientViewSource .....	1493
Overview .....	1494
Members .....	1495
ClientViewSource Constructor .....	1499
Methods .....	1499
DeferLoad Method.....	1500
Load Method.....	1501
LoadRange Method.....	1502
Refresh Method .....	1503
Properties.....	1503
AutoLoad Property.....	1506
BaseView Property.....	1507
CacheTimeout Property .....	1508
CurrentClientView Property.....	1509
DataView Property.....	1509
FilterDescriptors Property.....	1510
FilterOperator Property .....	1511
GroupDescriptors Property.....	1511
IsLoadingData Property.....	1512
LoadCommand Property .....	1513
LoadDelay Property.....	1514
LoadSize Property .....	1514
MoveToFirstOnLoad Property.....	1515
Name Property.....	1516
NameOverride Property.....	1517
PageSize Property .....	1517
SortDescriptors Property .....	1518
VirtualMode Property .....	1519
Events.....	1520
LoadedData Event .....	1520
PropertyChanged Event .....	1521

ClientViewSourceException .....	1522
Overview .....	1523
Members .....	1524
ClientViewSourceException Constructor .....	1526
ClientViewSourceException Constructor() .....	1526
ClientViewSourceException Constructor(String) .....	1527
ClientViewSourceException Constructor(String,Exception) .....	1528
Enumerations .....	1529
VirtualModeKind .....	1529
C1.Data.Transactions Namespace .....	1530
Overview .....	1530
Classes .....	1531
ClientTransaction .....	1531
Overview .....	1532
Members .....	1532
ClientTransaction Constructor .....	1534
Methods .....	1534
Commit Method .....	1535
Dispose Method .....	1536
Rollback Method .....	1536
Scope Method .....	1537
ScopeDataContext Method .....	1537
Properties .....	1538
HasChanges Property .....	1538
State Property .....	1539
Events .....	1540
PropertyChanged Event .....	1540
C1.Data.Util Namespace .....	1541
Overview .....	1541
Classes .....	1541
DesignTime .....	1541
Overview .....	1542
Members .....	1543
DesignTime Constructor .....	1543

C1.Silverlight.Data Namespace .....	1544
Overview .....	1544
Classes.....	1544
ControlHandler.....	1544
Overview .....	1545
Members.....	1547
ControlHandler Constructor .....	1548
Properties.....	1549
DataSource Property.....	1550
Fields .....	1551
DataSourceProperty Field .....	1551
C1.Silverlight.Data.RiaServices Namespace.....	1552
Overview .....	1552
Classes.....	1553
C1DataSource.....	1553
Overview .....	1554
Members.....	1555
C1DataSource Constructor .....	1564
Methods.....	1565
GetControlHandler Method.....	1568
Load Method.....	1569
Refresh Method .....	1570
RejectChanges Method.....	1571
SaveChanges Method .....	1571
SetControlHandler Method .....	1572
Properties.....	1573
ClientCache Property .....	1577
ClientScope Property .....	1578
DomainContext Property .....	1578
DomainContextTypeName Property.....	1579
Item Property.....	1580
Item(String) Property .....	1581
Item(Int32) Property .....	1581
RefreshInterval Property.....	1582

ViewSources Property .....	1583
Fields .....	1584
ControlHandlerProperty Field .....	1584
Events .....	1585
SavedChanges Event .....	1587
SavingChanges Event .....	1588
RiaClientCache .....	1589
Overview .....	1590
Members .....	1591
RiaClientCache Constructor .....	1593
Methods .....	1594
CreateScope Method .....	1595
GetDefault Method .....	1596
RegisterContext Method .....	1597
RegisterContext(DomainContext,Type) Method .....	1598
RegisterContext(DomainContext) Method .....	1599
Properties .....	1601
DomainContext Property .....	1601
IsLoading Property .....	1602
RiaClientScope .....	1603
Overview .....	1603
Members .....	1604
RiaClientScope Constructor .....	1605
Methods .....	1606
GetItems Method .....	1606
GetItems<T>(String,IDictionary<String,Object>) Method .....	1607
GetItems<T>(EntityQuery<T>) Method .....	1608
Properties .....	1609
ClientCache Property .....	1609
RiaServicesExtensions .....	1610
Overview .....	1611
Members .....	1611
Methods .....	1612
AsLive<T> Method .....	1612

RiaViewSource .....	1613
Overview .....	1614
Members .....	1615
RiaViewSource Constructor .....	1620
Properties.....	1620
Name Property.....	1623
Parameters Property.....	1624
QueryName Property.....	1625
RiaViewSourceCollection .....	1626
Overview .....	1627
Members .....	1627
RiaViewSourceCollection Constructor .....	1630
Properties.....	1631
Item Property .....	1632
C1.Util.Licensing Namespace .....	1633
Overview .....	1633
Classes.....	1633
LicenseMode .....	1633
Overview .....	1634
Members.....	1635
LicenseMode Constructor .....	1636
Methods.....	1636
GetEvaluation Method.....	1636
SetEvaluation Method .....	1637
Fields .....	1638
EvaluationProperty Field.....	1638
C1.Silverlight.LiveLinq.5 Assembly.....	1639
Namespaces .....	1639
(Global) Namespace.....	1639
Overview .....	1639
Classes.....	1640
C1LiveLinqInfo.....	1640
Overview .....	1640
Members.....	1641

Fields .....	1642
PublicKey_C1_Silverlight_LiveLinq Field .....	1642
PublicKey_System_Core Field .....	1643
C1.LiveLinq Namespace .....	1644
Overview .....	1644
Classes .....	1645
LiveViewExtensions .....	1645
Overview .....	1646
Members .....	1647
Methods .....	1648
AsLive Method .....	1649
AsLive<T>(IObservableSource<T>) Method .....	1650
AsLive<T>(IObservableSource<T>,ViewOrder) Method .....	1651
AsLive<T>(INotifyCollectionChanged) Method .....	1653
AsLive<T>(INotifyCollectionChanged,ViewOrder) Method .....	1654
AsLive<T>(ObservableCollection<T>) Method .....	1656
AsLive<T>(ObservableCollection<T>,ViewOrder) Method .....	1657
AsLive<T>(ReadOnlyObservableCollection<T>) Method .....	1659
AsLive<T>(ReadOnlyObservableCollection<T>,ViewOrder) Method .....	1660
AsNonUpdatable<T> Method .....	1661
AsUpdatable<T> Method .....	1662
LiveAggregate Method .....	1664
LiveAggregate<TSource>(View<TSource>,Expression<Func<TSource,TSource>,TSource>>,Expression<Func<TSource,TSource,TSource>>>,Expression<Func<TSource,TSource,Boolean>>) Method .....	1667
LiveAggregate<TSource,TAccumulate>(View<TSource>,TAccumulate,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,TAccumulate>>>,Expression<Func<TAccumulate,TSource,Boolean>>) Method .....	1670
LiveAggregate<TSource,TAccumulate,TResult>(View<TSource>,TAccumulate,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,TAccumulate>>>,Expression<Func<TAccumulate,TSource,Boolean>>,Expression<Func<TAccumulate,TResult>>) Method .....	1672
LiveAverage Method .....	1675
LiveAverage(View<Int32>) Method .....	1681
LiveAverage(View<Nullable<Int32>>) Method .....	1682

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Int32>>)	
Method .....	1684
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable	
<Int32>>>) Method .....	1685
LiveAverage(View<Int64>) Method .....	1687
LiveAverage(View<Nullable<Int64>>) Method.....	1688
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Int64>>)	
Method .....	1689
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable	
<Int64>>>) Method .....	1691
LiveAverage(View<Decimal>) Method .....	1692
LiveAverage(View<Nullable<Decimal>>) Method .....	1694
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Decimal	
>>) Method .....	1695
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable	
<Decimal>>>) Method .....	1697
LiveAverage(View<Double>) Method .....	1698
LiveAverage(View<Nullable<Double>>) Method.....	1699
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Double>	
>) Method .....	1701
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable	
<Double>>>) Method.....	1702
LiveAverage(View<Single>) Method .....	1704
LiveAverage(View<Nullable<Single>>) Method.....	1705
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Single>>	
) Method .....	1706
LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable	
<Single>>>) Method.....	1708
LiveCount Method .....	1710
LiveCount<T>(View<T>) Method .....	1710
LiveCount<T>(View<T>,Expression<Func<T,Boolean>>) Method .....	1711
LiveMax Method .....	1713
LiveMax(View<Double>) Method .....	1720
LiveMax(View<Nullable<Double>>) Method.....	1721
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Double>>)	
Method .....	1722

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>) Method .....	1724
LiveMax(View<Single>) Method .....	1725
LiveMax(View<Nullable<Single>>) Method .....	1726
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method .....	1728
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method .....	1729
LiveMax<TSource,TResult>(View<TSource>,Expression<Func<TSource,TResult>>) Method .....	1731
LiveMax<TSource>(View<TSource>) Method .....	1733
LiveMax(View<Int32>) Method .....	1734
LiveMax(View<Nullable<Int32>>) Method .....	1735
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method .....	1736
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method .....	1738
LiveMax(View<Decimal>) Method .....	1740
LiveMax(View<Nullable<Decimal>>) Method .....	1741
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method .....	1742
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method .....	1744
LiveMax(View<Int64>) Method .....	1745
LiveMax(View<Nullable<Int64>>) Method .....	1746
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method .....	1748
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>) Method .....	1749
LiveMin Method .....	1751
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>) Method .....	1757
LiveMin(View<Single>) Method .....	1759
LiveMin(View<Nullable<Single>>) Method .....	1760
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method .....	1761



LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method.....	1763
LiveMin<TSource,TResult>(View<TSource>,Expression<Func<TSource,TResult>>) Method .....	1764
LiveMin<TSource>(View<TSource>) Method .....	1766
LiveMin(View<Int32>) Method .....	1768
LiveMin(View<Nullable<Int32>>) Method.....	1769
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method .....	1770
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method .....	1772
LiveMin(View<Decimal>) Method .....	1773
LiveMin(View<Nullable<Decimal>>) Method .....	1774
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method .....	1776
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method.....	1777
LiveMin(View<Int64>) Method .....	1779
LiveMin(View<Nullable<Int64>>) Method.....	1780
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method .....	1781
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>) Method .....	1783
LiveMin(View<Double>) Method.....	1784
LiveMin(View<Nullable<Double>>) Method .....	1785
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method .....	1787
LiveSum Method .....	1788
LiveSum(View<Int32>) Method .....	1794
LiveSum(View<Nullable<Int32>>) Method.....	1795
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method .....	1796
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method .....	1798
LiveSum(View<Decimal>) Method.....	1799
LiveSum(View<Nullable<Decimal>>) Method .....	1801

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>)	
Method .....	1802
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<De	
cimal>>>) Method.....	1803
LiveSum(View<Int64>) Method .....	1805
LiveSum(View<Nullable<Int64>>) Method .....	1806
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Int64>>)	
Method .....	1807
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int	
64>>>) Method .....	1809
LiveSum(View<Double>) Method .....	1811
LiveSum(View<Nullable<Double>>) Method.....	1812
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Double>>)	
Method .....	1813
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Do	
uble>>>) Method .....	1815
LiveSum(View<Single>) Method .....	1816
LiveSum(View<Nullable<Single>>) Method.....	1817
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Single>>)	
Method .....	1819
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Sin	
gle>>>) Method.....	1820
SourceChangeEventArgs<T> .....	1822
Overview .....	1823
Members.....	1824
SourceChangeEventArgs<T> Constructor .....	1825
Methods.....	1826
Equals Method .....	1827
Equals(Object) Method .....	1827
Equals(SourceChangeEventArgs<T>) Method .....	1828
GetHashCode Method .....	1829
ToString Method .....	1830
Properties.....	1830
ChangeType Property .....	1831
Item Property .....	1832
Ordinal Property .....	1832

Enumerations.....	1833
Order .....	1833
SourceChangeType .....	1834
TransactionState .....	1835
Interfaces .....	1836
IObservableSource<T> .....	1836
Overview .....	1837
Members.....	1838
Methods.....	1839
EnableItemOrdinals Method.....	1839
Properties.....	1840
CreateNew Property .....	1841
IsDeletedStateAvailable Property .....	1841
SupportsItemOrdinals Property .....	1842
Events.....	1843
Changed Event .....	1843
ITransaction .....	1844
Overview .....	1845
Members.....	1845
Methods.....	1846
Commit Method.....	1847
Rollback Method .....	1847
Scope Method.....	1848
Properties.....	1849
HasChanges Property.....	1849
State Property .....	1850
C1.LiveLinq.LiveViews Namespace .....	1851
Overview .....	1851
Classes.....	1852
AggregationView<TSource,TResult> .....	1852
Overview .....	1853
Members.....	1854
Properties.....	1858
Value Property .....	1860

GroupView<TKey,TElement> .....	1860
Overview .....	1862
Members .....	1864
Methods .....	1868
Maintain Method .....	1869
PurgeEmptyGroups Method .....	1870
Rebuild Method .....	1871
ToString Method .....	1871
Properties .....	1872
DeferredMaintenance Property .....	1873
Key Property .....	1874
MaintenanceMode Property .....	1874
Parent Property .....	1875
OrderedView<T> .....	1876
Overview .....	1877
Members .....	1878
Methods .....	1882
ThenBy<TKey> Method .....	1885
ThenByDescending<TKey> Method .....	1886
PropertyIsNotVirtualException .....	1887
Overview .....	1888
Members .....	1888
Properties .....	1890
Context Property .....	1891
Message Property .....	1891
Property Property .....	1892
ResultType Property .....	1892
View .....	1893
Overview .....	1894
Members .....	1895
Methods .....	1897
AllowInResult Method .....	1898
AsCollectionViewFactory Method .....	1899
AsDynamic Method .....	1899

DeferMaintenance Method .....	1900
Maintain Method .....	1901
PurgeEmptyGroups Method .....	1902
Rebuild Method .....	1903
SetTransaction Method .....	1903
Properties.....	1905
Count Property.....	1906
CurrentItem Property .....	1906
DeferredMaintenance Property .....	1907
IsReadOnly Property .....	1908
MaintenanceMode Property .....	1909
MoveToFirstOnReset Property .....	1910
Order Property .....	1910
Rows Property.....	1911
Transaction Property .....	1912
View<T>.....	1912
Overview .....	1914
Members.....	1916
Methods.....	1919
AttachAggregationView Method .....	1921
AttachAggregationView<TResult>(Object,Func<View,AggregationView<T,TResult>>) Method .....	1922
AttachAggregationView<TResult>(Func<View,AggregationView<T,TResult>>) Method .....	1923
AttachView Method.....	1924
AttachView<TResult>(Func<View,View<TResult>>) Method.....	1924
AttachView<TResult>(Object,Func<View,View<TResult>>) Method .....	1925
Concat Method .....	1926
Concat(IObservableSource<T>) Method.....	1926
Concat(ObservableCollection<T>) Method .....	1927
Contains Method .....	1928
GetEnumerator Method .....	1929
GroupBy Method .....	1930
GroupBy<TKey>(Expression<Func<T,TKey>>) Method .....	1933

GroupBy<TKey,TElement>(Expression<Func<T,TKey>>,Expression<Func<T,TElement>>) Method .....	1934
GroupBy<TKey,TElement,TResult>(Expression<Func<T,TKey>>,Expression<Func<T,TElement>>,Expression<Func<TKey,IEnumerable<TElement>,TResult>>) Method .....	1935
GroupJoin Method .....	1937
GroupJoin<TInner,TKey,TResult>(IObservableSource<TInner>,Expression<Func<T,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<T,GroupView<TKey,TInner>,TResult>>) Method .....	1938
GroupJoin<TInner,TKey,TResult>(ObservableCollection<TInner>,Expression<Func<T,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<T,GroupView<TKey,TInner>,TResult>>) Method .....	1940
IndexOf Method .....	1942
Join Method .....	1943
Join<TInner,TKey,TResult>(IObservableSource<TInner>,Expression<Func<T,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<T,TInner,TResult>>) Method .....	1944
Join<TInner,TKey,TResult>(ObservableCollection<TInner>,Expression<Func<T,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<T,TInner,TResult>>) Method .....	1946
OrderBy<TKey> Method .....	1948
OrderByDescending<TKey> Method .....	1949
Select<TResult> Method .....	1950
SelectMany Method .....	1951
SelectMany<TCollection,TResult>(Expression<Func<T,IObservableSource<TCollection>>>,Expression<Func<T,TCollection,TResult>>) Method .....	1952
SelectMany<TResult>(Expression<Func<T,IObservableSource<TResult>>>) Method .....	1954
SelectMany<TCollection,TResult>(Expression<Func<T,ObservableCollection<TCollection>>>,Expression<Func<T,TCollection,TResult>>) Method ..	1955
SelectMany<TResult>(Expression<Func<T,ObservableCollection<TResult>>>) Method .....	1957
ToString Method .....	1958
Union Method .....	1958
Union(IObservableSource<T>) Method .....	1959
Union(ObservableCollection<T>) Method .....	1960
Where Method .....	1961
Properties .....	1962

Item Property .....	1963
Events.....	1964
Changed Event .....	1965
ViewRow .....	1965
Overview .....	1966
Members.....	1967
Methods.....	1969
BeginEdit Method .....	1970
CancelEdit Method.....	1971
Delete Method .....	1971
EndEdit Method .....	1972
OnPropertyChanged Method .....	1973
ToString Method .....	1974
Properties.....	1974
Item Property .....	1975
Item(Int32) Property .....	1976
Item(String) Property .....	1977
RowState Property.....	1978
Tag Property.....	1979
Value Property .....	1980
View Property .....	1980
Events.....	1981
PropertyChanged Event .....	1981
ViewRowAddingEventArgs.....	1982
Overview .....	1983
Members.....	1984
Properties.....	1984
Row Property .....	1985
ViewRowCollection .....	1985
Overview .....	1987
Members.....	1988
Methods.....	1990
Clear Method .....	1991
Contains Method .....	1992

CreateRow Method.....	1993
DeferRefresh Method .....	1994
GetEnumerator Method .....	1994
IndexOf Method.....	1995
Remove Method .....	1996
RemoveAt Method.....	1997
Properties.....	1998
AllowClear Property .....	1999
AllowEdit Property .....	2000
AllowNew Property .....	2000
AllowRemove Property .....	2001
Count Property.....	2002
GroupDescriptions Property .....	2003
Groups Property.....	2003
Item Property .....	2004
Properties Property.....	2005
SortDescriptions Property.....	2006
Events.....	2006
Changed Event .....	2007
ViewRowAdding Event.....	2008
ViewRowPropertyInfo.....	2009
Overview .....	2010
Members.....	2010
Properties.....	2011
ImmediateUpdate Property .....	2011
PropertyName Property.....	2012
Enumerations.....	2013
ViewMaintenanceMode .....	2013
ViewOrder .....	2014
ViewRowState .....	2015
C1.LiveLinq.LiveViews.Xml Namespace .....	2017
Overview .....	2017
Classes.....	2017
XmlExtensions .....	2017



Overview .....	2018
Members.....	2019
Methods.....	2021
AsLive Method .....	2022
AsLive(XDocument) Method .....	2022
AsLive(XElement) Method .....	2024
Attributes Method .....	2025
Attributes(View<XElement>) Method .....	2025
Attributes(View<XElement>,XName) Method.....	2026
BeginUpdate Method.....	2028
DescendantNodes<T> Method .....	2029
DescendantNodesAndSelf Method.....	2030
Descendants Method.....	2031
Descendants<T>(View<T>) Method.....	2032
Descendants<T>(View<T>,XName) Method .....	2033
DescendantsAndSelf Method .....	2035
DescendantsAndSelf(View<XElement>) Method .....	2035
DescendantsAndSelf(View<XElement>,XName) Method .....	2036
Elements Method.....	2038
Elements<T>(View<T>) Method .....	2038
Elements<T>(View<T>,XName) Method.....	2040
EndUpdate Method .....	2041
Nodes<T> Method .....	2042
Root Method.....	2043
C1.LiveLinq.Metadata Namespace .....	2044
Overview .....	2044
C1.WPF.Data.Entity.4 Assembly .....	2045
Namespaces.....	2045
C1.WPF.Data Namespace .....	2045
Overview .....	2045
Classes.....	2045
ControlHandler.....	2045
Overview .....	2046
Members.....	2047

ControlHandler Constructor .....	2050
Properties.....	2050
DataSource Property.....	2051
Fields .....	2052
DataSourceProperty Field .....	2052
C1.WPF.Data.Entities Namespace .....	2053
Overview .....	2053
Classes.....	2054
C1DataSource.....	2054
Overview .....	2055
Members.....	2056
C1DataSource Constructor .....	2080
Methods.....	2081
GetControlHandler Method.....	2091
Load Method.....	2092
Refresh Method .....	2093
RejectChanges Method.....	2093
SaveChanges Method .....	2094
SetControlHandler Method .....	2095
Properties.....	2096
ClientCache Property .....	2103
ClientScope Property .....	2104
ContextType Property.....	2104
DbContext Property .....	2105
Item Property.....	2106
Item(String) Property .....	2106
Item(Int32) Property .....	2107
ObjectContext Property .....	2108
RefreshInterval Property.....	2109
ViewSources Property.....	2110
Fields .....	2110
ControlHandlerProperty Field.....	2111
Events.....	2112
SavedChanges Event .....	2118

SavingChanges Event .....	2119
EntityViewSourceCollection.....	2120
Overview .....	2121
Members.....	2122
EntityViewSourceCollection Constructor.....	2125
Properties.....	2126
Item Property .....	2127

# DataSource for Entity Framework

## Overview

**DataSource for Entity Framework** adds ease-of-use and performance enhancements to the Entity Framework and RIA Services. It improves and simplifies data binding with these frameworks by solving common problems related to loading, paging, filtering and saving data. It also provides performance enhancements such as fast loading and transparent scrolling over large datasets with Virtual Mode.

## Help with WPF Edition

[DataSource for Entity Framework Overview](#) > Help with WPF Edition

### Getting Started

For information on installing ComponentOne Studio WPF Edition, licensing, technical support, namespaces and creating a project with the control, please visit [Getting Started with WPF Edition](#)

## Key Features

The following are some of the main features of **DataSource for Entity Framework** that you may find useful:

 **Note:** This version of **DataSource for Entity Framework** requires Entity Framework 6 or above, .NET Framework 4.5 or above, and Visual Studio 2012 or above.

- **Entity Framework Made Easy with Design-time Components**

C1DataSource allows you to set up your data sources directly on the designer surface, with easy-to-use property dialogs and very little code to write. Configure the C1DataSource control and apply server-side filter, sort and group descriptors quickly at design-time. Of course, if you prefer to do everything in code you have the freedom of doing so using the rich data class libraries.

- **Live Views with LiveLinq**

LINQ is the perfect tool for transforming raw data into custom views. C1DataSource makes LINQ even more powerful by making LINQ query statements live. C1DataSource includes LiveLinq, an extension library which augments the functionality of LINQ to speed up queries and provide live views. With LiveLinq you can shape your view however you want using LINQ operators without losing full updatability and bindability. "Bindability" means that your views

are not just static snapshots of their source data. They are “live” and automatically reflect changes in the data. With LiveLinq your query results are kept up-to-date without re-populating every time changes in your data occur.

- **Simplifies MVVM**

C1DataSource can simplify programming with the widely-adopted Model-View-ViewModel pattern known as MVVM. You have to write a lot more code to develop MVVM applications because of the additional code layer, the ViewModel, and the synchronization between the Model and ViewModel data members. With DataSource you can use live views as your ViewModel and not have to worry about writing any synchronization code. Live views are synchronized automatically with their sources and are much easier to create. You can use C1DataSource with any MVVM framework.

- **Virtual Mode for Large DataSets**

Handle infinitely large datasets with C1DataSource’s Virtual Mode. Virtual Mode allows you to navigate through large datasets asynchronously. It works like paging on the data layer but the user can scroll through the data as if all rows were on the client. As the user scrolls, chunks of data are retrieved from the source page by page and disposed of as necessary. You can use Virtual Mode with any GUI control, such as a DataGrid, C1FlexGrid, or any control you want. Data can also be modified. This feature is transparent to the developer; you can turn on Virtual Mode with one simple property setting.

- **Paging with Data Modification**

For applications that prefer a paging interface, C1DataSource also supports paging without any limitations on data modification. That means users can make changes on multiple pages in one session before having to push the changes back to the database. This is a substantial improvement over other paging implementations such as with the Microsoft RIA Services DomainDataSource. Paging with C1DataSource is also supported in WinForms where paging is not provided.

- **Smart Client-side Caching**

The key to most of DataSource for Entity Framework’ features is its built-in client-side data cache. C1DataSource maintains a cache of entities on the client. When new queries are executed, it does not necessarily go to the server. It checks the client-side cache first and will not go to the server if it can find the result in the cache. This significantly improves performance and speed because it minimizes the number of trips to and from the server.

- **Context Management**

With C1DataSource you can use a single data context for your entire application. This allows you to forget the troubles of programming against multiple contexts across multiple views. Without C1DataSource you might have multiple contexts which can be very difficult to write code when you need to use entities from different contexts together. This feature is made possible by the smart client-side cache.

- **Memory Management**

C1DataSource is optimized for both performance and memory consumption. It manages memory resources for you releasing old entity objects in the cache when necessary to prevent memory leaks. In doing so, it fully preserves consistency by maintaining required entities and entities modified by the user.

- **Server-side Filtering**

Data brought from the server to the client usually needs to be filtered, or restricted in some way, to avoid moving large amounts of data over the wire and heaping it on the client. C1DataSource makes this common task very simple. Instead of doing it manually in code, you can specify server-side filters as simple property settings on C1DataSource.

- **Client-Side Transactions**

DataSource for Entity Framework gives developers a simple and powerful mechanism for rolling back (cancelling) and accepting changes on the client without involving the server. C1DataSource makes it easy to implement Cancel/Undo and OK/Accept buttons anywhere, even in nested (child) dialog boxes and forms, modal and modeless.

- **Saving Modified Data**

C1DataSource's smart client caching makes it easy to write code that saves modified data back to the server. With just one line of code and one trip to the server, you can save multiple entities. DataSource does all of the heavy lifting; it maintains changed entities in the cache, and ensures the cache is always consistent, while also optimizing memory usage and performance.

- **Code-First Support at Design Time**

C1DataSource can also be used in code-first scenarios, without generating code from a model. If you need the C1DataSource "live" features, ensure that your entity classes implement the INotifyPropertyChanged interface and if their collection navigation properties exist, they use ObservableCollection interface.

- **Cross-Platform Support**

DataSource for Entity Framework includes a C1DataSource component which allows you to combine multiple client view sources using Entity Framework or RIA Services. C1DataSource is supported in WinForms (.NET 4.5 or above), WPF and Silverlight 4.

## C1DataSource Introduction

In designing the Entity Framework, Microsoft set out to create a platform agnostic means to allow developers to communicate easily with the database that underpins their desktop or server applications, and with WCF RIA Services, this principle was further extended to the Silverlight platform. Each of these frameworks provides developers with an almost ideal solution to promote data persistence (the controlled retrieval and return of data to an underlying database) but falls short of providing them with an easy way to create the application logic and interaction with bound GUI controls that forms such an intrinsic part of most applications that are being built nowadays. DataSource for Entity Framework has been specifically designed to

meet this shortfall. It enhances both frameworks by providing additional functionality and improving overall application performance. In short, it provides developers with the means to speed up the development of typical line of business applications and write less code to achieve their goal.

The following topics will examine these performance-enhancing features of the C1DataSource in more detail, but we'll begin by examining the critical improvements it makes to the two core frameworks.

## Unified Data Context

[C1DataSource Introduction](#) > Unified Data Context

In both the Entity Framework and WCF RIA Services, the developer is solely responsible for managing the contexts (sessions), creating them explicitly and often creating individual ones for each form in an application. Objects retrieved by one context bear no relation to those retrieved by another, even if they are essentially similar. This can create severe problems when different forms (or even different tabs on a tab control in a single form) need to interact. Up until now, the traditional way to solve these problems has been to repeatedly save data back to the underlying database and continually refresh the GUI to reflect any changes that have occurred. The result - a lot of repeated code and degradation in overall application performance brought about through repeated trips to the database server.

DataSource for Entity Framework provides a better solution by combining a unified data context and intelligent client cache. This means that an application needs only one data context which is available to all forms and controls therein. The intelligent client cache keeps track of all changes made to the context's objects, in essence enabling client-side transactional functionality, removing the need to continually save and refresh views, so application performance is improved and code simplified. Local data cache also enables client-side queries with full LINQ capabilities (filtering, sorting, grouping, and more) which would otherwise be impossible.



**Note:** DataSource for Entity Framework supports both kinds of Entity Framework contexts, the newer DbContext (the default context in Visual Studio) and the legacyObjectContext.

## More Powerful Data Binding

[C1DataSource Introduction](#) > More Powerful Data Binding

Out-of-the-box data binding support, both in Entity Framework and in RIA Services, is limited to binding to the same query result that was originally retrieved from the server. In other words, you can neither bind to a subset of, nor reshape in any way, the original query. Also, an out-of-the-box data source control (DomainDataSource) is available only for RIA Services in Silverlight, and it has serious limitations such as disallowing paging and filtering unless all data changes are committed to the database.

DataSource for Entity Framework provides data source controls for direct access to Entity Framework, for both WPF and for WinForms, as well as for RIA Services (free from the standard DomainDataSource limitations).

Apart from the data source controls, C1DataSource supports creating live views that allow easy declarative reshaping of collections for two-way live data binding, which includes not just sorting and filtering but also calculated fields and any LINQ operations. Using live views, developers can express a significant part, if not all, of application logic with declarative data binding producing shorter and more reliable code. Also, by defining live views over entity collections, a developer can create a view model layer in the application making it follow the popular MVVM (model-view-view model) pattern with very little code for creating a view model and no code at all for synchronizing it with the model, a task that without C1DataSource would require extensive manual coding.

## Virtual Data Access

[C1DataSource Introduction](#) > Virtual Data Access

With its unique virtual mode feature, DataSource for Entity Framework supports access and data binding to large data sets, ranging from thousand to millions of rows, without awkward paging. In this mode, C1DataSource automatically retrieves rows from the database when they are needed for display in bound controls and then disposes of them when they are no longer required. That way the large data set appears to be in memory and ready for data binding without additional code and without consuming more memory resources than is necessary for displaying the rows currently shown by the bound controls.

The most common way of dealing with large data sets is to use paging. Paging usually provides for a poor user experience compared with data grids and other GUI controls directly bound to the data source. Virtual mode makes it possible to bind rich GUI controls directly to large data sets without paging.

## WPF Quick Start

Getting started with **DataSource for Entity Framework** only takes a few simple steps. You need to add a data source to your project, connect it to a C1DataSource component, and add a grid for displaying the data.

This quick start tutorial uses the Northwind database that is installed with the product at the following location:

**Documents\ComponentOne Samples\WPF**

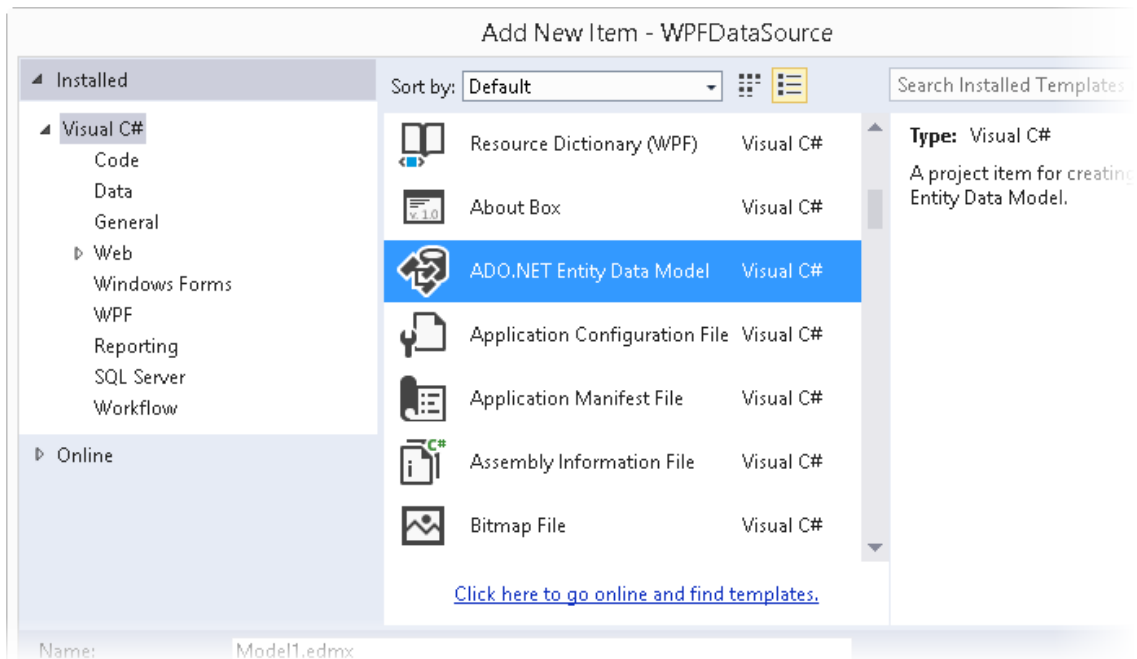
## Step 1 of 4: Adding a Data Source

[WPF Quick Start](#) > Step 1 of 4: Adding a Data Source

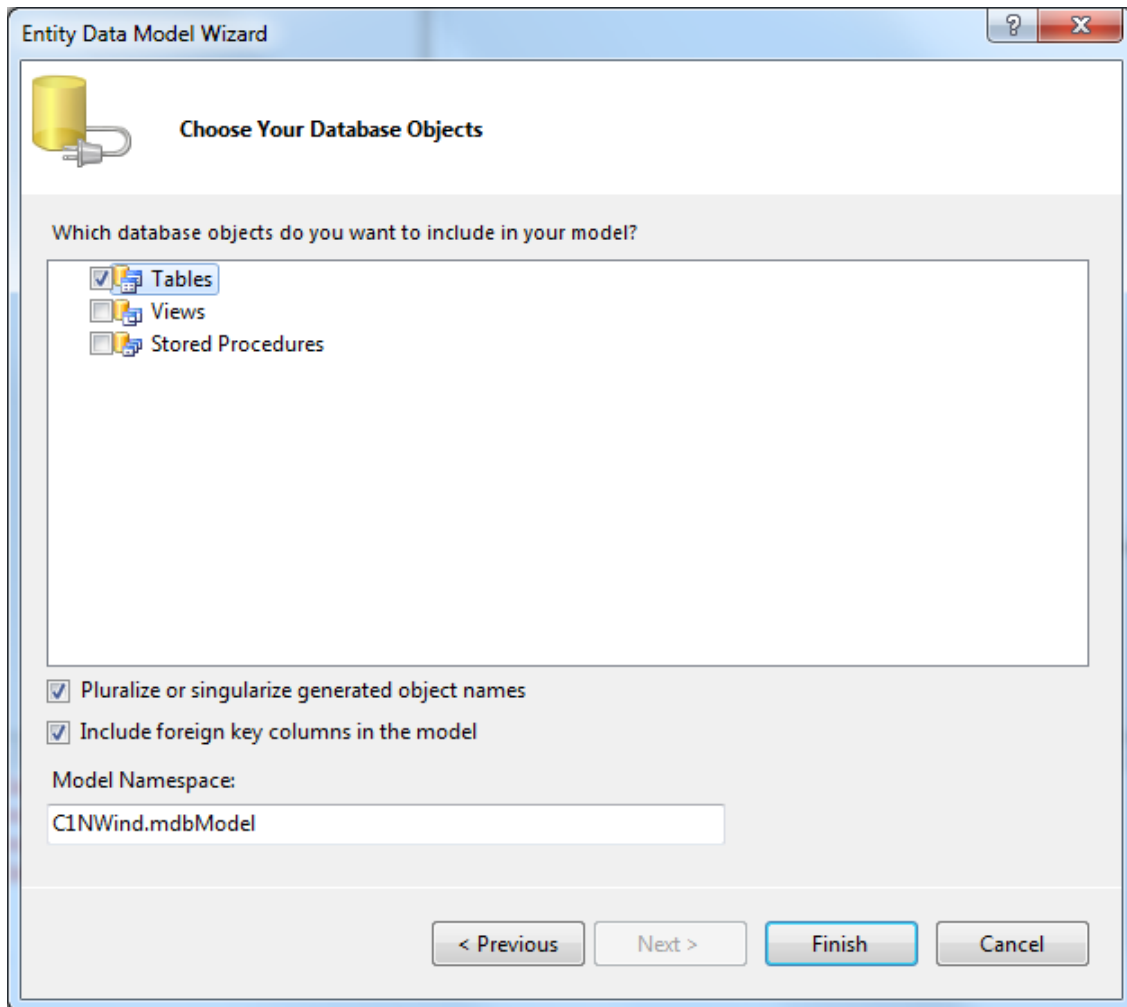


Begin by creating a new WPF project in Visual Studio. Then you will add a connection to the Northwind database.


1. In Visual Studio, choose **File | New | Project**.
2. Select the WPF Application template and click **OK**.
3. Next, add an Entity Data Model based on the Northwind database. This model will provide the data for the entire application.
4. In the Solution Explorer, right-click the project name and select **Add | New Item**.
5. Choose the **ADO.NET Entity Data Model** item and then click **Add**.



6. In the **Entity Data Model Wizard**, select **Generate** from database to generate the model from the Northwind database, and then click **Next**.
7. Click the **New Connection** button.
8. In the **Choose Data Source** dialog box, select **Microsoft SQL Server Database File** and click **Continue**.
9. Browse to find the NORTHWND.MDF, select it, and click **Open**. The **NORTHWND.MDF** is installed with the product at the following location:  
**Documents\ComponentOne Samples\Common**
10. Click **OK** and then click **Next**.
11. In the **Choose Your Database Objects** window, select **Tables** and click **Finish**. Note that if a Security Warning dialog box appears click **OK**.



12. In the Solution Explorer, expand the Model1.edmx node.
13. Delete the Model1.Context.tt file.
14. Delete the Model1.tt file.
15. In the Model Browser, right click the Diagram and select Add Code Generation Item from the context menu.
16. In the **Add New Item Dialog** box, select 'ComponentOne EF 6.x DbContext Generator'.

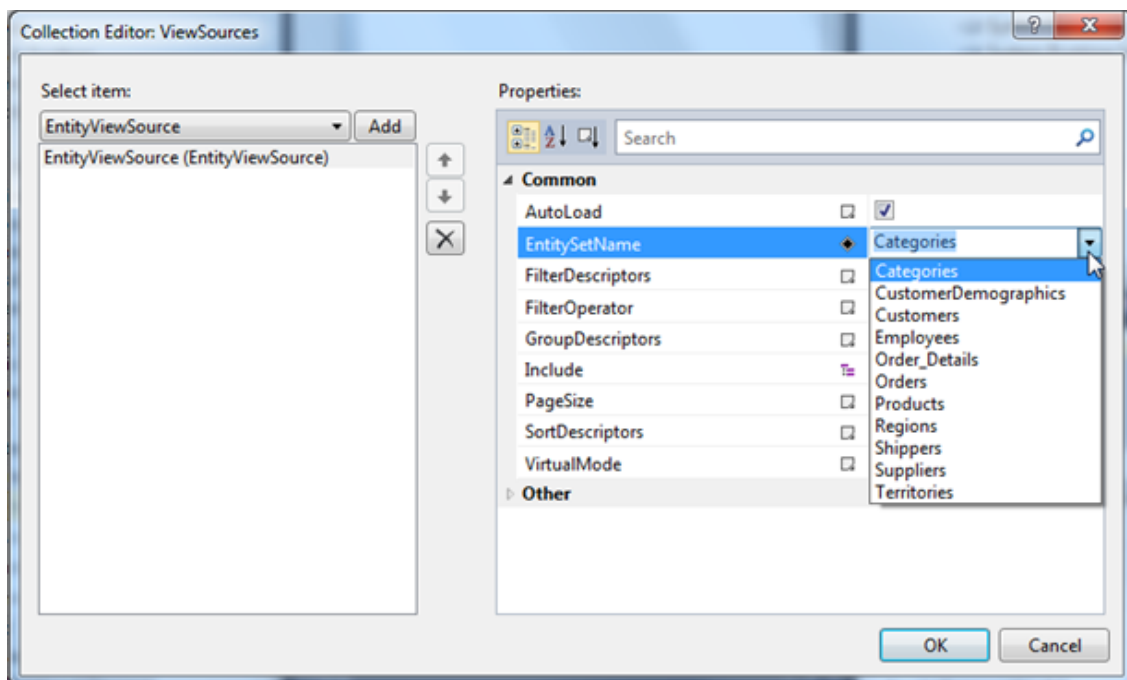
 **Note:** When you install DataSource for Entity Framework, ComponentOne EF6.x DbContext Code Generation Templates will be added to each version of Visual Studio that C1DataSource supports. These templates ensure that the DbContext models that you create, provide entities that support INotifyPropertyChanged.

Now build the project so the new Entity Data Model classes are generated and become available throughout the project. Once they are available, you will be able to connect the data to the C1DataSource component.

## Step 2 of 4: Connecting to the DataSource

[WPF Quick Start](#) > Step 2 of 4: Connecting to the DataSource

1. In this step, you'll add a C1DataSource component to the window (or page) and connect it to the Categories table of the data source. Double click the MainWindow.xaml in your Solution Explorer to open the page.
2. Drag a C1DataSource component from the Toolbox onto the window and name it c1DataSource1. This is a non-visual component, so it can be placed anywhere within the window's inner content. Note that if it does not appear in the toolbox, you can right-click the toolbox and choose Add Items. In the Choose Toolbox Items select WPF Components and click Browse to browse to the C1.WPF.Data.Entity.4.dll and then add it.
3. In the Properties window, set the C1DataSource's ObjectContextType property to the available item in the drop-down list. It should be something similar to AppName.NORTHWINDEntities.
4. Click the ellipsis button next to the ViewSources property to open the ViewSources Collection Editor.
5. Click Add and set the EntitySetName property to Categories.



6. Click OK to close the editor.

With the database connected to C1DataSource, all you need to do is add a grid to display the data.

## Step 3 of 4: Adding a Grid

[WPF Quick Start](#) > Step 3 of 4: Adding a Grid

In this step you will add a grid that will be used to display the data in the *Categories* table of the Northwind database. You can use **C1FlexGrid** or any grid you are familiar with, but in this example, we'll use a **DataGrid** control.

1. Drag a DataGrid control from the Toolbox onto your window.
2. Specify a binding for the DataGrid's ItemsSource property in XAML:

```
ItemsSource="{Binding [Categories], ElementName=c1DataSource1}"
```

3. Set the DataGrid's AutoGenerateColumns property to True; otherwise, the grid will not have any columns at run time.

Now simply run the project to view the grid!

## Step 4 of 4: Running the Project

[WPF Quick Start](#) > Step 4 of 4: Running the Project

Press **F5** to run the project. The data from the *Categories* table of the Northwind database is displayed.

CategoryID	CategoryName	Description
1	Beverages	Soft drinks, coffees, teas, beers, and ales
2	Condiments	Sweet and savory sauces, relishes, spreads, and se
3	Confections	Desserts, candies, and sweet breads
4	Dairy Products	Cheeses
5	Grains/Cereals	Breads, crackers, pasta, and cereal
6	Meat/Poultry	Prepared meats
7	Produce	Dried fruit and bean curd
8	Seafood	Seaweed and fish

# DataSource for Entity Framework in WPF

DataSource for Entity Framework includes a **C1DataSource** which facilitates Rapid Application Development by allowing developers to specify data sources on a rich designer surface that requires little or no additional coding. In addition, the C1DataSource includes classes that provide support for the same features that can be controlled through the designer surface, plus many extra features that provide greater control over it through code.

We'll begin our exploration of the C1DataSource component by looking at how we can control it via the designer surface. From there, we'll explore how it can be controlled dynamically at run time. Not all features available at run time will be represented here. You can consult the "Programming Guide" of this documentation and the Reference for more runtime features, including client-side transaction support and more.

## Simple Binding

[DataSource for Entity Framework in WPF](#) > Simple Binding

We'll begin by creating a new Windows Forms project in Visual Studio.

1. In Visual Studio, choose File | New Project.
2. Select the Windows Forms Application template and click OK.
3. Next, add an Entity Data Model based on the Northwind database. This model will provide the data for the entire application:
4. In the Solution Explorer, right-click the project name and select Add | New Item.
5. In the Data category, select the "ADO.NET Data Model" item and then click Add.
6. Use the Entity Data Model Wizard to select the database you want to use. In this example, you will use "NORTHWND.mdf", which should be installed in the Studio for WinForms\C1DataSource\Data folder.
7. In the Solution Explorer, expand the mode next to model1.edmx.
8. Delete the Model1.Context.tt file and the Model1.tt file.
9. Right click the model diagram and select Add Code Generation Item from the context menu. Select 'ComponentOne EF 6.x DbContext Generator' from the Add Code Generation Item Dialog box.



**Note:** When you install DataSource for Entity Framework, ComponentOne EF6.x DbContext Code Generation Templates will be added to each version of Visual Studio that C1DataSource supports. These templates ensure that the DbContext models that you create, provide entities that support INotifyPropertyChanged.

10. Now build the project so the new Entity Data Model classes are generated and become available throughout the project.
11. Next, add a C1DataSource component to the application and connect it to the Entity Data Model:

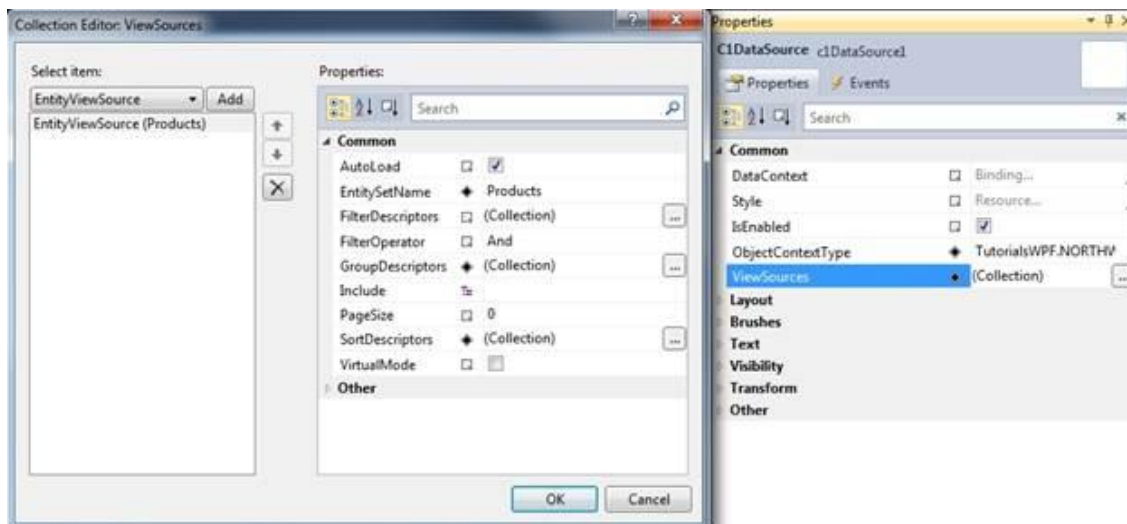
12. Drag a C1DataSource component from the Toolbox onto the form. This is a non-visual component, so it will appear on the tray below the form area rather than on the form itself.
13. Select the new component and choose View | Properties Window.
14. In the Properties window, set the ContextType property to the type of object context you want to use. In this case, there should be only one option in the drop-down list, something similar to "AppName.NORTHWINDEntities".

At this point, the C1DataSource has created an application-wide object (an EntityDataCache) that represents the Northwind database and has application scope. Additional C1DataSource objects on other forms will share that same object. As long as they are part of the same application, all C1DataSource objects share the sameObjectContext.

This unified object context is one of the main advantages C1DataSource provides. Without it, you would have to create multiple object contexts throughout the application, and each would have to be synchronized with the others and with the underlying database separately. This would be a non-trivial task, and any errors could compromise the integrity of the data. The unified object context handles that for you transparently. It efficiently caches data and makes it available to all views in a safe, consistent way.

Now that our C1DataSource has an ObjectContext to work with, we will go on to specify the entity sets it will expose to the application through its ViewSources collection. Note that if you are familiar with ADO.NET, you can think of the C1DataSource as a DataSet and the ViewSources collection as DataView objects.

15. In the properties of the C1DataSource, locate the ViewSourcesCollection property and open its editor dialog. Click Add and then select Products from the EntitySetName drop-down list.

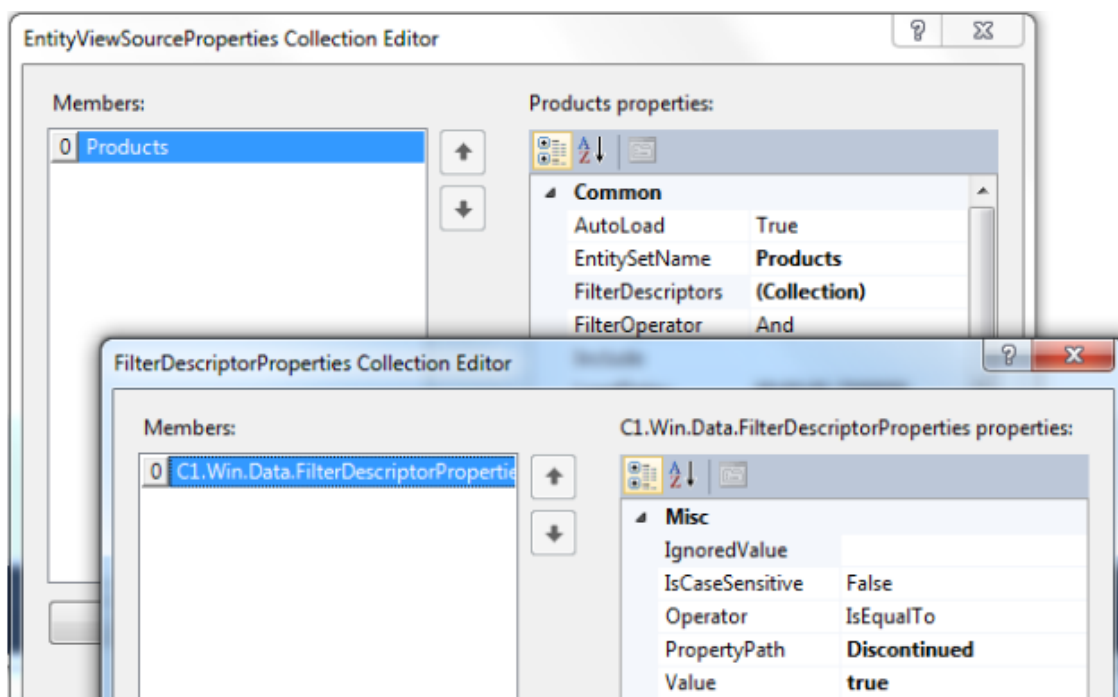


For this simple example, Products is all that is really necessary, but we could continue to create ViewSources within this C1DataSource in exactly the same way. We might, for example, create a ViewSource based on the Categories entity set allowing us to have a form that would be used to show the master-detail relationship between Categories and their Products. Conversely, there is no need to define all of the ViewSources that you might need in one single C1DataSource.

You could have a separate **C1DataSource** for each **ViewSource** that you need. All you need to do is ensure that the **C1DataSource** components that you use utilize the same **ContextType**.

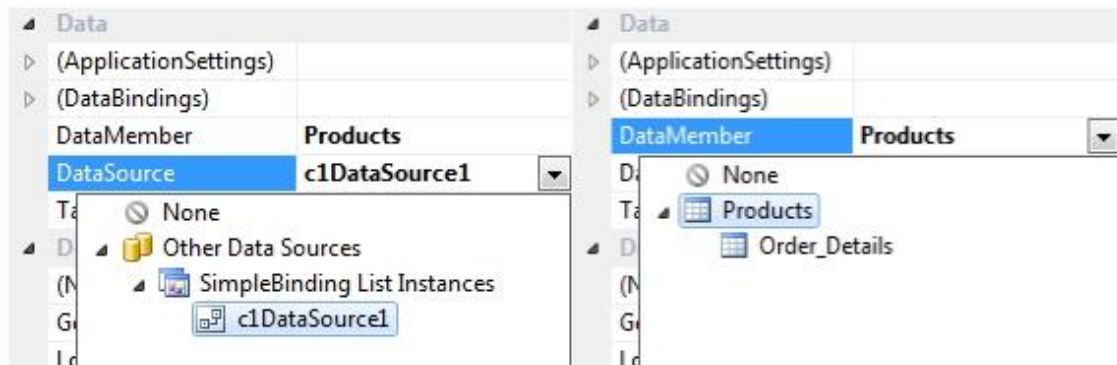
In reality we'll only want to bring a small subsection of the data contained within our database back to the client at any one time. This avoids overloading the client and network and also ensures that we are only presenting data that is relevant to the task our end user is engaged in. The traditional approach to this has been to create code (typically SQL queries) that we would run against the database to achieve our desired results. With **C1DataSource**, we can make use of the designer surface to achieve our goal without the need to write code by specifying server-side filters as property settings.

16. From the **ViewSourceCollection** editor, open the **FilterDescriptor** collection editor, add a filter descriptor and type the property name and a value for it for server-side filtering. If you need to filter more than one property, you can add additional filter descriptors.



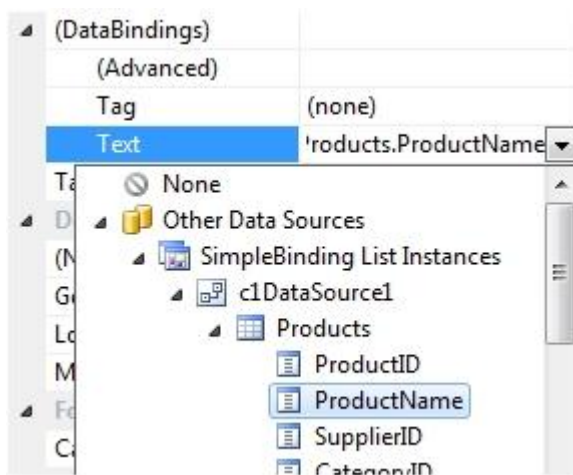
Using exactly the same methodology, you can add **SortDescriptors** and **GroupDescriptors** to provide sorting and grouping on the data you retrieve.

With our **C1DataSource** configured, add a grid control to the form. This can be **DataGridView**, a **C1FlexGrid** or indeed any grid that you are familiar with. Set the grid's **DataSource** property to the name of the **C1DataSource** (if you haven't specifically named the **C1DataSource**, then this will be *c1DataSource1*) and its **DataMember** property to *Products*. The **DataMember** property will, in fact, display a drop-down list of all the **ViewSources** (or Entity Sets) that we defined for the **C1DataSource**.



At this point, the grid will automatically generate columns for all the fields in the *Product* type, and most grids will allow you to further customize these columns and their layout via their built-in designers.

You could continue to add more controls to this form and bind them to specific items in the *Products* collection. To illustrate this point, add a **TextBox** control to the form. From the Properties window, expand the **DataBindings** section and bind its **Text** property to the *ProductName*, as illustrated.



Save, build and run the application again, and this time notice how the name of the product currently selected in the grid appears in the **TextBox** that you have just added to the form. If you then edit the product name in either control, the change will be immediately reflected in the other.

## Server-Side Filtering

[DataSource for Entity Framework in WPF](#) > Server-Side Filtering

We already mentioned that it is generally desirable to restrict the data returned from the server to the client and we demonstrated how `C1DataSource|tag=C1DataSource_Class` facilitates this with the use of the **FilterDescriptor Collection Editor**. Now we'll demonstrate how to provide the end user with the means to achieve server-side filtering.

The user will select a *Product Category* from a combo box, for example, although other GUI controls can be used, and that will load a **DataGrid** with new data from the server.



To implement server-side filtering, follow these steps:

1. Add a new form with a C1DataSource component to the project used in Simple Binding|document=WordDocuments\C1DataStudio-WPF.docx;topic=Simple Binding. You can make this form the project's start up form to save time when you run the project.
2. Establish the C1DataSource as before, but this time define two view sources: Categories and Products. Click the Add button twice in the ViewSourceCollection. Enter Categories next to the EntityViewSourceProperties.EntitySetName|keyword=EntityViewSource.EntitySetName property property for the first member (0) and enter Products for the second member (1).
3. Note that we left Value empty. This is because we will be setting it in code in response to a selection change event of the combo box.
4. Bind the grid using ItemsSource="{Binding ElementName=c1DataSource1, Path=Products}" as was shown earlier.
5. Set the AutoGenerateColumns property of the grid to True in XAML. Otherwise the grid will not have any columns at run time.

<DataGrid AutoGenerateColumns="True" ...

6. For the combo box, use the following bindings:

```
ItemsSource="{Binding ElementName=c1DataSource1, Path=Categories}"
DisplayMemberPath="CategoryName"
```

Just as with the grid and other controls, you can use the binding designer in the Properties window where you can choose from lists, or you can enter the binding directly in XAML.

7. Finally, add the following code to the form to handle the combo box's SelectionChanged event:

Visual Basic	Copy Code
<pre>Private Sub comboBox1_SelectionChanged(sender As System.Object, e As System.Windows.Controls.SelectionChangedEventArgs)     c1DataSource1.ViewSources("Products").FilterDescriptors(0).Value =         CType(comboBox1.SelectedValue, Category).CategoryID     c1DataSource1.ViewSources("Products").Load() End Sub</pre>	

C#	Copy Code
<pre>private void comboBox1_SelectionChanged(object sender,</pre>	

```

SelectionChangedEventArgs e)
{
    c1DataSource1.ViewSources["Products"].FilterDescriptors[0].Value =
        ((Category)comboBox1.SelectedItem).CategoryID;
    c1DataSource1.ViewSources["Products"].Load();
}

```


8. Save, build and run the application. Select a category in the combo box and notice how products related to that category are displayed in the grid. You can still edit the data in the grid exactly as before.

## The Power of Client Data Cache

[DataSource for Entity Framework in WPF](#) > The Power of Client Data Cache

The Server-Side Filtering|document=WordDocuments\C1DataStudio-WPF.docx;topic=Server-Side Filtering example shows how the **Entity Framework DataSource (EF DataSource)** improves and simplifies working with the Entity Framework in applications, made possible by its client-side data cache, a key feature of **EF DataSource**. It enables several important enhancements and makes writing application code much easier.

Let's start with a performance enhancement that can be seen in the Server-Side Filtering|document=WordDocuments\C1DataStudio-WPF.docx;topic=Server-Side Filtering example. When the user switches between categories, the grid loads products associated with them. The first time a category is chosen, there is a slight delay when the relevant data is retrieved. On subsequent occasions, when the same category is chosen, data retrieval is virtually instantaneous. Why? Because the data is being retrieved from the memory EntityDataCache rather than the server.

 **Note:** The EntityDataCache is actually smarter still, determining that a return trip to the server can be avoided, even in more complex cases where the queries may not be identical but can be fulfilled from the results of other queries already contained therein.

This performance enhancement may not be obvious if you are working on a single machine where no network interaction is required, but in real world applications it can make all the difference between a sluggish application that calls the server on every user action and a crisp interface with no delays.

The second important enhancement is in memory management. You might think based on what you've read and observed to date that the EntityDataCache continually accumulates data throughout its existence. If this were the case you would very quickly witness severe performance degradation. In reality the EntityDataCache is keeping track of what is stored within it and as data is no longer required is releasing it performing a self-cleansing operation at the same time. All of this is being done without the need for you to add extra code, and more importantly still it is ensuring that data integrity is being preserved at all times. Data

won't be released if required by other data, nor will data that has been altered in some way without those alterations having being saved.

This also relates to performance, because getting rid of unnecessary objects in memory improves performance, and vice versa, keeping large numbers of obsolete objects in memory leads to performance degradation. We mentioned before how **EF DataSource** simplifies context management by eliminating the need to create multiple data contexts thanks to the client cache. Now we should explain what the **EntityDataCache** actually is and how we can further use this information to our advantage.

The **EntityDataCache** is essentially the context. In terms of the **EF DataSource** namespaces, the cache is the **C1.Data.Entities.EntityClientCache** class, and it is in one-to-one correspondence with **ObjectContext** through its **ObjectContext** property. Both the cache and its underlying **ObjectContext** are created for you automatically if you use the **C1DataSource** control, however, you can create them explicitly and set the **C1DataSource.ClientCache** property in code, if necessary.

To see how simple application code becomes thanks to the client-side cache, let's add the functionality of saving modified data to the project in Server-Side Filtering|document=WordDocuments\C1DataStudio-WPF.docx;topic=Server-Side Filtering.

1. Simply add a button to the form and add a handler for the code:

Visual Basic	Copy Code
<pre>Private Sub button1_Click(sender As System.Object, e As System.Windows.RoutedEventArgs)     c1DataSource1.ClientCache.SaveChanges() End Sub</pre>	

C#	Copy Code
<pre>private void button1_Click(object sender, RoutedEventArgs e) {     c1DataSource1.ClientCache.SaveChanges(); }</pre>	

2. Save, build and run the application.
  - Select a category and then make some changes to the product information in the grid.
  - Select another category (and possibly a third) and again makes changes to the product details in the grid.

- Click the button you added, close the application, reopen it and select the same categories as before.

Observe how the changes you made have been saved. **EF DataSource** has provided, via the `EntityDataCache`, a way to alter the product details of several categories' products without the need to save those changes each time different category is selected. To achieve this effect without writing a lot of code, you'd need to keep the entities (the product details) from different categories in the same context. This would waste memory without releasing it, creating a memory leak. **EF DataSource** simplifies all of this for you while optimizing memory usage and performance.

Client-side caching also makes possible other important features of **EF DataSource**, such as client-side queries and, especially, live views. [Live](#)

[Views](#)`|document=WordDocuments\C1DataStudio-WPF.docx;topic=Live Views` is a feature that allows you to replace much of the complex application coding with simple data binding, which we'll learn more about later.

## Master-Detail Binding

[DataSource for Entity Framework in WPF](#) > Master-Detail Binding

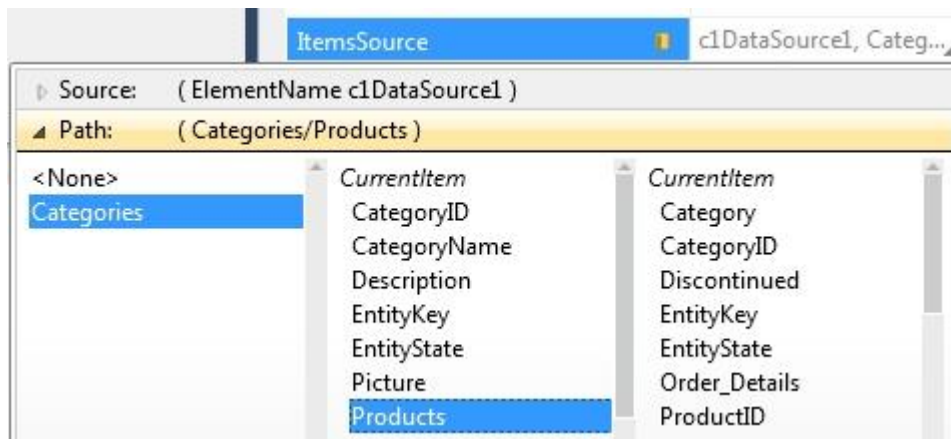
As we've already seen in the [Server-Side Filtering](#)`|document=WordDocuments\C1DataStudio-WPF.docx;topic=Server-Side Filtering` example, `C1DataSource` supports master-detail binding. With very large datasets, server-side filtering is by far the best solution, but with smaller datasets, client-side filtering can be just as efficient. This next scenario demonstrates client-side master-detail binding using a grid, instead of a combo box like in our previous example`|document=WordDocuments\C1DataStudio-WPF.docx;topic=Server-Side Filtering`, to select our categories.

To implement master-detail binding, follow these steps:

1. Using the project we created to demonstrate [Server-Side Filtering](#)`|document=WordDocuments\C1DataStudio-WPF.docx;topic=Server-Side Filtering``|document=WordDocuments\C1DataStudio-WPF.docx;topic=Server-Side Filtering`, add a new form with a `C1DataSource` component using the same `ObjectContextType` as before and create a `ViewSource` based on `Categories`. Note that you can make this the startup form to save time when you run the project.
2. Next, add the master grid to the form and bind it to `Categories`:

```
ItemsSource="{Binding ElementName=c1DataSource1, Path=Categories}"
```

3. Now add a second grid to the form below the one you've just configured. Its `DataSource` will again be the `C1DataSource`, but set its `DataMember` property to the `Products` node which you'll find underneath `Categories` as shown in the following picture:



Or you can type it directly in XAML:

```
ItemsSource="{Binding ElementName=c1DataSource1, Path=Categories/Products}"
```

4. Save, build and run the application.

This form of the **Path** parameter in binding, using `"/"`, `Path=Categories/Products`, causes the grid to automatically show the products that belong to the category that is currently selected. Running this on a single machine, as you probably are, you won't notice any significant time lapse as you select new categories in the master grid, and their respective products are displayed in the details grid. In the background, the **C1DataSource** is making use of an Entity Framework feature, implicit lazy loading, meaning that products are only being summoned for new categories as they are selected. For many scenarios, this is perfectly acceptable, but we began this section by specifically referring to master-detail relationships in small datasets. We might just as well fetch all of the products for all of the categories when the form loads and then display will be instantaneous whether on a single machine or across a network. To achieve this behavior, open the **ViewSourceCollection** editor and type *Products* in the **Include** property of the *Categories* view source.

## Large DataSets: Paging

[DataSource for Entity Framework in WPF](#) > Large DataSets: Paging

To show large amounts of data without bringing it all to the client at once, applications have traditionally used paging. Paging is not an ideal solution; it complicates the interface and makes it less convenient for the user, but it is preferred in some applications. For these cases, **C1DataSource** supports paging as expected, but it does so without the traditional limitations on data modification. The user can make changes in multiple pages in one session without being forced to send the changes from one page to the database before moving to the next page. That's a substantial enhancement compared to other paging implementations, such as the one in **DomainDataSource** in Microsoft RIA Services.



**Note: EF DataSource** does offer a solution to the drawbacks of paging; we'll cover virtual mode|document=WordDocuments\C1DataStudio-WPF.docx;topic=Large Datasets\ Virtual Mode later in this documentation.

To implement paging, follow these steps:

Using the project we created to demonstrate Master-Detail

Binding|document=WordDocuments\C1DataStudio-WPF.docx;topic=Master-Detail Binding, add a new form with a **C1DataSource** component using the same **ObjectContextType** as before. Note that you can make this the startup form to save time when you run the project.

1. Create a ViewSource in the ViewSourceCollection editor, entering Orders as the EntitySetName.
2. To enable paging, set the PageSize property to 10 for now, but you can choose any reasonable value for this property. It's simply determining the number of data rows that will be displayed on a page.
3. Next, add a grid to the form and bind it to Orders the same way we did before:

```
ItemsSource="{Binding ElementName=c1DataSource1, Path=Orders}"
```

4. Set the AutoGenerateColumns property of the grid to True in XAML.
5. To allow the user to move between pages, and to show the current page and page count, we need to add a few more controls. Add two buttons and a label using, for example, the following XAML:

```
<StackPanel Orientation="Horizontal" Grid.Row="1" Margin="2">
```

```
    <Button Padding="10,0,10,0" Margin="1" Content="&lt;"
    Click="MoveToPrevPage"/>
```

```
    <TextBlock VerticalAlignment="Center" Text="Page: "/>
```

```
    <TextBlock VerticalAlignment="Center" x:Name="pageInfo"/>
```

```
    <Button Padding="10,0,10,0" Margin="1" Content="&gt;"
    Click="MoveToNextPage"/>
```

```
</StackPanel>
```

6. Add the following code containing a RefreshPageInfo handler for the PropertyChanged event used to show current page number and page count, and handlers for the button Click events used to move to the next and previous pages:

Visual Basic	Copy Code
Imports C1.Data.DataSource	

```

Public Class Paging
    Private _view As ClientCollectionView
    Public Sub New()
        InitializeComponent()
        _view = C1DataSource1("Orders")
        RefreshPageInfo()
        AddHandler _view.PropertyChanged, AddressOf RefreshPageInfo
    End Sub
    Private Sub RefreshPageInfo()
        pageInfo.Text = String.Format("{0} / {1}", _view.PageIndex + 1,
        _view.PageCount)
    End Sub
    Private Sub btnPrevPage_Click(sender As System.Object, e As
System.EventArgs) Handles btnPrevPage.Click
        _view.MoveToPreviousPage()
    End Sub
    Private Sub btnNextPage_Click(sender As System.Object, e As
System.EventArgs) Handles btnNextPage.Click
        _view.MoveToNextPage()
    End Sub
End Class

```

C#

Copy Code

```

using C1.Data.DataSource;

namespace TutorialsWPF
{
    public partial class Paging : Window
    {
        ClientCollectionView _view;

        public Paging()
        {
            InitializeComponent();

            _view = c1DataSource1["Orders"];

            RefreshPageInfo();
            _view.PropertyChanged += delegate { RefreshPageInfo(); };
        }
    }
}

```

```

private void RefreshPageInfo()
{
    labelPage.Text = string.Format("{0} / {1}",
        _view.PageIndex + 1, _view.PageCount);
}

private void MoveToPrevPage(object sender, RoutedEventArgs e)
{
    _view.MoveToPreviousPage();
}

private void MoveToNextPage(object sender, RoutedEventArgs e)
{
    _view.MoveToNextPage();
}
}
}

```

7. Save, build and run the application. Page through the Orders. While you are moving between pages, try changing some data in the grid. Try changing data in one page, and then move to another page and try changing data there. Notice that C1DataSource allows you to do this without forcing you to save data to the database before leaving the page. This is an important enhancement compared with other paging implementations, including the one supported by DomainDataSource in Microsoft RIA Services (which is for Silverlight only, whereas EF DataSource supports this, as other features, for all three platforms: WinForms, WPF, Silverlight).
8. Try also to delete some orders. This is also allowed without restrictions, and moreover, the current page will automatically complete itself to keep the number of rows in the page unchanged.
9. And if you add a Save Changes button, using the following code as we did previously, then you will be able to save changes made in multiple pages by pressing that button when you are done.

Visual Basic	Copy Code
<pre> Private Sub btnSaveChanges_Click(sender As System.Object, e As System.Windows.RoutedEventArgs)     c1DataSource1.ClientCache.SaveChanges() End Sub </pre>	
C#	Copy Code



```
private void btnSaveChanges_Click(object sender, RoutedEventArgs e)
{
    c1DataSource1.ClientCache.SaveChanges();
}
```

All this functionality is what you would expect from a paging implementation that supports unrestricted data modification. Unfortunately, it is not easy to implement. For example, think of all possible cases where changing data on one page interferes with what should be shown on other pages, and so on. That is why paging usually imposes severe restrictions on data modifications, if they are at all allowed. For example, Microsoft `DomainDataSource` requires you to save all changes before changing pages. Fortunately, **EF DataSource** supports paging without restricting data modifications in any way.

As with many other features, unlimited modifiable paging is made possible by the client-side cache, see [The Power of Client Data Cache](#). That also means that paging implementation in **EF DataSource** is optimized both for performance and for memory consumption. The cache provides that optimization. It keeps recently visited pages in memory, so re-visiting them is usually instantaneous. And it manages memory resources, releasing old pages when necessary, to prevent memory leaks.

## Large Datasets: Virtual Mode

[DataSource for Entity Framework in WPF](#) > Large Datasets: Virtual Mode

As mentioned in the [Large Datasets: Paging](#)`|document=WordDocuments\C1DataStudio-WPF.docx;topic=Large Datasets\` Paging topic, **Entity Framework DataSource (EF DataSource)** has an even better solution than paging for dealing with large data and large numbers of rows.

What if using large datasets with thousands, or even millions, of rows was no longer a problem? What if you could use the same controls to display massive datasets as you would for smaller datasets, with no paging, no code changes, and all by setting one Boolean property? With **EF DataSource** you can, thanks to the magical **VirtualMode** property.

To implement virtual mode, follow these steps:

1. Using the project we created to demonstrate Large Datasets: Paging`|document=WordDocuments\C1DataStudio-WPF.docx;topic=Large Datasets\` Paging, add a new form with a `C1DataSource` component using the same `ObjectContextType` as before. Note that you can make this the startup form to save time when you run the project.
2. Create a `ViewSource` in the `ViewSourceCollection` editor. Use the largest table in our sample database: `Order_Details`.
3. Set the `VirtualMode` property to `Managed`. Another possible value is `Unmanaged`, but that is an advanced option that should be used with caution and only when necessary. The `Managed` option means that getting data from the server is managed by a grid control. With the `Managed` option, `EF DataSource` supports all main Microsoft and `ComponentOne` grid controls: `C1FlexGrid`, `C1DataGrid`, and `Microsoft DataGrid` for WPF. It is optimized for performance with those specific grid controls. The `Unmanaged`

option means that virtual mode is not driven by a specific control, will work with any bound controls but is subject to some limitations described here|document=WordDocuments\C1DataStudio-ProgrammingGuide.docx;topic=Unmanaged virtual mode limitations.

4. Add a grid to the designer and bind it to Order\_Details the same way we did it before:

```
ItemsSource="{Binding ElementName=c1DataSource1, Path=Order_Details}"
```

5. Set the AutoGenerateColumns property of the grid to True in XAML.

Now, since we selected the *Managed* option, we need to specify which grid control is driving the data. **EF DataSource** defines an attached property `C1DataSource.ControlHandler|tag=P_C1_WPF_Data_Entities_C1DataSource_ControlHandler` that is an object having properties affecting the control's behavior when it is bound to a `C1DataSource|keyword=C1DataSource` class. There is a boolean **VirtualMode** property in `C1DataSource.ControlHandler|tag=P_C1_WPF_Data_Entities_C1DataSource_ControlHandler` that marks the control as the main, "driving" control in *Managed* virtual mode.

6. Add the following inside the DataGrid markup in XAML (the added markup is shaded):

```
<DataGrid AutoGenerateColumns="True" Name="dataGrid1"
ItemsSource="{Binding ElementName=c1DataSource1, Path=Order_Details}">
  <c1:C1DataSource.ControlHandler>
    <c1:ControlHandler VirtualMode="True"/>
  </c1:C1DataSource.ControlHandler>

</DataGrid>
```

7. Save, build and run the application.

You'll see nothing earth-shattering, simply a grid you can navigate, scroll and modify row data as you see fit. It looks and behaves like any conventional data grid, which is precisely the point. Now you can use large datasets free of the drawbacks associated with paging, and all done without the need to write any code. And as an added benefit, you can use any GUI control you like so long as it has a **DataSource** property. This example has used a relatively modest-sized dataset, but **C1DataSource** running in virtual mode would be just as responsive with a much larger dataset; it does not depend on the number of rows. To further prove the point, look at the **OrdersDemo** sample installed with this product. The sample uses a larger database, with roughly 65,000 additional rows, but the responsiveness is the same as our example here. Again, it does not depend on the number of rows in the dataset.

How does this magic work? It works much like paging, only hiding its inner workings from the GUI controls, with paging occurring under the hood. The GUI controls see the data as if it is fetched to the client and ready for them. When GUI controls request data, **C1DataSource**, or **ClientViewSource** if it is used in code without a **C1DataSource** control, first checks whether it can serve the data from memory, from the same client-side cache it uses for all features. If it can't find it in memory, it transparently fetches the required data from the server. As with other features using the client-side cache, **C1DataSource** does not store the fetched data indefinitely, which would be a memory leak. It knows which parts of the data are needed for serving the GUI

controls and which parts should be kept because they are modified or related to modified parts, and so on. It releases old, unnecessary parts of data as needed. No code is required and any GUI controls can be used.

## Automatic Lookup Columns in Grids

[DataSource for Entity Framework in WPF](#) > Automatic Lookup Columns in Grids

A common scenario in data binding is for data classes to contain references to other data classes. For example, a **Product** object may contain references to **Category** and **Supplier** objects.

In ADO.NET, the references usually appear as foreign keys that map in other tables (e.g., **Product.CategoryID** and **Product.SupplierID**).

In the Entity Framework, you still get the key columns, but you also get the actual objects. So you have **Product.CategoryID** (usually an integer) and **Product.Category** (an actual Category object).

Displaying foreign keys in a grid is not very useful, because it is unlikely that users will remember that category 12 is "Dairy Products" or that supplier 15 is "ACME Imports". Allowing users to edit these keys would be even worse. A common way to work around this problem is to remove the related entity columns from any bound grids and, optionally, to replace them with custom columns that use combo boxes for editing the values, so called lookups. The combo boxes have to be bound to the related tables and have their properties set so they display relevant values (e.g., **Category.Name** or **Supplier.CompanyName**) and so they are synchronized with the values being displayed on the grid. This is not terribly hard to do, but it is a tedious and error-prone task that makes projects harder to create and maintain.

**C1DataSource** can do this tedious work for the developer; it can automatically change related entity columns so that they show combo box lookups. It can do this for several types of data grids, those that it supports. Currently supported WPF grids are: C1FlexGrid, C1DataGrid, and Microsoft DataGrid for WPF. Here we will show how to do this for **C1FlexGrid**.

C1DataSource provides an extender property called ControlHandler. If you place a C1FlexGrid control on a form that contains a C1DataSource, the grid will get an additional ControlHandler property. A ControlHandler is an object containing (at present) a single boolean property AutoLookup. This property set to True causes the C1DataSource to configure grid columns that contain references to other entities so that they show lookup combos.

To see how it works, follow these steps:

1. Using the project used in Simple Binding, add a new form with a C1DataSource component using the same ContextType as before.
2. Create a ViewSource in the ViewSourceCollection editor, entering Products as the EntitySetName.
3. Add a C1FlexGrid to the form and set its C1DataSource property to the C1DataSource and its DataMember property to Products.
4. Save, build and run the application. It will look like this:

Product ID ▲	Product Name	Category	Supplier
5	Chef Anton's Gumbo Mix	TutorialsWPF.Category	TutorialsWPF.Supplier
9	Mishi Kobe Niku	TutorialsWPF.Category	TutorialsWPF.Supplier
17	Alice Mutton	TutorialsWPF.Category	TutorialsWPF.Supplier
24	Guaraná Fantástica	TutorialsWPF.Category	TutorialsWPF.Supplier
28	Rössle Sauerkraut	TutorialsWPF.Category	TutorialsWPF.Supplier
29	Thüringer Rostbratwurst	TutorialsWPF.Category	TutorialsWPF.Supplier
42	Singaporean Hokkien Fried M	TutorialsWPF.Category	TutorialsWPF.Supplier
53	Perth Pasties	TutorialsWPF.Category	TutorialsWPF.Supplier

As you can see, the Category and Supplier columns are not useful at all. You could remove them or customize the grid by writing some code to create new columns, but there's an easier way.

5. Select the grid in the designer and find the property called "ControlHandler on c1DataSource1" in the Properties window.
6. Remember, this is an extender property and will be available only if there is a C1DataSource component on the form. Set the AutoLookup property there to True.
7. When you are done, run the project again and look at the Category and Supplier columns. Notice that now the grid shows the category name and supplier's company name instead of the generic strings. Also notice that you can edit the product's category and the product's supplier by picking from a drop-down list, complete with auto search functionality.

Product ID ▲	Product Name	Category
5	Chef Anton's Gumbo Mix	Condiments
9	Mishi Kobe Niku	Meat/Poultry
17	Alice Mutton	Meat/Poultry
24	Guaraná Fantástica	Beverages ▼
28	Rössle Sauerkraut	Condiments
29	Thüringer Rostbratwurst	Meat/Poultry
42	Singaporean Hokkien Fried Mee	Beverages
53	Perth Pasties	Produce
5	Chef Anton's Gumbo Mix	Grains/Cereals
9	Mishi Kobe Niku	Confections
17	Alice Mutton	Dairy Products
		Seafood

8. Finally, click the column headers to sort the grid by Category or Supplier and notice that the sorting is performed based on the value displayed on the grid. This is what anyone would expect, but surprisingly it is not easy to achieve using regular data binding.

The string value (name) shown by the combo box is determined following these rules:

1. If the entity class overrides the ToString method, then the string representation of the entity is obtained using the overridden ToString method. This should return a string that uniquely represents the entity. For example, it could be the content of a CompanyName column, or a combination of FirstName, LastName, and EmployeeID. This is the preferred method because it is simple, flexible, and easy to implement using partial classes (so your implementation will not be affected if the entity model is regenerated).
2. If the entity class does not override the ToString method, but one of its properties has the DefaultProperty attribute, then that property is used as the string representation of the entity.
3. If the entity class does not override the ToString method and has no properties marked with the DefaultProperty attribute, then the first column that contains the string "Name" or "Description" in its name will be used as a string representation for the entities.
4. If none of the above applies, then no lookup is created for the given entity type.

## Customizing View

[DataSource for Entity Framework in WPF](#) > Customizing View

In many situations, you may want to use custom views that do not correspond directly to the tables and views provided by the database. LINQ is the perfect tool for these situations allowing you to transform raw data using your language of choice using query statements that are flexible, concise, and efficient. **Entity Framework DataSource (EF DataSource)** makes LINQ even more powerful by making LINQ query statements *live*. That's why its LINQ implementation is called LiveLinq. We will show the power of LiveLinq in [Live Views](#)`|document=WordDocuments\C1DataStudio-WPF.docx;topic=Live Views`. For now, suffice it to say that you can transform your view to shape it whatever way you want using LINQ operators without losing full updatability and bindability.

Let's try, for example, one of the LINQ operators: **Select** (also called projection), to customize the fields (properties) of our view.

To customize a view, follow these steps:

1. Using the project we created to demonstrate Large Datasets: `Paging|document=WordDocuments\C1DataStudio-WPF.docx;topic=Large Datasets\`; Paging, add a new form with a C1DataSource component using the same ObjectContextType as before. Note that you can make this the startup form to save time when you run the project.
2. Create a ViewSource in the ViewSourceCollection editor. Use the Products table in our sample database.
3. Add a grid to the window designer and, as before, set its AutoGenerateColumns property to True. But this time we won't bind the grid to C1DataSource in XAML. Instead, we will bind it in code, because we will create a custom view in code.
4. Create a custom live view and bind the grid to that view with the following code:

Visual Basic

Copy Code

```


dataGrid1.ItemsSource = _
    (From p In c1DataSource1("Products").AsLive(Of Product)()
     Select New With
     {
         p.ProductID,
         p.ProductName,
         p.CategoryID,
         p.Category.CategoryName,
         p.SupplierID,
         .Supplier = p.Supplier.CompanyName,
         p.UnitPrice,
         p.QuantityPerUnit,
         p.UnitsInStock,
         p.UnitsOnOrder
     }).AsDynamic()

```

C#	Copy Code
<pre> dataGrid1.ItemsSource =     (from p in c1DataSource1["Products"].AsLive&lt;Product&gt;()      select new      {          p.ProductID,          p.ProductName,          p.CategoryID,          CategoryName = p.Category.CategoryName,          p.SupplierID,          Supplier = p.Supplier.CompanyName,          p.UnitPrice,          p.QuantityPerUnit,          p.UnitsInStock,          p.UnitsOnOrder      }).AsDynamic(); </pre>	

Here `c1DataSource1["Products"]` is a **ClientCollectionView** object. It is the view that is created by the view source that we set up in the designer (if you need to access the view source itself in code, it is also available, as `c1DataSource.ViewSources["Products"]`). The **AsLiveT\_Method**

method call is needed to specify the item type of the view (*Product*) so LiveLinq operators can be applied to it. The result, `c1DataSource1["Products"].AsLive<Product>()`, is a `View<Product>`. The **C1.LiveLinq.LiveViews.View** is the main class of LiveLinq, used for client-side live views. LINQ operators applied to live views preserve their updatability and bindability. They are the same usual LINQ operators, but the fact that they are applied to a live view gives them these additional features that are critically important for data binding applications.

 **Note:** `AsDynamic()` must be applied to this view because its result selector (the `select new...` code) uses an anonymous class. This is a minor LiveLinq limitation, only for anonymous classes. If you forget to add `AsDynamic()` to such view, you will be reminded by an exception.

5. Save, build and run the application.

The grid now shows the columns we defined in the 'select' clause of our LiveLinq view. Note also that all columns are modifiable. It may not seem like a big deal; however, it is an important feature that would be difficult to implement on your own for this customized view, especially when adding and deleting rows, which, as you can see here, is also supported. To try deleting a row, just select a row and press the **Delete** key. To try adding a row, you'll need to add a button to the window executing the following code:

Visual Basic	Copy Code
<pre>Dim newItem = CType(dataGrid1.ItemsSource, System.ComponentModel.IEditableCollectionView).AddNew()     dataGrid1.ScrollIntoView(dataGrid1.Rows.Count - 1, 0)</pre>	
C#	Copy Code
<pre>object newItem =  ((System.ComponentModel.IEditableCollectionView)dataGrid1.ItemsSource).AddNew (); dataGrid1.ScrollIntoView(newItem);</pre>	

This is needed only because Microsoft WPF DataGrid does not provide a built-in interface for adding new rows.

Bindability is achieved because the views are always 'live'; they aren't simple snapshots of static data. To prove the point, construct an exact replica of the form that we've just built. At this stage, you might find it easier to add a menu to your application to make accessing the forms you've created easier. Save, build and run the application. This time, open two instances of the

form you just created and change some data in the grid on one of the forms. Notice how the corresponding data in the grid on the other form is changed automatically. Remember, you have not had to write a single line of code for the grids in these forms to synchronize the changes that you've just made.

You will see more of what LiveLinq can do in the [Live Views](#)`|document=WordDocuments\C1DataStudio-WPF.docx;topic=Live Views topic.`

## Working with Data Sources in Code

[DataSource for Entity Framework in WPF](#) > Working with Data Sources in Code

Up to this point, we have been setting up data sources directly on the designer surface with very little code. **Entity Framework DataSource (EF DataSource)** has made it very easy, but sometimes you want or need to do everything in code. **EF DataSource** makes this possible as well. Everything we did previously can be done at run time in code.

An obvious way to go about this would be to use the **ClientViewSource** object that we have, in effect, been setting up in the designer as elements of the **ViewSourceCollection** of a **C1DataSource**, given that it can be created on its own without a **C1DataSource**. We could, however, take a step further back and use a lower level class **ClientView**. This would provide full control over loading data from the server and, since it is derived from `C1.LiveLinq.LiveViews.View<T>`, we can apply any LiveLinq operators to it. The ability to bind this to any GUI control whose datasource can be set to a `View<T>` also means that we'll end up with a fully editable view of our data.

Server-side filtering is, perhaps, the most common operation, because no one usually wants entire database tables brought to the client unrestricted. Earlier we saw how **EF DataSource** made it simple to perform without code, but now we'll try it in run-time code.

To begin using **EF DataSource** in run-time code without a **C1DataSource**, add a few lines to our project's main class to create a global client-side data cache. When we used **C1DataSource**, it was created for us behind the scenes. Now we create it explicitly using the following code:

Visual Basic	Copy Code
<pre>Imports C1.Data.Entities Class Application     Public Shared ClientCache As EntityClientCache     Private Sub Application_Startup(sender As Object, e As System.Windows.StartupEventArgs) Handles Me.Startup         ClientCache = EntityClientCache.GetDefault(GetType(NORTHWNDEntities))     End Sub End Class</pre>	



C#	Copy Code
<pre> using C1.Data.Entities;  public partial class App : Application {     public static EntityClientCache ClientCache;     public static NORTHWNDEntitiesObjectContext;      protected override void OnStartup(StartupEventArgs e)     {         base.OnStartup(e);          ObjectContext = new NORTHWNDEntities();         ClientCache = new EntityClientCache(ObjectContext);     } } </pre>	

This code creates a single application-wide (static) **ObjectContext** and associates it with **EntityClientCache**. As noted previously in [The Power of Client Data Cache](#)|document=WordDocuments\C1DataStudio-WPF.docx;topic=The Power of Client Data Cache topic, the ability to have a single context (and cache) for the entire application is a great simplification made possible by **EF DataSource**.

To perform server-side filtering in run-time code, follow these steps:

1. Create a new window, add a grid (dataGrid1 and set its AutoGenerateColumns property to True), a combo box (comboBox1), and a button (btnSaveChanges) to the form, and add the following code to the window class:

Visual Basic	Copy Code
<pre> Imports C1.Data.Entities Imports C1.Data Public Class DataSourcesInCode     Private _scope As EntityClientScope      Public Sub New()         ' This call is required by the designer.         InitializeComponent()     End Sub </pre>	

```

' Add any initialization after the InitializeComponent() call.
_scope = Application.ClientCache.CreateScope()
Dim viewCategories As ClientView(Of Category) = _scope.GetItems(Of
Category)()
    comboBox1.DisplayMemberPath = "CategoryName"
        comboBox1.ItemsSource = viewCategories
    BindGrid(viewCategories.First().CategoryID)
    End Sub
Private Sub BindGrid(categoryID As Integer)
    dataGrid1.ItemsSource =
        (From p In _scope.GetItems(Of Product)()).AsFiltered(
            Function(p As Product) p.CategoryID.Value =
categoryID)

        Select New With
        {
            p.ProductID,
            p.ProductName,
            p.CategoryID,
            p.Category.CategoryName,
            p.SupplierID,
            .Supplier = p.Supplier.CompanyName,
            p.UnitPrice,
            p.QuantityPerUnit,
            p.UnitsInStock,
            p.UnitsOnOrder
        }).AsDynamic()
    End Sub

Private Sub comboBox1_SelectionChanged(sender As System.Object,
e As System.Windows.Controls.SelectionChangedEventArgs) Handles
comboBox1.SelectionChanged
    If comboBox1.SelectedValue IsNot Nothing Then
        BindGrid(CType(comboBox1.SelectedValue,
Category).CategoryID)
    End If
End Sub

Private Sub btnSaveChanges_Click(sender As System.Object, e As
System.Windows.RoutedEventArgs) Handles btnSaveChanges.Click
    Application.ClientCache.SaveChanges()
End Sub
End Class

```

C#	Copy Code
<pre> using C1.Data.Entities; using C1.Data;  public partial class DataSourcesInCode : Window {     private EntityClientScope _scope;      public DataSourcesInCode()     {         InitializeComponent();          _scope = App.ClientCache.CreateScope();          ClientView&lt;Category&gt; viewCategories = _scope.GetItems&lt;Category&gt;();          comboBox1.DisplayMemberPath = "CategoryName";         comboBox1.ItemsSource = viewCategories;          BindGrid(viewCategories.First().CategoryID);     }      private void comboBox1_SelectionChanged(object sender, SelectionChangedEventArgs e)     {         if (comboBox1.SelectedValue != null) BindGrid(((Category)comboBox1.SelectedValue).CategoryID) ;     }      private void BindGrid(int categoryID)     {         dataGrid1.ItemsSource =             (from p in _scope.GetItems&lt;Product&gt;().AsFiltered(                 p =&gt; p.CategoryID == categoryID)             select new             { </pre>	

```

        p.ProductID,
        p.ProductName,
        p.CategoryID,
        CategoryName = p.Category.CategoryName,
        p.SupplierID,
        Supplier = p.Supplier.CompanyName,
        p.UnitPrice,
        p.QuantityPerUnit,
        p.UnitsInStock,
        p.UnitsOnOrder
    }).AsDynamic();
}

private void btnSaveChanges_Click(object sender,
RoutedEventArgs e)
{
    App.ClientCache.SaveChanges();
}
}

```

2. Save, build and run your application. You should see similar results to those you saw in the Server-Side Filtering|document=WordDocuments\C1DataStudio-WPF.docx;topic=Server-Side Filtering example, the only difference being that this time we've implemented it all in code.

The private field **\_scope** is the form's gateway to the global data cache. It is a pattern we recommend you follow in all the forms where you do not employ a **C1DataSource** component directly, as that does this for you automatically. It ensures that the entities the form needs stay in the cache while the form is alive, and that they are automatically released when all forms (scopes) holding them are released.

Creating a view showing all categories for the combo box is simple:

Visual Basic	Copy Code
<pre>Dim viewCategories As ClientView(Of Category) = _scope.GetItems(Of Category)()</pre>	
C#	Copy Code

```
ClientView<Category> viewCategories = _scope.GetItems<Category>();
```

To create the view to which the grid is bound that only provides those products associated with the chosen category in the combo box required one additional operator; `AsFiltered(<predicate>)`.

Visual Basic	Copy Code
<pre>From p In _scope.GetItems(Of Product)().AsFiltered(Function(p As Product) p.CategoryID.Value = categoryID)</pre>	
C#	Copy Code
<pre>from p in _scope.GetItems&lt;Product&gt;().AsFiltered(p =&gt; p.CategoryID == categoryID).</pre>	

Note that when this query is executed, the result does not necessarily require a round trip to the server to retrieve the products requested. The cache is examined first to see if it already contains the requested data, either because the required data has already been requested once before within this form or from another form in the application. Or, possibly a completely separate query run elsewhere in the application had requested that all products be returned, so the cache would already have all the product data. Again, this is a fundamental strength of **EF DataSource**. By providing your application with a global cache of data, its performance is continually improved throughout its lifetime.

Here we chose to create a new view, and bind the grid to it, every time the user selects a new category in the combo box (see the combo box's **SelectedValueChanged** event). However, we could have avoided the need to create new views all the time and instead created one single view using a special **BindFilterKey**, which we'll learn more about in the [Simplifying MVVM](#) [document=WordDocuments\C1DataStudio-WPF.docx;topic=Simplifying MVVM topic](#).

So, in summary, we replicated in code what we did on the design surface with **C1DataSource** in [Server-Side Filtering](#) [document=WordDocuments\C1DataStudio-WPF.docx;topic=Server-Side Filtering](#). We have even thrown in a little extra; we customized the fields shown in the grid columns as we did in [Customizing View](#) [document=WordDocuments\C1DataStudio-WPF.docx;topic=Customizing View](#) by adding a **Select** to our LiveLinq statement:

Visual Basic	Copy Code
--------------	-----------

```

Select New With
{
    p.ProductID,
    p.ProductName,
    p.CategoryID,
    p.Category.CategoryName,
    p.SupplierID,
    .Supplier = p.Supplier.CompanyName,
    p.UnitPrice,
    p.QuantityPerUnit,
    p.UnitsInStock,
    p.UnitsOnOrder
;

```

C#

Copy Code

```

select new
{
    p.ProductID,
    p.ProductName,
    p.CategoryID,
    CategoryName = p.Category.CategoryName,
    p.SupplierID,
    Supplier = p.Supplier.CompanyName,
    p.UnitPrice,
    p.QuantityPerUnit,
    p.UnitsInStock,
    p.UnitsOnOrder
};

```

Had we just wanted the raw product data returned from the table without any special formatting, we could have simply said;

```
select p;
```

# Live Views

[DataSource for Entity Framework in WPF](#) > Live Views

Client views are live; they are kept automatically up-to-date with changing data. But live views functionality is more general, live views (objects of class `C1.LiveLinq.LiveViews.View`|keyword=`View`(Of T) class from which `ClientView` is derived) can be defined on data of any kind, including but not limited to entities, and they support more LINQ query operators, such as grouping, sorting, joins and more (see [LiveLinq](#)).

All such operations (of which the most popular are grouping and sorting) can be applied to client views (because **ClientView** is derived from, For example:

LINQ	Copy Code
<pre>View productsByCategory = products     .OrderBy(p =&gt; p.ProductName).GroupBy(p =&gt; p.CategoryID); or in LINQ query syntax: View productsByCategory =     from p in products     orderby p.ProductName     group p by p.CategoryID into g     select new { Category = g.Key, Products = g };</pre>	

The resulting views are live, but they are not client views. This is not a defect of some sort; it is simply because such views don't need

`ClientView`|document=`XMLDocuments\Reference`;topic=`ClientView(T)` Class functionality.

Sorting and grouping can be performed entirely on the client without resorting to the server.

However, developers must be aware of this fact to avoid confusion. Once you applied a LINQ query operation to your view, it becomes a **View**, not a **ClientView** (however, it remains live, automatically reflects all changes you make to the data on the client). So, for example, if you need server-side filtering, use the `AsFilter`|keyword=`AsFiltered` method method, not the LINQ `Where` method. Use the LINQ `Where` method when you want filtering on the client.

## Simplifying MVVM

[DataSource for Entity Framework in WPF](#) > Simplifying MVVM

The Model-View-View-Model (**MVVM**) pattern is gaining in popularity as developers realize the benefits it gives their applications, making them easier to maintain and test and, particularly in the case of WPF and Silverlight applications, allowing a much clearer division of labor between the designer of the UI and the creator of the code that makes it work. However, without effective tools to aid program development based on **MVVM** patterns, programmers can

actually find themselves working harder because of the need to implement an extra layer (the View Model) and ensure that data is properly synchronized between that and the Model. This extra burden isn't onerous when you have a relatively small application with just a few simple collections (and best practice with MVVM is to use **ObservableCollection** as the datasource), but the bigger it becomes and the more collections it spawns, the worse it becomes. **Entity Framework DataSource (EF DataSource)** can ease the burden.

**EF DataSource** lets you use live views as your view model. Just create live views over your model collections and use them as your view model. Live views are synchronized automatically with their sources (model), so you don't need *any* synchronization code - it's all automatic. And live views are much easier to create than to write your own view model classes using **ObservableCollection** and filling those collections manually in code. You have all the power of LINQ at your disposal to reshape model data into live views. So, not only does synchronization code disappear, but the code creating view models is dramatically simplified.

To demonstrate how easy it is to follow the **MVVM** pattern using live views, let's create a form combining all features from two previous examples: the **Category-Products** master-detail from [Working with Data Sources in Code](#)`[document=WordDocuments\C1DataStudio-WPF.docx;topic=Working with Data Sources in Code` and the reshaping/filtering/ordering of data from [Live Views](#)`[document=WordDocuments\C1DataStudio-WPF.docx;topic=Live Views`. It will be a **Category-Products** view, showing non-discontinued products whose unit price is at least **30**, ordered by unit price, and displaying a customized set of product fields in a master-detail form where the user can select a category to show the products of that category. We'll follow the **MVVM** pattern. The form (view), called **CategoryProductsView**, only hosts GUI controls (a combo box and a grid) and does not have any code except what is used to set the data source to the view model.

All logic is in the view model class, separate from the GUI. More exactly, there are three classes: **CategoryViewModel**, **ProductViewModel**, and **CategoryProductsViewModel**. The first two are simple classes defining properties with no additional code:

Visual Basic	Copy Code
<pre>Public Class CategoryProductsViewModel     Private _scope As EntityClientScope     Private _categories As ICollectionView     Public Property Categories As ICollectionView         Get             Return _categories         End Get         Private Set(value As ICollectionView)             _categories = value         End Set     End Property     Private _products As ICollectionView</pre>	



```

Public Property Products As ICollectionView
    Get
        Return _products
    End Get
    Private Set(value As ICollectionView)
        _products = value
    End Set
End Property
Public Sub New()
    _scope = Application.ClientCache.CreateScope()
    Categories =
        From c In _scope.GetItems(Of Category)()
        Select New CategoryViewModel With
        {
            .CategoryID = c.CategoryID,
            .CategoryName = c.CategoryName
        }
    Products =
        From p In _scope.GetItems(Of
Product)().AsFilteredBound(Function(p) p.CategoryID.Value)
        .BindFilterKey(Categories,
"CurrentItem.CategoryID").Include("Supplier")
        Select New ProductViewModel With
        {
            .ProductID = p.ProductID,
            .ProductName = p.ProductName,
            .CategoryID = p.CategoryID,
            .CategoryName = p.Category.CategoryName,
            .SupplierID = p.SupplierID,
            .SupplierName = p.Supplier.CompanyName,
            .UnitPrice = p.UnitPrice,
            .QuantityPerUnit = p.QuantityPerUnit,
            .UnitsInStock = p.UnitsInStock,
            .UnitsOnOrder = p.UnitsOnOrder
        }
    End Sub
End Class

```

C#	Copy Code

```

public class CategoryViewModel
{
    public virtual int CategoryID { get; set; }
    public virtual string CategoryName { get; set; }
}

public class ProductViewModel
{
    public virtual int ProductID { get; set; }
    public virtual string ProductName { get; set; }
    public virtual int? CategoryID { get; set; }
    public virtual string CategoryName { get; set; }
    public virtual int? SupplierID { get; set; }
    public virtual string SupplierName { get; set; }
    public virtual decimal? UnitPrice { get; set; }
    public virtual string QuantityPerUnit { get; set; }
    public virtual short? UnitsInStock { get; set; }
    public virtual short? UnitsOnOrder { get; set; }
}

public class CategoryProductsViewModel
{
    private C1.Data.Entities.EntityClientScope _scope;

    public System.ComponentModel.ICollectionView Categories { get; private set; }
    public System.ComponentModel.ICollectionView Products { get; private set; }
}

public CategoryProductsViewModel()
{
    if (App.ClientCache == null)
        return;

    _scope = App.ClientCache.CreateScope();

    Categories =
        from c in _scope.GetItems<Category>()
        select new CategoryViewModel()
        {
            CategoryID = c.CategoryID,
            CategoryName = c.CategoryName

```

```

        };

        Products =
            from p in _scope.GetItems<Product>().AsFilteredBound(p =>
p.CategoryID)
                .BindFilterKey(Categories,
"CurrentItem.CategoryID").Include("Supplier")
            select new ProductViewModel()
            {
                ProductID = p.ProductID,
                ProductName = p.ProductName,
                CategoryID = p.CategoryID,
                CategoryName = p.Category.CategoryName,
                SupplierID = p.SupplierID,
                SupplierName = p.Supplier.CompanyName,
                UnitPrice = p.UnitPrice,
                QuantityPerUnit = p.QuantityPerUnit,
                UnitsInStock = p.UnitsInStock,
                UnitsOnOrder = p.UnitsOnOrder
            };
    }
}

```

Basically, it contains just two LiveLinq statements, nothing more.

Similar to what we saw in [Working with Data Sources in Code](#), using `AsFilteredBound()` gives us server-side filtering. In [Working with Data Sources in Code](#), we connected the filter key (which is the **Product.CategoryID** property) to **CategoryID** selected by the user using a combo box event. Here we can't do this because we must keep our code independent of GUI. So we use a **BindFilterKey** method to bind the filter key to the **Category.CategoryID** property of the item currently selected in the **Categories** collection.

The **Include("Supplier")** operator is not strictly necessary; it is used here for performance optimization. Without it, **Entity Framework** will fetch **Supplier** objects lazily, one-by-one, every time the user accesses an element of the **Products** collection whose **Supplier** was not yet fetched. This can cause delays, and it's generally much less efficient to fetch data in single rows than in batches, so we opted out of lazy loading here using **Include("Supplier")**, which tells **Entity Framework** to fetch supplier information in the same query with products.

Finally, we need to specify data binding for the GUI controls in the form (view) `CategoryProductsView.xaml`. Following the MVVM pattern, we do it in XAML (not in code), as shown below:

XAML	Copy Code
<pre> &lt;Grid&gt;     &lt;Grid.DataContext&gt;         &lt;local:CategoryProductsViewModel /&gt;     &lt;/Grid.DataContext&gt;     &lt;Grid.RowDefinitions&gt;         &lt;RowDefinition Height="Auto"/&gt;         &lt;RowDefinition /&gt;     &lt;/Grid.RowDefinitions&gt;     &lt;ComboBox HorizontalAlignment="Left" Margin="5" Name="comboBox1"         Width="221" ItemsSource="{Binding Categories}"         DisplayMemberPath="CategoryName" /&gt;     &lt;DataGrid Grid.Row="1" AutoGenerateColumns="True" Name="dataGrid1"         ItemsSource="{Binding Products}"/&gt; &lt;/Grid&gt; </pre>	

Defining DataContext like this,

XAML	Copy Code
<pre> &lt;Grid.DataContext&gt;     &lt;local:CategoryProductsViewModel /&gt; &lt;/Grid.DataContext&gt; </pre>	

we made our form create a **CategoryProductsViewModel** object as its data source, and we bound the combo box and the grid to it using {Binding Categories} and {Binding Products}.

## Using Entity Framework DataSource in MVVM with other MVVM Frameworks

[DataSource for Entity Framework in WPF](#) > Using Entity Framework DataSource in MVVM with other MVVM Frameworks

**Entity Framework DataSource (EF DataSource)** can be used to build Model-View-View-Model (**MVVM**) applications with any other MVVM frameworks.

**Entity Framework DataSource** offers several features to make your MVVM development easier:

- **Simplifies MVVM programming**

Given that **EF DataSource** can be used to simplify MVVM programming, as seen in the Simplifying MVVM|document=WordDocuments\C1DataStudio-WPF.docx;topic=Simplifying MVVM topic, it's clearly a tool that can be used for MVVM.

- **Helps create view model classes and alleviate code bloating**

There are a plethora of tools and frameworks that developers can use to aid them in their work with MVVM, but very few that can help them create view model classes. The majority are designed to help with tasks such as passing commands and messages between the view and view model. Creating view model classes and then synchronizing them with model data is left almost entirely to manual coding. This is the primary cause of code bloating in most MVVM applications and precisely the one that **EF DataSource** is designed to alleviate in a way that is entirely compatible with other frameworks.

- **Allows you to use any framework and live views**

You can use any framework you like to assist your MVVM application development and simply call on **EF DataSource** to provide live views|document=WordDocuments\C1DataStudio-WPF.docx;topic=Live Views to help you create view model classes.

To demonstrate these important points, we provide a sample based on the code from the well-known article by Josh Smith, one of the authors of MVVM, "WPF Apps With The Model-View-ViewModel Design Pattern" (<http://msdn.microsoft.com/en-us/magazine/dd419663.aspx>).

The sample can be found in the **Documents\ComponentOne Samples\WPF** folder.

Essentially all the files in our modified sample are the same as the originals (bar a few cosmetic changes) except one (ViewModels\OrdersViewModel.cs).

In this file, we build the view model class the **EF DataSource** way, using live views. You can see how many re-shaping functions are applied to model data to construct a view model, all done exclusively through LINQ. This made it easy and required little code. The best part is that it synchronizes automatically with model data when data in either of the two layers is changed - no synchronization code was necessary.

The fact that we only changed the way that the view model classes themselves were created (they are still derived from the original base class 'ViewModelBase') and made no other changes to the framework code that Josh Smith had employed in his original sample should serve as an example that **EF DataSource** is entirely compatible with other frameworks. You can continue to use your preferred frameworks when working with MVVM, but now you have an additional tool to make your MVVM development even easier.

## Samples

[DataSource for Entity Framework in WPF](#) > Samples

The following topics provide information about the samples.

# Getting Started Samples

[DataSource for Entity Framework in WPF](#) > [Samples](#) > Getting Started Samples

These are the sample projects described in the **Getting Started** part of the LiveLinq documentation. You can create those projects yourself following the complete step-by-step instructions in **Getting Started with LiveLinq**. The pre-created projects are included in the LiveLinq distribution package for your convenience.

## How To Samples

[DataSource for Entity Framework in WPF](#) > [Samples](#) > How To Samples

The following sections describe LiveLinq's HowTo samples.

## Indexing Samples

[DataSource for Entity Framework in WPF](#) > [Samples](#) > [How To Samples](#) > Indexing Samples

HowTo Indexing samples demonstrate basic use of the first of the two areas of LiveLinq functionality: using indexes to make LINQ queries faster. They don't use live views, except for LiveLinqToXml, where live views are necessary to define base data collections.

Every sample uses two queries: one the most basic and the other a little more complex, with a join. The queries are identical in functionality in all three samples, differing only in the nature of the underlying data: user object collections (LiveLinqToObjects), ADO.NET DataSet (LiveLinqDataSet) or XML (LiveLinqToXml).

Every sample shows two alternative ways of creating indexes: explicitly, by adding to the **Indexes** collection, or implicitly, by using hints in queries.

### LiveLinqToObjects

This sample shows how to use the **IndexedCollection** class to create and query data collections using indexes to speed up query performance.

#### LiveLinqToDataSet

This sample shows how to use LiveLinq to query data residing in an ADO.NET DataSet, how to create indexes over that data that make querying faster than regular LINQ to DataSet queries. Both strongly typed and untyped datasets are demonstrated in the sample.

#### LiveLinqToXml

This sample shows how to use LiveLinq to query XML data. It creates indexes on 'customers' and 'orders' live views defined directly over XML data. This easy XML indexing allows to dramatically speed up LINQ queries over XML data, often make them hundreds of times faster than regular LINQ to XML, see Query Performance sample application (LiveLinqQueries) for performance metrics.

# Live View Samples

[DataSource for Entity Framework in WPF](#) > [Samples](#) > [How To Samples](#) > Live View Samples

HowTo LiveViews samples show the second and main part of LiveLinq functionality: LINQ queries as live views, automatically reflecting changes in the data on which they are based, so they can be used to build applications declaratively, minimizing procedural, manual coding.

Every sample uses two views over the same data: one the most basic (filtered view) and the other a little more complex, with a join. The functionality is identical in all three samples, differing only in the nature of the underlying data: user object collections (LiveLinqToObjects), ADO.NET DataSet (LiveLinqDataSet) or XML (LiveLinqToXml).

To see live views in action, you can try, for example:

- Change the ShipCity value from "London" to "Colchester" or vice versa in any of the two grids and leave the current row (to commit the change). Observe that the value in the other grid changes accordingly.
- Change the ShipCity value to any string other than "London" and "Colchester" in any of the two grids. Observe that the row disappears from both grids.

See also the Live views sample application (LiveLinqIssueTracker) for more advanced functionality.

## LiveViewsObjects

This sample demonstrates basic live view functionality for views based on user-defined object collections (LiveLinq to Objects).

## LiveViewsDataSet

This sample demonstrates basic live view functionality for views based on ADO.NET DataSet (LiveLinq to DataSet). This sample uses a typed data set, but untyped data sets can be used as well, as shown in the LiveLinqToDataSet sample.

## LiveViewsXml

This sample demonstrates basic live view functionality for views based on XML (LiveLinq to XML).

# Programming Guide

[DataSource for Entity Framework in WPF](#) > Programming Guide

The following sections provide information concerning **Entity Framework DataSource** programming. For additional information about LiveLinq-specific features see the LiveLinq Programming Guide|document=WordDocuments\LiveLinq-ProgrammingGuide.doc;topic=Programming Guide.

# Query Operators Supported in Live Views

[DataSource for Entity Framework in WPF](#) > [Programming Guide](#) > Query Operators Supported in Live Views

LiveLinq supports all LINQ query operators in its queries. Not all operators have LiveLinq-specific implementation benefiting from indexes and other query execution optimization techniques, but such operations simply use the standard LINQ to Objects (or LINQ to XML) implementations, so it is transparent to the user.

However, not all query operations can be used in live views. This is because not all query operations have incremental view maintenance algorithms, as some would require re-populating from scratch (requering) every time they are maintained. If your query contains such operations, you won't be able to use that query to create a live view. An attempt to do that will cause a compilation error.

Here is the list of query operators allowed in live views:

Operator	Notes
<b>Select</b>	Overload with <b>selector</b> depending on the index is not allowed.
<b>Where</b>	Overload with <b>predicate</b> depending on the index is not allowed.
<b>Join</b>	Overload with <b>comparer</b> is not allowed.
<b>GroupJoin</b>	Overload with <b>comparer</b> is not allowed.
<b>OrderBy</b>	Overload with <b>comparer</b> is not allowed.
<b>OrderByDescending</b>	Overload with <b>comparer</b> is not allowed.
<b>GroupBy</b>	Overload with <b>comparer</b> is not allowed.
<b>SelectMany</b>	Overloads with <b>selector</b> and <b>collectionSelector</b> depending on the index are not allowed.
<b>Union</b>	
<b>Concat</b>	
<b>Aggregate</b>	Use <b>LiveAggregate</b> method if you want to optimize performance with incremental view maintenance.
<b>Count</b>	Use <b>LiveCount</b> method if you want to optimize performance with incremental view maintenance.
<b>Min/Max</b>	Use <b>LiveMin/LiveMax</b> methods if you want to optimize performance



	with incremental view maintenance.
<b>Sum</b>	Use <b>LiveSum</b> method if you want to optimize performance with incremental view maintenance.
<b>Average</b>	Use <b>LiveAverage</b> method if you want to optimize performance with incremental view maintenance.

## Query Expressions Supported in Live Views

[DataSource for Entity Framework in WPF](#) > [Programming Guide](#) > Query Expressions Supported in Live Views

Apart from the limitations on query operators, a query must satisfy an additional condition in order to be used as a live view. Fortunately, this condition is satisfied in most cases, so you should care about it only if your query contains some special expressions, which is relatively rare (except that all classes used in LiveLinq to Objects must satisfy the property notification condition, but that was already mentioned in Using the built-in collection class `IndexedCollection<T>` (LiveLinq to Objects)|tag=Using\_the\_built\_in\_collection\_class\_IndexedCollectionT\_LiveLinq\_to\_Objects). Unfortunately, this condition is not verified by LiveLinq automatically, so it is your responsibility to make sure it is satisfied. If this condition is not satisfied, your view will not react to changes in its base data. We give two descriptions of this condition: one short, for basic understanding, and another detailed, for advanced usage.

## View Maintenance Mode

[DataSource for Entity Framework in WPF](#) > [Programming Guide](#) > View Maintenance Mode

Live views can be used in different kinds of applications. They can be used in GUI, interactive applications and in non-GUI, batch processing-style applications (see How to use live views in non-GUI code). Live views are optimized for both modes, GUI (interactive) and non-GUI (batch). They distinguish between these modes and operate accordingly. In GUI, live views react to a change immediately when the change happens. They do it fast, using incremental algorithms without re-populating themselves, (see Maintaining the view: Incremental View Maintenance), but still it is not always suitable for batch, non-interactive processing. In GUI, interactive programs, with data binding, immediate reaction to change is what is usually needed because the user needs to see the change on the screen. In batch processing, on the other hand, a view may be accessed long after the change occurred or even not at all. So updating that view before it is actually accessed by the program may be an unnecessary drain on resources. By default, live views distinguish between these two modes automatically, but the programmer has an option to control that using the `MaintenanceMode|keyword=MaintenanceMode` Property property. In **Immediate** mode, which is the default for GUI, the view is maintained immediately on every change. In **Deferred** mode, which is the default in a non-GUI case (when there are no

listeners attached to the view), the view is maintained on demand. It is not kept in sync with base data until it is needed or until there is a request for data from that view. When such a request arrives, the view sees that it is in a "dirty" unsynchronized state, so it automatically updates (maintains) itself to come in sync with the changed base data.

## Updatable Views

[DataSource for Entity Framework in WPF](#) > [Programming Guide](#) > Updatable Views

Live views are bi-directionally updatable. The first direction is from base data to the view: the base (source) data can be modified and the view will be automatically updated and then synchronized with the changed base data. That is what makes the views *live*. There is also the second, opposite direction: data can be changed directly in the view. This can be done programmatically (using the `ViewRow|keyword=ViewRow Class` class), and it can also be done via data binding. Updating data directly in the view is especially common if you bind a GUI control, such as a grid, to a view. An example of updating data in a view via data binding can be seen in one of the first samples, **Basic LiveLinq**, among others.

We call a view *updatable* if data can be modified directly in the view. This term only concerns the second direction of updating data. The first direction mentioned above, updating base data, is always possible without restrictions for any live views. So, when we say that a view is not updatable (is *read-only*), it does not mean that the data in the view cannot change. It can change to reflect changes in base data because every view in LiveLinq is live. It merely means that data cannot be changed directly in the view, you need to change base data if you want to modify it.

Not all live views are updatable, and even if the view as a whole is updatable, some properties of it may be read-only. These are properties that don't have direct correspondence to a property in the view's source data. Updating a view ultimately means updating that view's base data, that is, one of the view's sources, because the view itself is only virtual, does not have data of its own, shows (filtered and shaped) data of its sources. This is why a view field (property) can be updated only if it can be put into a direct correspondence with a property in the source. For example, in the following view **City** and **CompanyName** are updatable because they directly correspond to the **City** and **CompanyName** fields of the **customers** source. But **FullName** is not updatable, because there is no single field (property) in the source to which it corresponds, it is a combination of two source properties:

C#	Copy Code
<pre>for c in customers where c.City == "London" select new {     c.City,</pre>	

```

        c.CompanyName,
        FullName = c.FirstName + " " + c.LastName
    }

```

Any update of a view, which can be adding, deleting or modifying a view item, is performed by performing the corresponding operation, adding, deleting or modifying a single item in one of the view's sources. Although the effect is usually obvious and it usually does what is intended, it is important to know this simple rule, to understand it formally and exactly, because the result can sometimes be unexpected if you don't take this rule into account. For example, if you modify an item (or add a new one) in the above view so its **City** value is not "London", that item will disappear from the view.

The above rule stating that updating a view item is equivalent to updating an item of one of the view's sources, implies that only one of the view's sources can be updatable. A **Join** view has two sources, so we must determine which of them is updatable. By default, a join view is read-only. If you need to make one of its two parts updatable, use the extension method `AsUpdatable()`|keyword=`AsUpdatable(T)` Method, for example:

C#	Copy Code
<pre> for o in orders.AsUpdatable() join c in customers on o.CustomerID equals c.CustomerID select new {     o.OrderDate,     o.Amount,     c.CompanyName } </pre>	

Here order data (date and amount) will be updatable and customer data (**CustomerName**) read-only.

To find out if a view is updatable, use the property `View.IsReadOnly`|keyword=`IsReadOnly` Property. The list of all properties of a view accessible to data binding and programmatic access, with full information including their updatability, can be obtained using the property `ViewRowCollection.Properties`|keyword=`Properties` Property.

Updating data directly in a view in code is done through a special class `ViewRow`|keyword=`ViewRow` Class, each view row representing a view item for programmatic access and data binding purposes. For details of using this class in code see reference documentation for `ViewRowCollection`|keyword=`ViewRowCollection` Class and `ViewRow`|keyword=`ViewRow` Class.

# Live View Performance

[DataSource for Entity Framework in WPF](#) > [Programming Guide](#) > Live View Performance

First performance consideration with live views is that, naturally, live view functionality has a price. Maintaining the view in sync with base data is fast and optimized, but still it consumes some resources when base data changes and the view is maintained. And it consumes some additional resources when it is first populated as well. That additional cost is not high, but it exists, manifesting itself mostly in additional memory consumption: when a live view is populated, it creates some internal data in memory to help it maintain itself faster on base data changes.

This additional memory consumption is moderate—roughly equivalent to the amount of memory needed for the resulting list itself. So, although additional resources needed for live view functionality are light to moderate, the obvious suggestion is to use live views only where you are really interested in keeping the result of a query live, or, in other words, where you need that result more than once and the base data is changing so that the result is changing too.

Other than this, there are two different aspects to live view performance.

## LiveLinq Query Performance: Logical Optimization

[DataSource for Entity Framework in WPF](#) > [Programming Guide](#) > LiveLinq Query Performance: Logical Optimization

Standard LINQ to Objects does not perform any logical optimization, it executes queries exactly as they are written. And, of course, standard LINQ to Object does not use indexes for optimization. In comparison, LiveLinq performs physical optimization (using indexes, if they are present) and logical optimization (re-writing the query in a more efficient form before processing it).

However, LiveLinq does not contain a full-scale query optimizer like one in a relational database such as SQL Server or Oracle, so it can still matter how a query is written. But it is largely limited to join ordering. LiveLinq does not try to re-order your joins. Joins are executed always in the same order as specified in the query. So, you should avoid writing queries with obviously inefficient join order. But it must be noted that it is a relatively rare problem (and the same consideration, of course, applies to the standard LINQ to Objects anyway). It can be an issue in a query like

LINQ	Copy Code
<pre>from a in A join b in B on a.k equals b.k where b.p == 1</pre>	(1)

if there are, for example, only 10 elements in **B** with **b.p == 1** and only 100 elements in **A** that satisfy **a.k == b.k** and **b.p == 1**, but the overall number of elements in **A** is many times higher, for example, 10000. Then the query above will require 10000 "cycles", whereas rewritten with the right join order,

LINQ	Copy Code
<pre>from b in B join a in A on b.k equals a.k where b.p == 1</pre>	(2)

this query will require only 100 cycles to complete. Note that the position of **where** is not important. The above query has the same performance as

LINQ	Copy Code
<pre>from b in B where b.p == 1 join a in A on b.k equals a.k</pre>	(3)

That's because of LiveLinq query optimization: LiveLinq re-writes (2) internally to (3) when it executes it, because it is preferable to check conditions before performing other operations and to exclude elements that don't contribute to the result. This is one of the logical optimizations performed by LiveLinq internally.

## LiveLinq Query Performance: Tuning Indexing Performance

[DataSource for Entity Framework in WPF](#) > [Programming Guide](#) > LiveLinq Query Performance: Tuning Indexing Performance

Query speedup achieved on any particular query by LiveLinq using indexes for optimization depends on how that query is written. This dependence is usually not dramatic. LiveLinq does a fair job of recognizing opportunities to use indexes for optimization regardless of the way the query is written. For example, it will still execute a query efficiently using indexes even if the condition consists of multiple terms connected with logical operators.

# Live Views How To: Create Views Based on Other Views and Create Indexes on Views

[DataSource for Entity Framework in WPF](#) > [Programming Guide](#) > Live Views How To: Create Views Based on Other Views and Create Indexes on Views

Multiple queries with parameters can often be replaced with a single “big” live view. That can help making code clear and simple and can often make it faster. This technique is based on two features of live views:

- A view can be based on other views in the same way as it can be based on base data.
- Indexes can be created for a view in the same way as indexes are created for base data.

So, instead of issuing many queries by varying a parameter value, you can create a single live view, create an index for it, and, when you need the list of items corresponding to a particular value of the parameter, you simply get those items from the index for that particular value.

It is illustrated in the LiveLinqIssueTracker demo|tag=Live views sample application (LiveLinqIssueTracker);document=WordDocuments\LiveLinq-Samples.doc:

The **Assigned Issues** form uses multiple views, a separate view for every employee: views depending on a parameter **employeeID** (we are giving LiveLinq to DataSet versions of the views here, Objects and XML versions are similar):

LINQ	Copy Code
<pre>from i in _dataSet.Issues.AsLive() join p in _dataSet.Products.AsLive()     on i.ProductID equals p.ProductID join f in _dataSet.Features.AsLive()     on new { i.ProductID, i.FeatureID }         equals new { f.ProductID, f.FeatureID } join e in _dataSet.Employees.AsLive()     on i.AssignedTo equals e.EmployeeID where i.AssignedTo == employeeID select new Issue {     IssueID = i.IssueID,     ProductName = p.ProductName,     FeatureName = f.FeatureName,     Description = i.Description,     AssignedTo = e.FullName };</pre>	

The demo application also contains an alternative implementation of the same functionality, in the form **Assigned Issues 2**. That form uses a single view containing data for all employees, instead of multiple views depending on a parameter. This single view does not have parameters:

LINQ	Copy Code
<pre> _bigView =     from i in _dataSet.Issues.AsLive()     join p in _dataSet.Products.AsLive()         on i.ProductID equals p.ProductID     join f in _dataSet.Features.AsLive()         on new { i.ProductID, i.FeatureID }             equals new { f.ProductID, f.FeatureID }     join e in _dataSet.Employees.AsLive()         on i.AssignedTo equals e.EmployeeID     select new Issue     {         IssueID = i.IssueID,         ProductName = p.ProductName,         FeatureName = f.FeatureName,         Description = i.Description,         AssignedToID = e.EmployeeID,         AssignedToName = e.FullName     }; </pre>	

That view is indexed by the employee id field:

LINQ	Copy Code
<pre> _bigView.Indexes.Add(x =&gt; x.AssignedToID); </pre>	

so we can retrieve the items for a particular employee id at any time very fast.

Moreover, we can create a live view of that data given an employee id value (which is **comboAssignedTo.SelectedIndex** in this demo), simply by creating a view over the big view:

LINQ	Copy Code
<pre> from i in _bigView where i.AssignedToID == comboAssignedTo.SelectedIndex select i; </pre>	

# Live Views How To: Use Live Views to Create Non-GUI Applications as a Set of Declarative Rules Instead of Prodedural Code

[DataSource for Entity Framework in WPF](#) > [Programming Guide](#) > Live Views How To: Use Live Views to Create Non-GUI Applications as a Set of Declarative Rules Instead of Prodedural Code

In addition to making GUI applications declarative, LiveLinq also enables a programming style (which we tentatively call **view-oriented programming**) that makes non-GUI, batch processing declarative too. See [How to use live views in non-GUI code](#) for the introductory description of this technique.

The LiveLinqIssueTracker demo application|tag=Live views sample application (LiveLinqIssueTracker);document=WordDocuments\LiveLinq-Samples.doc contains a **Batch Processing** form where this technique is demonstrated by implementing two actions:

- (1) Assign as many unassigned issues to employees as possible. Looking at the feature to which a particular issue belongs, find an employee assigned to that feature, and, if that employee does not have other issues for that feature, assign that issue to that employee.
- (2) Collect information on all open (not fixed) issues for a given employee (perhaps for the purpose of emailing that list to that employee).

We perform action (1) by defining the following view (we are giving LiveLinq to DataSet versions of the views here, Objects and XML versions are similar):

C#	Copy Code
<pre>_issuesToAssign =     from i in issues     where i.AssignedTo == 0     join a in _dataSet.Assignments.AsLive()         on new { i.ProductID, i.FeatureID }             equals new { a.ProductID, a.FeatureID }     join i1 in issues         on new { i.ProductID, i.FeatureID, a.EmployeeID }             equals new { i1.ProductID, i1.FeatureID, EmployeeID = i1.AssignedTo }     into g     where g.Count() == 0     join e in _dataSet.Employees.AsLive()</pre>	



```

        on a.EmployeeID equals e.EmployeeID
select new IssueAssignment
{
    IssueID = i.IssueID,
    EmployeeID = a.EmployeeID,
    EmployeeName = e.FullName
};

```

and performing the operation with simple code based on that view:

C#	Copy Code
<pre> foreach (IssueAssignment ia in _issuesToAssign)     _dataSet.Issues.FindByIssueID(ia.IssueID).AssignedTo = ia.EmployeeID; </pre>	

Action (2) is performed by defining the following view:

C#	Copy Code
<pre> from i in _dataSet.Issues.AsLive() join p in _dataSet.Products.AsLive()     on i.ProductID equals p.ProductID join f in _dataSet.Features.AsLive()     on new { i.ProductID, i.FeatureID }         equals new { f.ProductID, f.FeatureID } join em in _dataSet.Employees.AsLive()     on i.AssignedTo equals em.EmployeeID where i.AssignedTo == employeeID &amp;&amp; !i.Fixed select i.IssueID; </pre>	

If we need to perform the operations (1) and (2) just once, then there is no point in using live views. But suppose we are writing a program that needs to perform those actions (1) and (2) many times as steps of the overall algorithm it is executing. This is a common case, especially in server-side programming.

Without live views, we would either need to requery each time, which is very costly, or we would need to create and maintain some collections throughout our processing algorithm. Those collections would need to be maintained by manual code, often complicated, and written by different programmers and often containing bugs. Different people write different functions (actions (1) and (2) are, of course, just two small examples of such functions), they need to know what others are doing if they want to keep it consistent. All this is hard to maintain. When a new

action or function is added a year after that, the logic that connects everything is by that time forgotten, the new action breaks that logic, and so the vicious cycle of too complicated programming goes on.

Compare this with how it can be done with live views:

Actions (1) and (2) are not actually procedures, they are declarative rules. Rule (1) states that this view contains the assignments to be made. Whatever changes are made to our data, this view will always contain the needed assignments, regardless of anything we add a year from now or at any other time. The logic is guaranteed to be correct because it is a declarative rule, not procedural logic.

And that rule (1) works fast. We can perform that action a thousand times over data that is always changing, and every time it will recompute only the required amount of data, only the amount that is affected by changes. It will perform only the needed **incremental** recomputations.

Same with rule (2): it is a declarative rule, and it executes fast without repeating calculations, recalculating only those parts that are affected by changes.

If a substantial part of our algorithm is programmed in this **view-oriented** way, as a set of rules like (1) and (2), then they all work together, their consistency is automatically maintained.

Ideally, we can represent our entire algorithm with views/rules like that. If not, a part of it can be implemented like that and the rest can be done with the usual procedural code.

## LiveLinq

Make LINQ faster and get live views with **LiveLinq**. This unique class library augments the functionality of LINQ using indexing and other optimizations to speed up LINQ queries up to 100 times faster or more. And with live view, your LINQ query result is kept up-to-date without re-populating it every time its base data changes.

## LiveLinq Overview

[LiveLinq](#) > LiveLinq Overview

LiveLinq is a class library that augments the functionality of LINQ in two related directions:

- **It makes LINQ faster**

LiveLinq uses indexing and other optimizations to speed up LINQ queries in memory. Speed gains can be hundreds and even thousands of times on queries with high selectivity conditions. Typical/average gains are lower but still significant; a 10-50 times speedup can be considered typical.

- **It adds support for live views to LINQ**

A *live view* is a LINQ query result that is kept constantly up-to-date without re-populating it every time its base data changes. This makes LiveLinq extremely useful in common data-binding scenarios where objects are edited and may be filtered in or out of views, have their associated subtotals updated, and so on. In old jargon, one could say that LINQ queries correspond to snapshots, while LiveLinq views correspond to dynasets. Since live views automatically react to changes, they greatly widen the sphere of declarative programming, not only in data binding and GUI but in many other programming scenarios as well.

## LiveLinq in Silverlight

[LiveLinq](#) > LiveLinq in Silverlight

LiveLinq can be used in Silverlight (starting with Silverlight 4) as well as in .NET Framework. Use the assembly

**C1.Silverlight.LiveLinq.dll** instead of **C1.LiveLinq.dll**.

LiveLinq indexing features are not available in Silverlight, but live views are fully supported in Silverlight as well as in .NET Framework. The following namespaces (containing indexing features and ADO.NET support unavailable in Silverlight) are not present in the Silverlight version of LiveLinq:

C1.LiveLinq.AdoNet

C1.LiveLinq.Indexing

C1.LiveLinq.Indexing.Search

Two sample projects are included in Samples\LiveLinq\HowTo\LiveViews showing how to use LiveLinq in Silverlight with collections (ObservableCollection in the sample) and with XML:

LiveViews-Silverlight-Objects

LiveViews-Silverlight-XML

These are only samples using LiveLinq in Silverlight with in-memory objects, without database. Many more Silverlight samples using RIA Services for database access can be found in other samples, tutorials, and documentation of ComponentOne Studio for Entity Framework.

## How to use LiveLinq

### How to Query Collections with LiveLinq

[LiveLinq](#) > How to use LiveLinq > How to Query Collections with LiveLinq

To use LiveLinq in Visual Studio, start by adding the LiveLinq assembly, C1.LiveLinq.dll, to your project's References. Then add the 'using' directives to the source file that will enable you to use LiveLinq in code:

C#	Copy Code
<pre>using C1.LiveLinq; using C1.LiveLinq.Indexing; using C1.LiveLinq.Collections;</pre>	

(Not all of them are always necessary, but let's put all of them there at once for simplicity)

In order to query a collection in LiveLinq, we need to wrap it in an interface

**IIndexedSource<T>** that will tell LiveLinq to take over. Otherwise, standard LINQ will be used. This wrapping is done with a call to the **AsIndexed** extension method, for example:

C#	Copy Code
<pre>from c in source.AsIndexed() where c.p == 1 select source;</pre>	

However, not every collection can be wrapped this way. For instance, if we try this with a **List<T>** source, we'll get a compilation error. To be usable in LiveLinq, a collection must support change notifications, it must notify LiveLinq when changes are made to its objects and when objects are added to or deleted from the collection. This requirement is satisfied for collections used for data binding, that is, implementing either **IBindingList** (WinForms data binding) or **INotifyCollectionChanged/INotifyPropertyChanged** (WPF data binding)

In particular, **AsIndexed()** is applicable to ADO.NET collections (**DataTable**, **DataRowView**) and to LINQ to XML collections.

## Using the Built-in Collection Class

### IndexedCollection<T>(LiveLinq to Objects)

[LiveLinq](#) > [How to use LiveLinq](#) > [How to Query Collections with LiveLinq](#) > Using the Built-in Collection Class `IndexedCollection(LiveLinq to Objects)`

Suppose first that we don't care what collection we use for our objects. We have a **Customer** class, something like the following:

C#	Copy Code
<pre>public class Customer {     public string Name { get; set; }     public string City { get; set; }</pre>	

```
}
```

We rely on LiveLinq to supply the collection class we need.

Then we need look no further than the **C1.LiveLinq.Collections.IndexedCollection** class that is supplied by LiveLinq and is specifically optimized for LiveLinq use:

C#	Copy Code
<pre>var customers = new IndexedCollection&lt;Customer&gt;(); customers.Add(cust1); customers.Add(cust2); ... var query =     from c in customers where c.City == "London" select c;</pre>	

Note that we can simply use **customers** instead of **customers.AsIndexed()**. It is because the **IndexedCollection<T>** class already implements the **IIndexedSource<T>** interface that LiveLinq needs, there is no need to use the **AsIndexed()** extension method to wrap it into that interface.

There is an important consideration to be kept in mind using your own classes such as **Customer** above for collection elements. The **Customer** class above is so basic that it does not support property notifications. If you set one of its properties in code, nobody will notice it, including any indexes and live views|tag=How to create a live view you may create over that collection. Therefore it is highly necessary to provide property change notifications in such classes. Property change notifications is a standard .NET feature recommended for a variety of reasons, LiveLinq just adds another reason to do that. You can support property change notifications in your class by implementing the **INotifyPropertyChanging** and **INotifyPropertyChanged** interfaces. Or you can use LiveLinq for that, by deriving your class from **IndexableObject** Class and calling **OnPropertyChanging/OnPropertyChanged** like this:

C#	Copy Code
<pre>public class Customer : IndexableObject {     private string _name;     public string Name     {         get {return _name} ;         set         {</pre>	

```

        OnPropertyChanged("Name");
        _name = value;
        OnPropertyChanged("Name");
    }
}
private string _city;
public string City
{
    get {return _city} ;
    set
    {
        OnPropertyChanged("City");
        _city = value;
        OnPropertyChanged("City");
    }
}
}

```

## Using ADO.NET Data Collections (LiveLinq to DataSet)

[LiveLinq](#) > [How to use LiveLinq](#) > [How to Query Collections with LiveLinq](#) > Using ADO.NET Data Collections (LiveLinq to DataSet)

ADO.NET **DataTable** can also be used in LiveLinq, for example:

C#	Copy Code
<pre> CustomersDataTable customers = ... var query =     from c in customers.AsIndexed() where c.City == "London" select c; </pre>	

For that, you'll need to add the following to your source file:

C#	Copy Code
<pre> using C1.LiveLinq.AdoNet; </pre>	

The **AsIndexed()** used will be

C1.LiveLinq.AdoNet.AdoNetExtensions.AsIndexed|keyword=AsIndexed Method (DataTable). It is specifically optimized for data in ADO.NET DataSets.

## Using XML Data (LiveLinq to XML)

[LiveLinq](#) > [How to use LiveLinq](#) > [How to Query Collections with LiveLinq](#) > Using XML Data (LiveLinq to XML)

LiveLinq can query data directly from XML in memory (stored in a LINQ to XML **XDocument** class). It dramatically speeds up LINQ to XML performance by supporting XML indexing (not supported by the regular LINQ to XML).

To use LiveLinq to XML, you need to add the following to your source file:

C#	Copy Code
<pre>using C1.LiveLinq.LiveViews.Xml;</pre>	

Then you need to create some live views over your XML data. Unlike LiveLinq to Objects and LiveLinq to DataSet, where you can query data without live views, in LiveLinq to XML queries and live views are always used together. You are using LiveLinq to XML versions of LINQ query operators if you apply them to a live view. Otherwise, they will be the standard, not LiveLinq query operators. To start creating live views, simply apply the extension method **AsLive()**|keyword=AsLive Method (XDocument) to an **XDocument**:

C#	Copy Code
<pre>XDocument doc = ... View&lt;XDocument&gt; docView = doc.AsLive();</pre>	

Once you have a live view, you can use the familiar (from LINQ to XML) query operators **Elements**, **Descendants** and others (see XmlExtensions Class|keyword=XmlExtensions class) as well as all Standard Query Operators Supported in Live Views|tag=Query Operators Supported in Live Views to define a live view. For example, the following code defines the collection of orders in the document:

C#	Copy Code
<pre>View&lt;XElement&gt; orders = docView.Descendants("Orders");</pre>	

This collection is live; it is automatically kept up to date with the data in the **XDocument** that can be changed by your program. As a result, you can work with data in this collection in the same way as you would work with any dynamic collection, including ADO.NET DataTable or DataView. In particular, you can create indexes over it and use them for speeding up queries and for fast programmatic searches as shown in other sections of this documentation. It means that LiveLinq to XML makes it possible to work with XML data in memory without losing performance, so it is no longer necessary to create custom collection classes or use ADO.NET or other frameworks working with XML data. All you need is LINQ to XML with addition of LiveLinq.

Once you have some live views defined over XML data, you can query them. For example, the following code gives you a query for orders of a particular customer:

C#	Copy Code
<pre>var query = from Order in orders.AsIndexed() where (string)Order.IndexedAttribute("CustomerID") == "ALFKI"</pre>	

**Note.** It is important to distinguish between live views and queries in LiveLinq to XML. Since you always start with a live view, your query will be a live view by default, unless you specify otherwise. In the example above, we used **AsIndexed()** to specify that we only need a query, don't need that query to define a live view. Live views extend query functionality, so they can be used instead of queries. However, live views have performance overhead, so you should avoid using a live view where a simple query would suffice. As a general rule, avoid creating live views and throwing them away after using them just once to get results. Live views are designed to remain active (hence they are called *live*) and show up-to-date data.

## Using Bindable Collection Classes (LiveLinq to Objects)

[LiveLinq](#) > [How to use LiveLinq](#) > [How to Query Collections with LiveLinq](#) > Using Bindable Collection Classes (LiveLinq to Objects)

Any bindable collection class (a class implementing change notifications for data binding) can be used in LiveLinq queries after it is wrapped in an **IndexedCollection<T>**-derived adapter class, which is done by applying a **ToIndexed()** extension method to it.

Using **ToIndexed()**, you can apply LiveLinq to various common data collections. For example, since the **EntityCollection** and **ObjectResult** classes of the ADO.NET Entity Framework are bindable, you can use ADO.NET Entity Framework data in LiveLinq queries and views.



# LiveLinq to Objects: IndexedCollection (T) and other Collection Classes

[LiveLinq](#) > [How to use LiveLinq](#) > [How to Query Collections with LiveLinq](#) > LiveLinq to Objects: IndexedCollection (T) and other Collection Classes

Querying general collection classes in LiveLinq is called LiveLinq to Objects, to distinguish it from LiveLinq to DataSet and LiveLinq to XML that query such specific data sources as DataSet and XML. So use LiveLinq to DataSet if you need to query data in a DataSet and LiveLinq to XML if your data is in XML. For other cases, we already saw two options in LiveLinq to Objects:

- a. Use **IndexedCollection<T>** if you don't have preexisting collection classes and rely on LiveLinq to supply them.
- b. **Apply ToIndexed()** to preexisting collection classes that support data binding.

And there is also a more advanced option that is not frequently needed:

- c. Define your own collection class, usually derived from **IndexedCollection<T>**, if you want some non-standard functionality.

And finally, an option that can also be considered advanced, but, actually, is not particularly difficult, and allows you to bring virtually any collection into the LiveLinq orbit:

- d. You can make any existing collection class **C** usable in LiveLinq provided that it somehow lets you know when changes occur. Then you can create an adapter class implementing the **C1.LiveLinq.IObservableSource<T>** interface and delegating most of its functionality to that preexisting collection class. Having an **IObservableSource<T>** implementation, you can apply the **ToIndexed()** keyword=**ToIndexed(T)** Method (**IObservableSource(T)**) extension method to it, it will return **IIndexedSource<T>**. LiveLinq extension methods implementing query operators take **IIndexedSource<T>** as their argument, see **IndexedQueryExtensions**.

## How to Create Indexes

[LiveLinq](#) > [How to use LiveLinq](#) > [How to Create Indexes](#)

Now that we can query our collections in LiveLinq, we need to create some indexes for them, otherwise LiveLinq won't query them faster than the standard LINQ does.

Suppose we have a collection implementing **IIndexedSource<Customer>**, for example, like this:

C#	Copy Code

```
var customers = new IndexedCollection<Customer>();
```

Or like the following:

C#

Copy Code

```
var customers = CustomersDataTable.AsIndexed();
```

The **IIndexedSource<T>** interface has an **Indexes** collection (which is actually the only thing it has), so we use that collection to create an index like this:

C#

Copy Code

```
var indexByCity = customers.Indexes.Add(x => x.City);
```

That creates an index of customers by city. Once the index is created, it will be automatically maintained on every change made to the **customers** collection. This maintenance comes with a performance cost. The cost is not high; index maintenance is fast and you can usually ignore it as long as you don't have too many indexes, but it may become a concern if you modify the collection very intensively.

To avoid heavy index maintenance cost, you can use the **BeginUpdate/EndUpdate** methods while making massive changes to a collection that has indexes attached to it, for example, while populating the collection.

This index will make queries such as the following code execute very fast because LiveLinq will use the index to go directly to the required items instead of searching for them by traversing the entire collection.

C#

Copy Code

```
from c in customers where c.City == "London";
```

Indexes can speed up many queries, not just this simple one, they can also speedup range conditions (with inequalities), joins, grouping, and other LINQ operators. See LiveLinq Query Performance: Tuning Indexing Performance for the full description of what classes of queries benefit from indexes.

Explicitly creating indexes by adding them to the collection as shown above is only needed if you want direct control over their lifetimes and/or direct access to the indexes (**Index<T>** objects) to use them programmatically (see How to use indexes programmatically). If all you

need is to optimize a few LiveLinq queries, you can use the alternative, implicit method of creating indexes, using so called *hints* in LiveLinq queries. A hint `.Indexed()`|keyword=`Indexed(T) Method (T)` is an extension method that can be applied to a property in a query. It does not change the value of that property; it only tells LiveLinq to create an index on that property (if possible). So, instead of creating an index by city explicitly as shown above, you could write the query like this:

C#	Copy Code
<pre>from c in customers where c.City.Indexed() == "London"</pre>	

That would tell LiveLinq to create an index by city if it has not been already created before. This hint does not affect the value of the property, that is, the value of `c.City.Indexed()` is the same as the value of `c.City`.

## How to use Indexes Programmatically

[LiveLinq](#) > [How to use LiveLinq](#) > [How to use Indexes Programmatically](#)

Indexes are used by LiveLinq to optimize query execution, but they are also accessible for programmatic use. You can use indexes directly in code, calling methods of the **Index<T>** class to perform a variety of fast searches. This makes LiveLinq indexes useful even outside the framework of LINQ, outside queries.

For example, searching for a particular value can look like this:

C#	Copy Code
<pre>indexByCity.Find("London")</pre>	

or even like this:

C#	Copy Code
<pre>indexByCity.FindStartingWith("L")</pre>	

The **Index<T>** class also has other methods for performing fast searches and fast joins and groupings that can be useful in any code, not just in queries: `FindGreater|keyword=FindGreater Method`, `FindBetween|keyword=FindBetween Method`, **Join**, **GroupJoin**, and a few others.

Examples of programmatic searches (without LINQ) can be found in the LiveLinq indexing demo, see Query Performance sample application (LiveLinqQueries). In fact, every query that is

shown in that demo has an alternative implementation via direct programmatic search in code without LINQ.

## How to Create a Live View

[LiveLinq](#) > [How to use LiveLinq](#) > [How to Create a Live View](#)

Consider a simple query like the following:

C#	Copy Code
<pre>from a in A where a.p == 1 select a</pre>	

In standard LINQ, the result of this query is a snapshot. The result collection is formed at the time when the query is executed and it does not change after that. If one of its objects changes so it no longer satisfies the condition, that object will not be removed from the result collection. And if an object in **A** changes so it now satisfies the condition, that object will not be added to the result collection. The result of even such a simple query is not live, not dynamic, does not change automatically when the base collection **A** changes, not kept in sync automatically with the base data.

In LiveLinq, if we create a view based on that query, it will be live, dynamic, will change automatically when the base data changes, will be automatically kept in sync with the base data.

All we need to do to make a view out of this query is to use the extension method **.AsLive()**:

C#	Copy Code
<pre>var view = from a in A.AsLive() where a.p == 1 select a;</pre>	

The **AsLive()** extension method is the analog of `AsIndexed()|keyword=AsIndexed Method/ToIndexed()`, it can be used everywhere where the `AsIndexed()|keyword=AsIndexed Method/ToIndexed()` extension methods can be used. So, live views are supported in all cases in LiveLinq to Objects, LiveLinq to XML and LiveLinq to DataSet (with some restrictions on the supported query operators, see Query Operators Supported in Live Views). The difference between `AsIndexed()|keyword=AsIndexed Method/ToIndexed()` and **AsLive()** is that **AsLive()** creates a view, thus enabling LiveLinq both to query to populate the view and to maintain the view after it has been initially populated. The `AsIndexed()|keyword=AsIndexed Method/ToIndexed()` extension methods do only the first part, enable LiveLinq querying.

The example above is a very simple one, it's just one simple Where condition. There are other tools that can do the same, for example, **DataView** in ADO.NET or **CollectionView** in WPF. The power of LiveLinq is that it supports most of the LINQ operators, including joins and others. So

you can create a live view based not just on a simple condition, but on virtually any query you need.

## How to Bind GUI Controls to a Live View

[LiveLinq](#) > [How to use LiveLinq](#) > [How to Bind GUI Controls to a Live View](#)

LiveLinq views can be used as data sources for GUI controls because they implement the data binding interfaces so you can simply bind any control to it as to any other data source. For example, in WinForms:

C#	Copy Code
<pre>View&lt;T&gt; view = from a in A.AsLive() join b in B.AsLive() on a.k equals b.k                where a.p == 5 select new {a, b}; dataGridView1.DataSource = view;</pre>	

Data binding has long been the tool of choice for most developers creating GUI, but its scope was limited to simple cases only. You could bind to your base data, like your base collections, tables of a data set, etc, but not to your derived data, not to data shaped by some query. Some shaping was supported by some tools, but it was very limited, mostly just filtering and sorting. If you had data derived/shaped in any more complex way, for example, with joins (a very common case), you could not use data binding (more exactly, you could only use it for one-time snapshot binding, show data once, no changes allowed).

LiveLinq makes data binding extremely powerful by removing these limitations. Now you have the full power of LINQ to bind to.

With live views, you can create entire GUI, sophisticated applications, virtually without procedural code, using declarative data binding alone. An example of such application is the [LiveLinqIssueTracker demo](#)<sup>tag=Live views sample application</sup> (LiveLinqIssueTracker);document=WordDocuments\LiveLinq-Samples.doc.

## How to Use Live Views in Non-GUI Code

[LiveLinq](#) > [How to use LiveLinq](#) > [How to Use Live Views in Non-GUI Code](#)

Using live views is not limited to GUI. In a more general way, live views enable a declarative style of programming which we tentatively call **view-oriented programming**. Non-GUI, batch processing code can also be made declarative using live views.

Generally speaking, any application operates on some data, and any data can be seen as either based or derived data. Base data contains the main objects your application is dealing with, like Customers, Orders, Employees, et cetera. Those objects (collections containing all objects of a certain class, sometimes called the *extent* of a class) are the objects that your application

modifies when it needs to make some change in the base data. But applications rarely operate on this base data directly, they rarely directly operate on the entire extent of a class. Usually they filter and shape those extents and combine them together to get to a slice of data they need for a particular operation. This shaping/filtering/joining/slicing of data is the process of getting **derived** data (as opposed to **base**, underlying data). Before the emergence of LINQ, it was done by purely procedural code, with all the ensuing complexity. LINQ made it declarative, which is a big step forward. But, although declarative, it is not live, not dynamic, and does not react to changes in base data automatically. Accordingly, its reaction to change is not declarative; the programmer needs to react to changes in procedural, imperative code. Complexity is diminished by LINQ, but reacting to change remains complex. And reacting to change is pervasive because change is everywhere.

LiveLinq live views make reacting to change declarative too, thus closing the circle of declarativeness. Now entire applications, not just GUI, can be made virtually entirely declarative.

To give just a small example, we can consider a sample in the LiveLinqIssueTracker demo|tag=Live views sample application (LiveLinqIssueTracker);document=WordDocuments\LiveLinq-Samples.doc. It has two operations performed on some issue data:

- (1) Assign as many unassigned issues to employees as possible, using information on pending issues, product features (every issue belongs to a feature) and assignments.
- (2) Collect information on open issues for given employees.

Each of these two operations (and in a real application there is, of course, many more operations like this) works on data shaped by a query with joins (see the actual queries in Live Views How To: Use Live Views to Create Non-GUI Applications as a Set of Declarative Rules Instead of Procedural Code). Both operations can be performed more than once during program execution. Data is changing while these and other operations on data are performed. Live views used to implement these operations change automatically with that changing data, so the operations can be kept simple and completely declarative, and as a result of that, robust, reliable and flexible, easily modifiable when business requirements change.

There is nothing else needed for this style of programming in addition to what we already know about how to create live views.

## Programming Guide

The following sections provide information concerning **LiveLinq** programming.

## Working with Entities in Code

[Programming Guide](#) > Working with Entities in Code

**Entity Framework DataSource (EF DataSource)** is non-intrusive in the sense that you can do whatever you want with entities in code using the regular Entity Framework (or RIA) methods and properties and that will work well with **EF DataSource**. You can add, delete and modify

entities using the regular methods (no special **EF DataSource** object model is needed for that), and **EF DataSource** collections will automatically reflect changes that you make to the entities. For example, to add a new entity, you don't need to use some special **EF DataSource** method; therefore, none exists. Just add a new entity as you always do in EF (or RIA) and it will automatically appear in corresponding **EF DataSource** collections. If an **EF DataSource** collection has a *Where* condition, it will appear there only if it satisfies the condition. Same with deleting and modifying entities: do it the regular EF (or RIA) way, and **EF DataSource** reflects the changes automatically, so they are automatically reflected in bound controls.

However, there is one important restriction that must be strictly observed. It does not limit what you can do; it only requires you in some (relatively rare) cases to add a method call to your code notifying **EF DataSource** of what you are doing:



## CAUTION

**Never fetch entities from the server by directly querying the context without notifying EF DataSource.**

All entities must be either fetched by one of the **EF DataSource**'s **GetItems** methods, or, if you want to fetch them by querying an object context directly, you must call the **ClientScope.AddRef** method to notify **EF DataSource** that you fetched entities from the server.

If you use the **C1DataSource** control or **ClientViewSource**, you fetch entities either implicitly if **AutoLoad** = true, or explicitly when you call the **ClientViewSource.Load** method. In both cases it is done through **EF DataSource**, so **EF DataSource** is aware of the newly fetched entities.

When you need to fetch entities in code without using **ClientViewSource**, the standard way to do it is by using one of the **EF DataSource**'s **GetItems** methods

(**EntityClientScope.GetItems**|keyword=**EntityClientScope.GetItems** method,

**RiaClientScope.GetItems**|keyword=**RiaClientScope.GetItems** method). Here too, **EF DataSource**

is aware of the fetched entities. However, occasionally you may need to retrieve entities by issuing a query directly to the server without involving **EF DataSource**. For example, in Entity Framework you can take an **ObjectContext** that is used by **EF DataSource** and create queries:

C#	Copy Code
<pre> ObjectContext context = scope.ClientCache.ObjectContext; // or ObjectQuery query = ((NORTHWNDEntities)context).Customers; // or query = context.CreateObjectSet&lt;Customer&gt;(); // or query = context.CreateQuery&lt;Customer&gt;("..."); </pre>	

In RIA Services such code can look like this:

C#	Copy Code
<pre>DomainContext context = scope.ClientCache.DomainContext; var query = ((DomainService1)context).Customers; // or var entities = context.Load( ((DomainService1)context).GetCustomersQuery()).Entities;</pre>	

Having a query, you can get entities directly from the server by enumerating the query result:

C#	Copy Code
<pre>foreach (Customer c in query) /* do something */</pre>	

or by binding a control to it:

C#	Copy Code
<pre>dataGrid.ItemsSource = query;</pre>	

If you do this without calling **ClientScope.AddRef**, you can get entities from the server without **EF DataSource** knowing about it. Those entities will be in the same cache with other entities, indistinguishable from them, so **EF DataSource** will manage them as other entities, including possibly releasing them, evicting them from the cache when it does not need them. That will make the entities inaccessible, but your program may still need them! So, very bad things can happen if you fetch entities to a **EF DataSource**-governed context without using **EF DataSource** and without notifying it that entities are fetched. Fortunately, adding that notification is easy, just call **ClientScope.AddRef** for all fetched entities like this:

C#	Copy Code
<pre>foreach (Customer c in query) {</pre>	



```
scope.AddRef(c);  
// do something  
}
```

It will tell **EF DataSource** that the entities should not be released as long as the scope is alive.

## Using ClientView in Code

[Programming Guide](#) > Using ClientView in Code

**Entity Framework DataSource (EF DataSource)** supports both visual and "all code" style of programming:

- Use the **C1DataSource** control if you want point-and-click, RAD-style application development.
- Use the **ClientViewSource** class if you want to separate your code from GUI but keep the ease of use of the **C1DataSource**. The **ClientViewSource** class is independent of GUI; it can be used in code with any of the GUI platforms (WPF, Silverlight, WinForms). It can be used completely separately from GUI, including in the view model layer of an MVVM application. At the same time, the **ClientViewSource** is the same object **C1DataSource** uses, so you can keep the ease of use characteristic of **C1DataSource** (but code-only, without visual designers). In fact, **C1DataSource** is just a collection of **ClientViewSource** objects plus visual designer support for them.
- For the most complete control over your code, you can use the third, lowest level of the **EF DataSource** object mode: the **ClientView** class. If you prefer pure code, especially (but not only) using the MVVM pattern, this is the recommended level. It is still easy to program with, it is mostly based on LINQ which promotes a functional style of programming, intuitive and expressive.

The rest of this section is devoted to programming using the **ClientView** class.

## Client-Side Transactions

[Programming Guide](#) > Client-Side Transactions

**Entity Framework DataSource (EF DataSource)** gives developers a powerful mechanism for canceling changes on the client without involving the server. It is called *transaction* because it is similar to the common concept of database transaction in that it allows you to ensure that a certain group of changes (unit of work) is either made in its entirety or canceled in its entirety—that your code never makes incomplete or inconsistent changes to entities in memory. It is important to understand that these transactions have no affect in any way and are completely

independent from database transactions. To distinguish them from database transactions, we sometimes call them *client-side transactions*.

Client-side transactions are especially useful in implementing **Cancel/Undo** buttons/commands. Doing this without **EF DataSource** requires cancelling all changes in the object context. **EF DataSource** client-side transactions make partial canceling of changes possible. And the transactions can even be nested, so you can have, for example, a dialog box with a **Cancel** button (which cancels only the changes made in that dialog box, not all changes in the object context made elsewhere in the application), and from that dialog box you can open another dialog box with its own **Cancel** button. Using a nested (child) transaction in the child dialog ensures that its **Cancel** button cancels (rolls back) only the changes made in the child dialog box, so the user can return to editing data in the parent dialog box and then accept or cancel changes in it using the parent transaction.

The easiest way to work with client-side transactions is by associating them with live views. For example, if we have a data grid bound to a live view:

C#	Copy Code
<pre>var ordersView = from o in customer.Orders.AsLive()                   select new OrderInfo                   {                       OrderID = o.OrderID,                       OrderDate = o.OrderDate,                   }; dataGrid1.ItemsSource = ordersView;</pre>	

we can create a transaction and associate it with the view like this:

C#	Copy Code
<pre>var transaction = _scope.ClientCache.CreateTransaction(); ordersView.SetTransaction(transaction);</pre>	

To create a child (nested) transaction, instead of calling the method **ClientCacheBase.CreateTransaction**, use the **ClientTransaction** constructor by passing it the parent transaction as a parameter:

C#	Copy Code

```
var transaction = new ClientTransaction(parentTransaction);.
```

Once a transaction is associated with a view by calling `View.SetTransaction|keyword=SetTransaction` method, it tracks all changes made through that view, via data binding (for example, changes made by the end user in the grid bound to the view as in the example above) as well as programmatically in code. Rolling back the transaction (calling **`ClientTransaction.Rollback`**) cancels the changes tracked by that transaction.

It is often necessary to bind GUI controls to a single object (as opposed to binding to a collection of objects). A single object can't be represented by a live view, so we don't have the convenience of `View.SetTransaction|keyword=SetTransaction` method, but in this case we can use the **`ClientTransaction.ScopeDataContext`** method. In WPF and Silverlight, you can use this method to set the **`DataContext`** so it will be used for data bindings specified in XAML:

```
DataContext = transaction.ScopeDataContext(order);
```

The resulting **`DataContext`** wraps the original one and performs the same data binding but with the additional benefit of all changes made through that data binding being tracked by the 'transaction', so they can be rolled back if necessary.

In WinForms, you can use the same **`ScopeDataContext`** and bind to the resulting object, for example, like this:

C#	Copy Code
<pre>var dataContext = transaction.ScopeDataContext(order); textBox.DataBindings.Add(new Binding("Text", dataContext, "OrderDate"));</pre>	

Finally, sometimes you need to change (or add or delete) some entities in code, not through a live view or data binding, and want those changes to be tracked by a transaction. You can do it using the **`ClientTransaction.Scope()`** method. That method opens a *scope* for a transaction. When you modify entities while in the scope of a transaction, those changes are tracked by that transaction. That method is designed to be used with the 'using' construct that conveniently closes the scope on exit, like this:

C#	Copy Code
<pre>using (transaction.Scope()) {     Customer.Orders.Add(order); }</pre>	

```
}
```

All three methods of using transactions described above are demonstrated in the **Transactions** sample project that comes with **EF DataSource**. It also demonstrates how a form with a **Cancel** button can be implemented inside another form that also has a **Cancel** button using child (nested) transactions.

## Virtual Mode

[Programming Guide](#) > Virtual Mode

Virtual mode is the technology allowing **Entity Framework DataSource (EF DataSource)** to provide data binding of GUI controls to very large data sets with no delays, no code, and without resorting to paging, with a simple setting of a property. This property is **ClientViewSource.VirtualMode**. It has three possible values:

Value	Discription
<b>None</b>	Virtual mode is not enabled. This is the default value.
<b>Managed</b>	<p>Virtual mode (fetching data from the server) is driven by a grid control. The grid control driving it is specified by setting an extender (attached) property on the grid. Most popular grids are supported, but not all grids (see the list below). If you need virtual mode with one of the supported grids, use the <b>Managed</b> option, do not use <b>Unmanaged</b>. Performance is optimal with the <b>Managed</b> option because it is specifically optimized for the grid in question. In addition to the grid control there can be other simple controls bound to the same data source (for example, TextBox controls), but no "complex" bound control (for example, another grid or list) in addition to the driving grid should be used.</p> <p><b>Managed</b> mode is currently supported for the following grid controls:</p> <p><b>ComponentOne:</b> C1FlexGrid (WinForms, WPF, Silverlight), C1DataGrid (WPF, Silverlight).</p> <p><b>Microsoft:</b> DataGridView (WinForms), DataGrid (WPF), DataGrid (Silverlight; but see below about a problem with Microsoft DataGrid for Silverlight).</p>
<b>Unmanaged</b>	Virtual mode (fetching data from the server) is driven by the data source itself, regardless of what type of controls are bound to it. Use this option if you need

	<p>virtual mode with a grid/list/etc. control that does not belong to the list of supported grids below (but note that "simple" controls such as TextBox that are only bound to the current record and not to the entire data set can be used without limitations with either <b>Managed</b> or <b>Unmanaged</b> mode). Performance in <b>Unmanaged</b> is almost as good as in Managed, but it is subject to some limitations. See <b>Unmanaged</b> virtual mode limitations for more information. Those limitations are not very restricting, but they cannot be checked automatically, so, before deciding to use the <b>Unmanaged</b> option, make sure you do not break the limitation rules document=WordDocuments\C1DataStudio-ProgrammingGuide.docx;topic=Unmanaged virtual mode limitations. Also, in <b>Unmanaged</b> mode, the <b>PageSize</b> property must be set to a number greater than your GUI control will show at any given time (PageSize is ignored in Managed mode).</p>
--	---

## Unmanaged Virtual Mode Limitations

[Programming Guide](#) > [Virtual Mode](#) > Unmanaged Virtual Mode Limitations

The difference between **Managed** and **Unmanaged** options is seen when data is retrieved from the server. In **Managed** mode, data is retrieved when the "driving" grid needs it, for example, when the user scrolls or navigates in the grid. In **Unmanaged** mode data is fetched from the server (if it is not already present in the cache) on every data request a bound control makes, without taking into consideration which control makes the request and for what purpose. The data source fetches

**ClientViewSource.PageSize**|tag=P\_C1\_Data\_DataSource\_ClientCollectionView\_PageSizerows around every row that is requested from it. But it does not keep/hold those rows because (unlike in **Managed** mode) it does not know what rows are visible in bound controls. It holds only the current row and the last requested row (both with

**ClientViewSource.PageSize**|tag=P\_C1\_Data\_DataSource\_ClientCollectionView\_PageSizerows before and

**ClientViewSource.PageSize**|tag=P\_C1\_Data\_DataSource\_ClientCollectionView\_PageSize rows after them). All other rows are not held, that is, they can be released, evicted from cache, when

**Entity Framework DataSource** needs to cleanup/compact the cache. In most common scenarios this cannot happen, and in general you can prevent it from happening by making sure that you

**Never bind more than one scrollable control to a data source in Unmanaged virtual mode.**

You can bind any number of simple (bound to a single row) controls like TextBox. You can bind any grid or list or other "scrollable" control (bound to the entire data set as opposed to a single row) as well. But you should never bind two scrollable controls that can be scrolled to two different locations (rows) in the data set that are more than

**ClientViewSource.PageSize**<sup>tag=P\_C1\_Data\_DataSource\_ClientCollectionView\_PageSize</sup> apart from one another, because only one (the latest requested) of these rows will be held, the other can be released by **Entity Framework DataSource** (at unpredictable time).

## Other Virtual Mode Limitations

[Programming Guide](#) > [Virtual Mode](#) > Other Virtual Mode Limitations

- Binding two independently scrollable controls (such as grids or lists that can be scrolled to different, independent locations in the data set) to a single data source in virtual mode is not supported in **Managed** mode as well as in **Unmanaged**, and for the same reason. The difference is just that in **Managed** mode it is less dangerous because you will immediately see the effect; there is no element of unpredictability in the behavior. If in addition to the main "driving" bound control there is another scrollable control bound to the same data source, that control is not "driving", that is, scrolling in it does not fetch data; therefore, if you scroll beyond the rows that are visible in the main control, you will see empty rows.
- In both **Managed** and **Unmanaged** modes, grids, lists, and other controls bound to the entire data set (as opposed to a single record), must not do anything with all rows of the data source at once. In other words, a control should not perform actions in its code that involve going in a loop over all rows of the data source and doing something for each row. That is for the simple reason that the number of rows in virtual mode can be very large, many thousands or even millions of rows. It is, in fact, unlimited. Properly designed controls that do not rely on an assumption that the number of rows in its data source is small, will do fine in **Entity Framework DataSource (EF DataSource)** virtual mode. But those that were designed specifically for small data sources, those that loop through all their rows, can cause long delays with very large number of rows. C1FlexGrid and C1DataGrid (for WinForms/WPF/Silverlight) are OK, as are Microsoft DataGridView (WinForms) and Microsoft DataGrid for WPF. But Microsoft DataGrid for Silverlight (in its current version, Silverlight 4) is not recommended for **EF DataSource** virtual mode because it loops through all rows to compute its height.

# API Reference









## C1.Data.Entity.4 Assembly



### Namespaces

#### C1.Data Namespace

#### Overview

#### Classes

	Class	Description
	<a href="#">ClientCacheBase</a>	Represents the client-side cache, the central hub of data access in the Studio for Entity Framework.
	<a href="#">ClientScope</a>	Defines a scope of data access. Provides facilities to create <a href="#">client views</a> .
	<a href="#">ClientView&lt;T&gt;</a>	Represents a <i>client view</i> , that is, a <a href="#">live view</a> that is connected to a remote source, such as an <b>System.Data.Objects.ObjectContext</b> .
	<a href="#">ClientViewLoadedEventArgs</a>	Provides data for the <a href="#">ClientView&lt;T&gt;.Loaded</a> event.
	<a href="#">DataExtensions</a>	Extension methods provided by Studio for Entity Framework.
	<a href="#">FilteredView&lt;T&gt;</a>	Represents a <a href="#">client view</a> filtered by a filter key function, an <a href="#">operator</a> and a <a href="#">filter key value</a> .
	<a href="#">PageChangingEventArgs</a>	Provides data for the <a href="#">C1.Data.DataSource.ClientCollectionView.PageChanging</a> event.
	<a href="#">PagingView&lt;T&gt;</a>	Represents a paged <a href="#">client view</a> .

	<a href="#">ProgressiveView&lt;T&gt;</a>	Represents a <a href="#">client view</a> that loads entities sequentially (progressively) page by page. The user sees the result and can interact with it before all pages are loaded.
	<a href="#">SavedChangesEventArgs</a>	Provides data for the <code>C1DataSource.SavedChanges</code> event.

## See Also

### Reference

[C1.Data.Entity.4 Assembly](#)

## Classes

### ClientCacheBase

[C1.Data Namespace](#) : ClientCacheBase Class

Represents the client-side cache, the central hub of data access in the Studio for Entity Framework.

## Object Model

ClientCacheBase

## Syntax

Visual Basic (Declaration)	
<b>Public MustInherit Class</b> ClientCacheBase	
C#	
<b>public abstract class</b> ClientCacheBase	

## Remarks

Usually, a single instance of this class is created on application startup with an `ObjectContext/DomainContext` as a parameter and exists during the entire application lifetime, while each form, window, or user control works with data using a [ClientScope](#) created by calling the [CreateScope](#) method.

It is the base class for platform-specific implementations, such as `C1.Data.Entities.EntityClientCache` (Entity Framework), `C1.Silverlight.Data.RiaServices.RiaClientCache` (RIA Services).



# Inheritance Hierarchy

System.Object  
    **C1.Data.ClientCacheBase**  
        C1.Data.Entities.EntityClientCache  
        C1.Silverlight.Data.RiaServices.RiaClientCache

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Members](#)  
[C1.Data Namespace](#)

## Overview

[C1.Data Namespace](#) : ClientCacheBase Class

Represents the client-side cache, the central hub of data access in the Studio for Entity Framework.

## Object Model

ClientCacheBase

## Syntax

Visual Basic (Declaration)	
<code>Public MustInherit Class ClientCacheBase</code>	
C#	
<code>public abstract class ClientCacheBase</code>	

## Remarks

Usually, a single instance of this class is created on application startup with an ObjectContext/DomainContext as a parameter and exists during the entire application lifetime, while each form, window, or user control works with data using a [ClientScope](#) created by calling the [CreateScope](#) method.

It is the base class for platform-specific implementations, such as  
[C1.Data.Entities.EntityClientCache](#) (Entity Framework),  
[C1.Silverlight.Data.RiaServices.RiaClientCache](#) (RIA Services).

## Inheritance Hierarchy

System.Object

**C1.Data.ClientCacheBase**

[C1.Data.Entities.EntityClientCache](#)

[C1.Silverlight.Data.RiaServices.RiaClientCache](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Members](#)

[C1.Data Namespace](#)

## Members






[Methods](#)

[C1.Data Namespace](#) : [ClientCacheBase](#) Class

The following tables list the members exposed by [ClientCacheBase](#).

## Public Methods

	Name	Description
≡	<a href="#">BulkChanges</a>	Used to group massive changes to entities and to allow manual explicit changes of entity states.
≡	<a href="#">CleanupCache</a>	Forces unused memory to be released, unused entities to be detached from the context. It is usually done automatically, so programmers rarely need to call this method in code.
≡	<a href="#">Clear</a>	Clears client-side cache entirely. Call this method if you want to make sure that following queries will fetch fresh data from the server.

	<a href="#">CreateScope</a>	Creates a <a href="#">ClientScope</a> that defines a scope of data access.
	<a href="#">CreateTransaction</a>	Creates a <a href="#">C1.Data.Transactions.ClientTransaction</a> that allows you to easily cancel changes made in transaction scope.
	<a href="#">Refresh</a>	Refreshes data in all <a href="#">C1DataSource</a> controls connected to this <a href="#">ClientCacheBase</a> .
	<a href="#">RejectChanges</a>	Reverts all pending changes for this <a href="#">ClientCacheBase</a> . It is recommended to call this method instead of <code>%System.Data.Objects.ObjectContext.Refresh(System.Data.Objects.RefreshMode, object)%</code> .
	<a href="#">SaveChanges</a>	Persists all changes to the server. It is recommended to call this method instead of <b><code>System.Data.Objects.ObjectContext.SaveChanges()</code></b> .

[Top](#)

## See Also

### Reference

[ClientCacheBase Class](#)



[C1.Data Namespace](#)







## Methods

[C1.Data Namespace](#) : [ClientCacheBase Class](#)

For a list of all members of this type, see [ClientCacheBase members](#).

## Public Methods

	Name	Description
	<a href="#">BulkChanges</a>	Used to group massive changes to entities and to allow manual explicit changes of entity states.
	<a href="#">CleanupCache</a>	Forces unused memory to be released, unused entities to be detached from the context. It is usually done automatically, so programmers rarely need to call this method in code.

	<a href="#">Clear</a>	Clears client-side cache entirely. Call this method if you want to make sure that following queries will fetch fresh data from the server.
	<a href="#">CreateScope</a>	Creates a <a href="#">ClientScope</a> that defines a scope of data access.
	<a href="#">CreateTransaction</a>	Creates a <a href="#">C1.Data.Transactions.ClientTransaction</a> that allows you to easily cancel changes made in transaction scope.
	<a href="#">Refresh</a>	Refreshes data in all C1DataSource controls connected to this ClientCacheBase.
	<a href="#">RejectChanges</a>	Reverts all pending changes for this <a href="#">ClientCacheBase</a> . It is recommended to call this method instead of %System.Data.Objects.ObjectContext.Refresh(System.Data.Objects.RefreshMode, object)%.
	<a href="#">SaveChanges</a>	Persists all changes to the server. It is recommended to call this method instead of <b>System.Data.Objects.ObjectContext.SaveChanges()</b> .

[Top](#)

## See Also

### Reference

[ClientCacheBase Class](#)

[C1.Data Namespace](#)

### BulkChanges Method

[Example](#)

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : BulkChanges Method

A delegate that makes changes in entities.

Used to group massive changes to entities and to allow manual explicit changes of entity states.

## Syntax

Visual Basic (Declaration)

```
Public Sub BulkChanges( _  
    ByVal makeChanges As System.Action _
```

```
)
```

C#

```
public void BulkChanges(  
    System.Action makeChanges  
)
```

## Parameters

*makeChanges*

A delegate that makes changes in entities.

## Remarks

Internal state of the client-side cache and all existing [client views](#) based on the cache are kept unchanged, aren't updated while the given [makeChanges](#) is executed. After the delegate completes its execution (having modified multiple entities), the client-side cache internal state is restored and client views are updated (maintained) to reflect the changes made in entities during the delegate's execution.

There are two main scenarios where you should consider calling this method:

1. Using this method when you make a lot of changes to entities can improve performance because the change processing is deferred, occurs only once after all changes are done instead of every time on each change. Depending on the amount of changes, the speedup can be considerable.
2. You must use this method when you need to make changes to entity states by calling any of the following methods:
  - `System.Data.Objects.ObjectStateEntry.ChangeState/SetModified/AcceptChanges`,
  - `System.Data.Objects.ObjectContext.AcceptAllChanges`,
  - `System.ServiceModel.DomainServices.Client.DomainContext.RejectChanges`,
  - `System.ServiceModel.DomainServices.Client.Entity.AcceptChanges/RejectChanges`,
  - `System.ServiceModel.DomainServices.Client.EntitySet.AcceptChanges/RejectChanges`,
  - `System.Windows.Controls.DomainDataSource.RejectChanges`.

Calling these methods without wrapping them with [BulkChanges](#) can corrupt the client-side cache.

## Example

- C#

```
var scope = clientCache.CreateScope();  
clientCache.BulkChanges(delegate {  
    foreach(var detail in scope.GetItems<Order_Details>())
```

```

        detail.Discount *= 2;
    });

```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)  
[ClientCacheBase Members](#)

### CleanupCache Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : CleanupCache Method

Forces unused memory to be released, unused entities to be detached from the context. It is usually done automatically, so programmers rarely need to call this method in code.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> CleanupCache()	
C#	
<b>public void</b> CleanupCache()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)  
[ClientCacheBase Members](#)

### Clear Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : Clear Method

Clears client-side cache entirely. Call this method if you want to make sure that following queries will fetch fresh data from the server.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Clear()	
C#	
<b>public void</b> Clear()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

### CreateScope Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : CreateScope Method

Creates a [ClientScope](#) that defines a scope of data access.

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> CreateScope() <b>As</b> <a href="#">ClientScope</a>	
C#	
<b>public</b> <a href="#">ClientScope</a> CreateScope()	

### Return Value

A new [ClientScope](#).

## Remarks

Usually, every form, window, or user control creates a [ClientScope](#) and uses it to access entities.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

[ClientScope Class](#)

### CreateTransaction Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : CreateTransaction Method

Creates a [C1.Data.Transactions.ClientTransaction](#) that allows you to easily cancel changes made in transaction scope.

## Syntax

Visual Basic (Declaration)

```
Public Function CreateTransaction() As ClientTransaction
```

C#

```
public ClientTransaction CreateTransaction()
```

### Return Value

A new [C1.Data.Transactions.ClientTransaction](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)



## Refresh Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : Refresh Method

Refreshes data in all C1DataSource controls connected to this ClientCacheBase.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Refresh()	
C#	
<b>public void</b> Refresh()	

## Remarks

This method calls C1DataSource.Refresh() for every C1DataSource connected to this ClientCacheBase. Changes made on the client are preserved.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

## RejectChanges Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : RejectChanges Method

Reverts all pending changes for this [ClientCacheBase](#). It is recommended to call this method instead of %System.Data.Objects.ObjectContext.Refresh(System.Data.Objects.RefreshMode, object)%.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> RejectChanges()	

C#	
----	--

<code>public void RejectChanges()</code>	
--	--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

### SaveChanges Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : SaveChanges Method

Persists all changes to the server. It is recommended to call this method instead of **System.Data.Objects.ObjectContext.SaveChanges()**.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<code>Public Sub SaveChanges()</code>	
---------------------------------------	--

C#	
----	--

<code>public void SaveChanges()</code>	
--	--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

# ClientScope

[C1.Data Namespace](#) : ClientScope Class

Defines a scope of data access. Provides facilities to create [client views](#).

## Object Model

ClientScope

## Syntax

Visual Basic (Declaration)	
<code>Public Class ClientScope</code>	
C#	
<code>public class ClientScope</code>	

## Remarks

Usually, one scope is created per form/window. Entities [pinned to the scope \(marked as needed\)](#) are not evicted from the cache until the scope is [disposed](#) or collected by the garbage collector.

This class is a base class for platform-specific scopes, such as [C1.Data.Entities.EntityClientCache](#) (Entity Framework) and [C1.Silverlight.Data.RiaServices.RiaClientCache](#) (RIA Services).

## Inheritance Hierarchy

- System.Object
  - C1.Data.ClientScope**
    - [C1.Data.Entities.EntityClientScope](#)
    - [C1.Silverlight.Data.RiaServices.RiaClientScope](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

## Overview

[C1.Data Namespace](#) : ClientScope Class

Defines a scope of data access. Provides facilities to create [client views](#).

## Object Model

ClientScope

## Syntax

Visual Basic (Declaration)

```
Public Class ClientScope
```

C#

```
public class ClientScope
```

## Remarks

Usually, one scope is created per form/window. Entities [pinned to the scope](#) (marked as [needed](#)) are not evicted from the cache until the scope is [disposed](#) or collected by the garbage collector.

This class is a base class for platform-specific scopes, such as [C1.Data.Entities.EntityClientCache](#) (Entity Framework) and [C1.Silverlight.Data.RiaServices.RiaClientCache](#) (RIA Services).

## Inheritance Hierarchy

System.Object

**C1.Data.ClientScope**

[C1.Data.Entities.EntityClientScope](#)

[C1.Silverlight.Data.RiaServices.RiaClientScope](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference


## Members

[Properties](#) [Methods](#)

[C1.Data Namespace](#) : ClientScope Class


The following tables list the members exposed by [ClientScope](#).

### Public Constructors

	Name	Description
	<a href="#">ClientScope Constructor</a>	Initializes a new instance of <a href="#">ClientScope</a> class with the given <a href="#">ClientCacheBase</a> .




[Top](#)

### Public Properties

	Name	Description
	<a href="#">ClientCache</a>	Gets the <a href="#">ClientCacheBase</a> to which this <a href="#">client scope</a> is connected.

[Top](#)

### Public Methods

	Name	Description
	<a href="#">AddRef</a>	Overloaded. Marks an entity as needed. Needed entities are not detached/released from the context until the client scope is disposed.
	<a href="#">Dispose</a>	Marks the scope as disposed. Entities that were marked needed by a disposed scope may be disposed of (evicted from the cache, detached from context) unless they are needed by other active scopes.
	<a href="#">Release</a>	Overloaded. Unmark a needed entity.

[Top](#)

### See Also

## Reference

[ClientScope Class](#)

[C1.Data Namespace](#)

## ClientScope Constructor

[C1.Data Namespace](#) > [ClientScope Class](#) : ClientScope Constructor

An instance of the [ClientCacheBase](#) class to which the new [client scope](#) is connected.

Initializes a new instance of [ClientScope](#) class with the given [ClientCacheBase](#).

## Syntax

Visual Basic (Declaration)

```
Public Function New( _  
    ByVal clientCache As ClientCacheBase _  
)
```

C#

```
public ClientScope(  
    ClientCacheBase clientCache  
)
```

## Parameters

*clientCache*

An instance of the [ClientCacheBase](#) class to which the new [client scope](#) is connected.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientScope Class](#)

[ClientScope Members](#)

## Methods

[C1.Data Namespace](#) : ClientScope Class

For a list of all members of this type, see [ClientScope members](#).

## Public Methods

	Name	Description
⇒💎	<a href="#">AddRef</a>	Overloaded. Marks an entity as needed. Needed entities are not detached/released from the context until the client scope is disposed.
⇒💎	<a href="#">Dispose</a>	Marks the scope as disposed. Entities that were marked needed by a disposed scope may be disposed of (evicted from the cache, detached from context) unless they are needed by other active scopes.
⇒💎	<a href="#">Release</a>	Overloaded. Unmark a needed entity.

[Top](#)

## See Also

### Reference

[ClientScope Class](#)

[C1.Data Namespace](#)

### AddRef Method

[C1.Data Namespace](#) > [ClientScope Class](#) : AddRef Method

Marks an entity as needed. Needed entities are not detached/released from the context until the client scope is disposed.

## Overload List

Overload	Description
<a href="#">AddRef(Object)</a>	Marks an entity as needed. Needed entities are not detached/released from the context until the client scope is disposed.
<a href="#">AddRef(Type)</a>	Mark all entities of a given type as needed. All entities of that type will not be detached from the context until the client scope is disposed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientScope Class](#)

[ClientScope Members](#)

AddRef(Object) Method

[C1.Data Namespace](#) > [ClientScope Class](#) > [AddRef Method](#) : AddRef(Object) Method

An entity to be marked as needed.

Marks an entity as needed. Needed entities are not detached/released from the context until the client scope is disposed.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Sub AddRef( _     ByVal entity As System.Object _ )</pre>	
C#	
<pre>public void AddRef(     System.object entity )</pre>	

### Parameters

*entity*

An entity to be marked as needed.

## Remarks

Client views and C1DataSource classes mark entities as needed automatically. Use this method only when you fetch entities using other means, bypassing Studio for EF classes with direct access to the underlying object context. When you no longer need those entities, call [Release\(Object\)](#). AddRef and Release are counting, every AddRef call must be balanced by a Release call.

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientScope Class](#)  
[ClientScope Members](#)  
[Overload List](#)

#### AddRef(Type) Method

[C1.Data Namespace](#) > [ClientScope Class](#) > [AddRef Method](#) : AddRef(Type) Method

An entity type to mark as needed.

Mark all entities of a given type as needed. All entities of that type will not be detached from the context until the client scope is disposed.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Sub AddRef( _     ByVal entityType As System.Type _ )</pre>	
C#	
<pre>public void AddRef(     System.Type entityType )</pre>	

### Parameters

*entityType*

An entity type to mark as needed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientScope Class](#)  
[ClientScope Members](#)  
[Overload List](#)

## Dispose Method

[C1.Data Namespace](#) > [ClientScope Class](#) : Dispose Method

Marks the scope as disposed. Entities that were marked needed by a disposed scope may be disposed of (evicted from the cache, detached from context) unless they are needed by other active scopes.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Dispose()	
C#	
<b>public void</b> Dispose()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientScope Class](#)  
[ClientScope Members](#)

## Release Method

[C1.Data Namespace](#) > [ClientScope Class](#) : Release Method

Unmark a needed entity.

## Overload List

Overload	Description
<a href="#">Release(Object)</a>	Unmark a needed entity.

<a href="#">Release(Type)</a>	Unmark a needed entity type. Calling this method does not release memory.
-------------------------------	---

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientScope Class](#)

[ClientScope Members](#)

[Release\(Object\) Method](#)

[C1.Data Namespace](#) > [ClientScope Class](#) > [Release Method](#) : [Release\(Object\) Method](#)

An entity that was marked as needed using [AddRef\(Object\)](#).

Unmark a needed entity.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Release( _     ByVal entity As System.Object _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool Release(     System.object entity )</pre>	

### Parameters

*entity*

An entity that was marked as needed using [AddRef\(Object\)](#).

### Return Value

True if the [entity](#) was unmarked; otherwise, False (the entity is not unmarked until every [AddRef](#) is balanced by a [Release](#)).

## Remarks

Calling this method does not release memory by itself. The [entity](#) becomes unneeded, so it can be disposed of at cache cleanup time.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientScope Class](#)  
[ClientScope Members](#)  
[Overload List](#)

Release(Type) Method

[C1.Data Namespace](#) > [ClientScope Class](#) > [Release Method](#) : Release(Type) Method

An entity type that was marked as needed using [AddRef\(Type\)](#).

Unmark a needed entity type. Calling this method does not release memory.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Release( _     ByVal entityType As System.Type _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool Release(     System.Type entityType )</pre>	

### Parameters

*entityType*

An entity type that was marked as needed using [AddRef\(Type\)](#).

### Return Value

True if the entity type was unmarked (every Addrref was balanced with Release); otherwise, False.

## Remarks

Calling this method does not release memory until cache cleanup time.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference


[ClientScope Class](#)  
[ClientScope Members](#)  
[Overload List](#)

## Properties

[C1.Data Namespace](#) : [ClientScope Class](#)

For a list of all members of this type, see [ClientScope members](#).

## Public Properties

	Name	Description
	<a href="#">ClientCache</a>	Gets the <a href="#">ClientCacheBase</a> to which this <a href="#">client scope</a> is connected.

[Top](#)

## See Also

### Reference

[ClientScope Class](#)  
[C1.Data Namespace](#)

## ClientCache Property

[C1.Data Namespace](#) > [ClientScope Class](#) : ClientCache Property

Gets the [ClientCacheBase](#) to which this [client scope](#) is connected.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property ClientCache As ClientCacheBase</code>	
C#	
<code>public ClientCacheBase ClientCache {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientScope Class](#)

[ClientScope Members](#)

## ClientView<T>

[C1.Data Namespace](#) : ClientView<T> Class

The type of the elements in the view.

Represents a *client view*, that is, a [live view](#) that is connected to a remote source, such as an **System.Data.Objects.ObjectContext**.

## Object Model

ClientView<T>

## Syntax

Visual Basic (Declaration)	
<code>Public Class ClientView(Of T)          Inherits C1.LiveLinq.LiveViews.View(Of T)          Implements C1.LiveLinq.Indexing.IIndexedSource(Of T),          C1.LiveLinq.IObservableSource(Of T)</code>	
C#	
<code>public class ClientView&lt;T&gt; : C1.LiveLinq.LiveViews.View&lt;T&gt;,          C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;, C1.LiveLinq.IObservableSource&lt;T&gt;</code>	

## Type Parameters

*T*

The type of the elements in the view.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

**C1.Data.ClientView<T>**

[C1.Data.FilteredView<T>](#)

[C1.Data.PagingView<T>](#)

[C1.Data.ProgressiveView<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Members](#)

[C1.Data Namespace](#)

### Overview

[C1.Data Namespace](#) : ClientView<T> Class

The type of the elements in the view.

Represents a *client view*, that is, a [live view](#) that is connected to a remote source, such as an **System.Data.Objects.ObjectContext**.

## Object Model

ClientView<T>

## Syntax

Visual Basic (Declaration)

```
Public Class ClientView(Of T)
```

Inherits <a href="#">C1.LiveLinq.LiveViews.View(Of T)</a> Implements <a href="#">C1.LiveLinq.Indexing.IIndexedSource(Of T)</a> , <a href="#">C1.LiveLinq.IObservableSource(Of T)</a>	
C#	
<pre>public class ClientView&lt;T&gt; : <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a>,  <a href="#">C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</a>, <a href="#">C1.LiveLinq.IObservableSource&lt;T&gt;</a></pre>	

## Type Parameters

*T*

The type of the elements in the view.

## Inheritance Hierarchy

```
System.Object
  C1.LiveLinq.LiveViews.View
    C1.LiveLinq.LiveViews.View<T>
      C1.Data.ClientView<T>
        C1.Data.FilteredView<T>
        C1.Data.PagingView<T>
        C1.Data.ProgressiveView<T>
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Members](#)  
[C1.Data Namespace](#)

### Members


[Properties](#) [Methods](#) [Events](#)




[C1.Data Namespace](#) : [ClientView<T>](#) Class

The following tables list the members exposed by [ClientView<T>](#).

## Public Properties



















	Name	Description
	<a href="#">AutoLoad</a>	Gets or sets a boolean value indicating whether the <a href="#">client view</a> must be loaded automatically when its data is accessed.
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">IsLoaded</a>	Gets a value indicating whether the <a href="#">client view</a> is <a href="#">loaded</a> .
	<a href="#">IsLoading</a>	Gets a value indicating whether the <a href="#">client view</a> is being <a href="#">loaded</a> .
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Scope</a>	Gets the <a href="#">client scope</a> to which this client view belongs.
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## Public Methods

	Name	Description
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">C1.LiveLinq.LiveViews.View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsFiltered</a>	Filters the view on the server side using a <a href="#">predicate</a> .
	<a href="#">AsFilteredBound&lt;TKey&gt;</a>	Filters the view on the server using a key selector function and configurable <a href="#">value</a> and <a href="#">operator</a> .
	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )



⇒  <a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">CancelLoad</a>	Cancels the current <a href="#">loading operation</a> .
⇒  <a href="#">Concat</a>	Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒  <a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">Include</a>	Specifies related objects to include while loading the <a href="#">client view</a> .
⇒  <a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">Join&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

≡	<a href="#">Load</a>	Loads the entities of the <a href="#">client view</a> .
≡	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">Paging</a>	Overloaded. Applies paging to this <a href="#">client view</a> .
≡	<a href="#">ProgressiveLoading</a>	Overloaded. Specifies that the <a href="#">client view</a> loading is performed not in a single trip to the server but in multiple batches, each loading a page of a limited size so the user sees the result and can interact with it before all pages are loaded.
≡	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">Refresh</a>	Loads the entities of the <a href="#">client view</a> ignoring the client-side cache.
≡	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">C1.LiveLinq.LiveViews.View.Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">Union</a>	Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">Loaded</a>	Occurs when the <a href="#">client view</a> has finished <a href="#">loading</a> succesfully, and also when an exception has been thrown during <a href="#">loading</a> .

[Top](#)

## See Also

### Reference

[ClientView<T> Class](#)  
[C1.Data Namespace](#)


### Methods

[C1.Data Namespace](#) : [ClientView<T> Class](#)

For a list of all members of this type, see [ClientView<T> members](#).

## Public Methods

	Name	Description
≡	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">C1.LiveLinq.LiveViews.View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">AsFiltered</a>	Filters the view on the server side using a <a href="#">predicate</a> .
≡	<a href="#">AsFilteredBound&lt;TKey&gt;</a>	Filters the view on the server using a key selector function and configurable <a href="#">value</a> and <a href="#">operator</a> .
≡	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">CancelLoad</a>	Cancels the current <a href="#">loading operation</a> .
≡	<a href="#">Concat</a>	Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

⇒ 	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">Include</a>	Specifies related objects to include while loading the <a href="#">client view</a> .
⇒ 	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">Join&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">Load</a>	Loads the entities of the <a href="#">client view</a> .
⇒ 	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ 	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">Paging</a>	Overloaded. Applies paging to this <a href="#">client view</a> .
⇒ 	<a href="#">ProgressiveLoading</a>	Overloaded. Specifies that the <a href="#">client view</a> loading is performed not in a single trip to the server but in multiple batches, each loading a page of a limited size so

		the user sees the result and can interact with it before all pages are loaded.
⇒💎	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒💎	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒💎	<a href="#">Refresh</a>	Loads the entities of the <a href="#">client view</a> ignoring the client-side cache.
⇒💎	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒💎	<a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒💎	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">C1.LiveLinq.LiveViews.View.Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒💎	<a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒💎	<a href="#">Union</a>	Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒💎	<a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference



[ClientView<T> Class](#)

[C1.Data Namespace](#)

## AsFiltered Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : AsFiltered Method

A function to apply each element to test the condition.

Filters the view on the server side using a [predicate](#).

## Syntax

Visual Basic (Declaration)

```
Public Overridable Function AsFiltered( _  
    ByVal predicate As System.Linq.Expressions.Expression(Of Func(Of  
T, Boolean)) _  
) As ClientView(Of T)
```

C#

```
public virtual ClientView<T> AsFiltered(  
    System.Linq.Expressions.Expression<Func<T, bool>> predicate  
)
```

## Parameters

*predicate*

A function to apply each element to test the condition.

## Return Value

A [client view](#) that contains elements of this view that satisfy the [predicate](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

## AsFilteredBound<TKey> Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : AsFilteredBound<TKey> Method

The type of the value used for filtering.

A function that returns a key value for filtering given a view item.

Filters the view on the server using a key selector function and configurable [value](#) and [operator](#).

## Syntax

Visual Basic (Declaration)

```
Public Overridable Function AsFilteredBound(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
) As FilteredView(Of T)
```

C#

```
public virtual FilteredView<T> AsFilteredBound<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector  
)
```

### Parameters

*keySelector*

A function that returns a key value for filtering given a view item.

### Type Parameters

*TKey*

The type of the value used for filtering.

### Return Value

A [FilteredView<T>](#) that contains elements of this view that have keys satisfying the condition.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

### CancelLoad Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : CancelLoad Method

Cancels the current [loading operation](#).

## Syntax

Visual Basic (Declaration)

```
Public Sub CancelLoad()
```

C#

```
public void CancelLoad()
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

### Include Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : Include Method

A dot-separated list of related objects to load along with the entities of this [client view](#).

Specifies related objects to include while loading the [client view](#).

## Syntax

Visual Basic (Declaration)

```
Public Overridable Function Include( _  
    ByVal path As System.String _  
) As ClientView(Of T)
```

C#

```
public virtual ClientView<T> Include(  
    System.string path  
)
```

## Parameters

*path*

A dot-separated list of related objects to load along with the entities of this [client view](#).

## Return Value

A [client view](#) that loads the related objects together with its entities every time it fetches entities from the server.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

### Load Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : Load() Method

Loads the entities of the [client view](#).

## Syntax

Visual Basic (Declaration)

```
Public Sub Load()
```

C#

```
public void Load()
```

## Remarks

If the entities are already in the cache, there will be no roundtrip to the server.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

### Paging Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : Paging Method

Applies paging to this [client view](#).

## Overload List

Overload	Description
<a href="#">Paging&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Int32)</a>	Applies paging to this <a href="#">client view</a> .
<a href="#">Paging&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Boolean,Int32)</a>	Applies paging to this <a href="#">client view</a> .

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

### Paging<TKey>(Expression<Func<T,TKey>>,Int32) Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) > [Paging Method](#) :

[Paging<TKey>\(Expression<Func<T,TKey>>,Int32\) Method](#)

The type of the sort key.

A function specifying a sort key.

A value for the [PageSize](#) property, the number of items to load in a page.

Applies paging to this [client view](#).

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Paging(Of TKey)( _  
    ByVal sortKeySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)), _  
    ByVal pageSize As System.Integer _  
) As PagingView(Of T)
```

C#

```
public PagingView<T> Paging<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> sortKeySelector,  
    System.int pageSize  
)
```

### Parameters

*sortKeySelector*

A function specifying a sort key.

*pageSize*

A value for the [PageSize](#) property, the number of items to load in a page.

### Type Parameters

*TKey*

The type of the sort key.

### Return Value

A [paged client view](#).

## Remarks

Sorting is required, paging entities is impossible without sort.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)  
[Overload List](#)

Paging<TKey>(Expression<Func<T,TKey>>,Boolean,Int32) Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) > [Paging Method](#) :

Paging<TKey>(Expression<Func<T,TKey>>,Boolean,Int32) Method

The type of the sort key.

A function specifying a sort key.

A boolean value indicating whether sorting must be performed in the ascending order (descending, if false).

A value for the [PageSize](#) property, the number of items to load in a page.

Applies paging to this [client view](#).

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Paging(Of TKey)( _  
    ByVal sortKeySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)), _  
    ByVal ascending As System.Boolean, _  
    ByVal pageSize As System.Integer _  
) As PagingView(Of T)
```

C#

```
public PagingView<T> Paging<TKey>(  
    System.Linq.Expressions.Expression<Func<T,TKey>> sortKeySelector,  
    System.bool ascending,  
    System.int pageSize  
)
```

### Parameters

*sortKeySelector*

A function specifying a sort key.

*ascending*

A boolean value indicating whether sorting must be performed in the ascending order (descending, if false).

*pageSize*

A value for the [PageSize](#) property, the number of items to load in a page.

## Type Parameters

*TKey*

The type of the sort key.

## Return Value

A [paged client view](#).

## Remarks

Sorting is required, paging entities is impossible without sort.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)  
[Overload List](#)

## ProgressiveLoading Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : ProgressiveLoading Method

Specifies that the [client view](#) loading is performed not in a single trip to the server but in multiple batches, each loading a page of a limited size so the user sees the result and can interact with it before all pages are loaded.

## Overload List



Overload	Description
<a href="#">ProgressiveLoading&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Int32)</a>	Specifies that the <a href="#">client view</a> loading is performed not in a single trip to the server but in multiple batches, each loading a page of a limited size so the user sees the result and can interact with it before all pages are loaded.
<a href="#">ProgressiveLoading&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Boolean,Int32)</a>	Specifies that the <a href="#">client view</a> loading is performed not in a single trip to the server but in multiple batches, each loading a page of a limited size so the user sees the result and can interact with it before all pages are loaded.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

[ProgressiveLoading<TKey>\(Expression<Func<T,TKey>>,Int32\) Method](#)

[C1.Data Namespace](#) > [ClientView<T> Class](#) > [ProgressiveLoading Method](#) :

ProgressiveLoading<TKey>(Expression<Func<T,TKey>>,Int32) Method

The type of the sort key.

A function specifying a sort key.

The size of the page.

Specifies that the [client view](#) loading is performed not in a single trip to the server but in multiple batches, each loading a page of a limited size so the user sees the result and can interact with it before all pages are loaded.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function ProgressiveLoading(Of TKey)( _  
    ByVal sortKeySelector As System.Linq.Expressions.Expression(Of Func(Of  
T,TKey)), _  
    ByVal LoadSize As System.Integer _  
) As ProgressiveView(Of T)
```

C#

```
public ProgressiveView<T> ProgressiveLoading<TKey>(  
    System.Linq.Expressions.Expression<Func<T,TKey>> sortKeySelector,  
    System.int LoadSize  
)
```

### Parameters

*sortKeySelector*

A function specifying a sort key.

*loadSize*

The size of the page.

### Type Parameters

*TKey*

The type of the sort key.

### Return Value

A [client view](#) that loads the same entities as the source view but does it progressively.

## Remarks

Sorting is required, loading entities progressively is impossible without sort.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)  
[Overload List](#)

[ProgressiveLoading<TKey>\(Expression<Func<T,TKey>>,Boolean,Int32\) Method](#)

[C1.Data Namespace](#) > [ClientView<T> Class](#) > [ProgressiveLoading Method](#) :

[ProgressiveLoading<TKey>\(Expression<Func<T,TKey>>,Boolean,Int32\) Method](#)

The type of the sort key.

A function specifying a sort key.

A boolean value indicating whether sorting must be performed in the ascending order (descending, if false).

The size of the page.

Specifies that the [client view](#) loading is performed not in a single trip to the server but in multiple batches, each loading a page of a limited size so the user sees the result and can interact with it before all pages are loaded.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function ProgressiveLoading(Of TKey)( _  
    ByVal sortKeySelector As System.Linq.Expressions.Expression(Of Func(Of  
T,TKey))), _  
    ByVal ascending As System.Boolean, _  
    ByVal loadSize As System.Integer _  
) As ProgressiveView(Of T)
```

C#

```
public ProgressiveView<T> ProgressiveLoading<TKey>(
    System.Linq.Expressions.Expression<Func<T,TKey>> sortKeySelector,
    System.bool ascending,
    System.int loadSize
)
```

## Parameters

*sortKeySelector*

A function specifying a sort key.

*ascending*

A boolean value indicating whether sorting must be performed in the ascending order (descending, if false).

*loadSize*

The size of the page.

## Type Parameters

*TKey*

The type of the sort key.

## Return Value

A [client view](#) that loads the same entities as the source view but does it progressively.

## Remarks

Sorting is required, loading entities progressively is impossible without sort.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)  
[Overload List](#)

## Refresh Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : Refresh Method

Loads the entities of the [client view](#) ignoring the client-side cache.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Refresh()	
C#	
<b>public void</b> Refresh()	

## Remarks

Use this method to refresh data with any changes that may have occurred on the server

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference


[ClientView<T> Class](#)  
[ClientView<T> Members](#)










## Properties



[C1.Data Namespace](#) : [ClientView<T> Class](#)

For a list of all members of this type, see [ClientView<T> members](#).

## Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	Gets or sets a boolean value indicating whether the <a href="#">client view</a> must

		be loaded automatically when its data is accessed.
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">IsLoaded</a>	Gets a value indicating whether the <a href="#">client view</a> is <a href="#">loaded</a> .
	<a href="#">IsLoading</a>	Gets a value indicating whether the <a href="#">client view</a> is being <a href="#">loaded</a> .
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from

		<a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Scope</a>	Gets the <a href="#">client scope</a> to which this client view belongs.
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## See Also

### Reference

[ClientView<T> Class](#)

[C1.Data Namespace](#)

### AutoLoad Property

[C1.Data Namespace](#) > [ClientView<T> Class](#) : AutoLoad Property

Gets or sets a boolean value indicating whether the [client view](#) must be loaded automatically when its data is accessed.

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> AutoLoad <b>As</b> System.Boolean	
C#	
<b>public</b> System. <b>bool</b> AutoLoad { <b>get</b> ; <b>set</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

## IsLoaded Property

[C1.Data Namespace](#) > [ClientView<T> Class](#) : IsLoaded Property

Gets a value indicating whether the [client view](#) is [loaded](#).

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property IsLoaded As System.Boolean</code>	
C#	
<code>public System.bool IsLoaded {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

## IsLoading Property

[C1.Data Namespace](#) > [ClientView<T> Class](#) : IsLoading Property

Gets a value indicating whether the [client view](#) is being [loaded](#).

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property IsLoading As System.Boolean</code>	
C#	
<code>public System.bool IsLoading {get;}</code>	

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)

Scope Property

[C1.Data Namespace](#) > [ClientView<T> Class](#) : Scope Property

Gets the [client scope](#) to which this client view belongs.

Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Scope As ClientScope</code>	
C#	
<code>public ClientScope Scope {get;}</code>	

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2


See Also

Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)

Events

>

Name	Description
 <a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )



Occurs when the [client view](#) has finished [loading](#) successfully, and also when an exception has been thrown during [loading](#).

[Top](#)

## See Also

### Reference

[ClientView<T> Class](#)

[C1.Data Namespace](#)

### Loaded Event

[C1.Data Namespace](#) > [ClientView<T> Class](#) : Loaded Event

Occurs when the [client view](#) has finished [loading](#) successfully, and also when an exception has been thrown during [loading](#).

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> Loaded <b>As</b> System.EventHandler(Of ClientViewLoadedEventArgs)	
C#	
<b>public event</b> System.EventHandler<ClientViewLoadedEventArgs> Loaded	

## Event Data

The event handler receives an argument of type [ClientViewLoadedEventArgs](#) containing data related to this event. The following **ClientViewLoadedEventArgs** properties provide information specific to this event.

Property	Description
<a href="#">Error</a>	Gets the loading error if the loading failed.
<a href="#">HasError</a>	Gets a value indicating whether the loading has failed. If true, inspect the <a href="#">Error</a> property for details.
<a href="#">IsErrorHandled</a>	Gets a value indicating whether the loading error has been marked as handled by calling <a href="#">MarkErrorAsHandled</a> .

<a href="#">Items</a>	Gets all entities loaded by a <a href="#">client view</a> .
<a href="#">TotalItemCount</a>	Gets the total number of rows for the original query without any paging applied to it.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

## ClientViewLoadedEventArgs

[C1.Data Namespace](#) : ClientViewLoadedEventArgs Class

Provides data for the [ClientView<T>.Loaded](#) event.

## Object Model

ClientViewLoadedEventArgs

## Syntax

Visual Basic (Declaration)

```
Public Class ClientViewLoadedEventArgs
    Inherits System.EventArgs
```

C#

```
public class ClientViewLoadedEventArgs : System.EventArgs
```

## Inheritance Hierarchy

System.Object

System.EventArgs

**C1.Data.ClientViewLoadedEventArgs**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Members](#)  
[C1.Data Namespace](#)

## Overview

[C1.Data Namespace](#) : ClientViewLoadedEventArgs Class

Provides data for the [ClientView<T>.Loaded](#) event.

## Object Model

ClientViewLoadedEventArgs

## Syntax

Visual Basic (Declaration)	
<pre>Public Class ClientViewLoadedEventArgs     Inherits System.EventArgs</pre>	
C#	
<pre>public class ClientViewLoadedEventArgs : System.EventArgs</pre>	

## Inheritance Hierarchy

System.Object  
    System.EventArgs  
        **C1.Data.ClientViewLoadedEventArgs**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

## [ClientViewLoadedEventArgs Members](#)

### [C1.Data Namespace](#)






## Members

[Properties](#) [Methods](#)

[C1.Data Namespace](#) : [ClientViewLoadedEventArgs](#) Class


The following tables list the members exposed by [ClientViewLoadedEventArgs](#).

## Public Properties

	Name	Description
	<a href="#">Error</a>	Gets the loading error if the loading failed.
	<a href="#">HasError</a>	Gets a value indicating whether the loading has failed. If true, inspect the <a href="#">Error</a> property for details.
	<a href="#">IsErrorHandled</a>	Gets a value indicating whether the loading error has been marked as handled by calling <a href="#">MarkErrorAsHandled</a> .
	<a href="#">Items</a>	Gets all entities loaded by a <a href="#">client view</a> .
	<a href="#">TotalItemCount</a>	Gets the total number of rows for the original query without any paging applied to it.

[Top](#)

## Public Methods

	Name	Description
	<a href="#">MarkErrorAsHandled</a>	For the case where <a href="#">HasError</a> is true, this method marks the error as handled. If this method is not called, an exception will be thrown.

[Top](#)

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)


[C1.Data Namespace](#)

## Methods

[C1.Data Namespace](#) : ClientViewLoadedEventArgs Class

For a list of all members of this type, see [ClientViewLoadedEventArgs members](#).

## Public Methods

	Name	Description
	<a href="#">MarkErrorAsHandled</a>	For the case where <a href="#">HasError</a> is true, this method marks the error as handled. If this method is not called, an exception will be thrown.

[Top](#)

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)

[C1.Data Namespace](#)

### MarkErrorAsHandled Method

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : MarkErrorAsHandled Method

For the case where [HasError](#) is true, this method marks the error as handled. If this method is not called, an exception will be thrown.

## Syntax

Visual Basic (Declaration)	
<code>Public Sub MarkErrorAsHandled()</code>	
C#	
<code>public void MarkErrorAsHandled()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also






### Reference

## Properties

[C1.Data Namespace](#) : [ClientViewLoadedEventArgs Class](#)

For a list of all members of this type, see [ClientViewLoadedEventArgs members](#).

## Public Properties

	Name	Description
	<a href="#">Error</a>	Gets the loading error if the loading failed.
	<a href="#">HasError</a>	Gets a value indicating whether the loading has failed. If true, inspect the <a href="#">Error</a> property for details.
	<a href="#">IsErrorHandled</a>	Gets a value indicating whether the loading error has been marked as handled by calling <a href="#">MarkErrorAsHandled</a> .
	<a href="#">Items</a>	Gets all entities loaded by a <a href="#">client view</a> .
	<a href="#">TotalItemCount</a>	Gets the total number of rows for the original query without any paging applied to it.

[Top](#)

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)  
[C1.Data Namespace](#)

### Error Property

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : Error Property

Gets the loading error if the loading failed.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property Error As System.Exception
```

C#

```
public System.Exception Error {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)

[ClientViewLoadedEventArgs Members](#)

### HasError Property

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : HasError Property

Gets a value indicating whether the loading has failed. If true, inspect the [Error](#) property for details.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property HasError As System.Boolean
```

C#

```
public System.bool HasError {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)

[ClientViewLoadedEventArgs Members](#)



## IsErrorHandled Property

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : IsErrorHandled Property

Gets a value indicating whether the loading error has been marked as handled by calling [MarkErrorAsHandled](#).

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> IsErrorHandled <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool IsErrorHandled { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)

[ClientViewLoadedEventArgs Members](#)

## Items Property

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : Items Property

Gets all entities loaded by a [client view](#).

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> Items <b>As</b> System.Collections.IEnumerable	
C#	
<b>public</b> System.Collections.IEnumerable Items { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)  
[ClientViewLoadedEventArgs Members](#)

### TotalItemCount Property

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : TotalItemCount Property

Gets the total number of rows for the original query without any paging applied to it.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> TotalItemCount <b>As</b> System.Integer	
C#	
<b>public</b> System.int TotalItemCount { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)  
[ClientViewLoadedEventArgs Members](#)

## DataExtensions

[C1.Data Namespace](#) : DataExtensions Class

Extension methods provided by Studio for Entity Framework.

## Object Model

DataExtensions

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public MustInherit NotInheritable Class DataExtensions</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static class DataExtensions</pre>	

## Inheritance Hierarchy

System.Object

**C1.Data.DataExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DataExtensions Members](#)

[C1.Data Namespace](#)

## Overview

[C1.Data Namespace](#) : DataExtensions Class

Extension methods provided by Studio for Entity Framework.

## Object Model

DataExtensions

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public MustInherit NotInheritable Class DataExtensions</pre>	

C#	
[System.Runtime.CompilerServices.Extension()] public static class DataExtensions	

## Inheritance Hierarchy

System.Object  
**C1.Data.DataExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DataExtensions Members](#)  
[C1.Data Namespace](#)


## Members

[Methods](#)

[C1.Data Namespace](#) : DataExtensions Class

The following tables list the members exposed by [DataExtensions](#).

## Public Methods

	Name	Description
	<a href="#">ExecuteIn&lt;T&gt;</a>	Executes a <a href="#">query</a> in a <a href="#">scope</a> , so the loaded entities are pinned to the <a href="#">scope</a> (marked as needed) by calling <a href="#">ClientScope.AddRef</a> for each of them.

[Top](#)

## See Also

### Reference


[DataExtensions Class](#)  
[C1.Data Namespace](#)

# Methods

[C1.Data Namespace](#) : DataExtensions Class

For a list of all members of this type, see [DataExtensions members](#).

## Public Methods

	Name	Description
	<a href="#">ExecuteIn&lt;T&gt;</a>	Executes a <a href="#">query</a> in a <a href="#">scope</a> , so the loaded entities are pinned to the <a href="#">scope</a> (marked as needed) by calling <a href="#">ClientScope.AddRef</a> for each of them.

[Top](#)

## See Also

### Reference

[DataExtensions Class](#)

[C1.Data Namespace](#)

### ExecuteIn<T> Method

[C1.Data Namespace](#) > [DataExtensions Class](#) : ExecuteIn<T> Method

The type of items returned by the query.

The query to execute inside the [client scope](#).

The [client scope](#) to execute the query in.

Executes a [query](#) in a [scope](#), so the loaded entities are pinned to the [scope](#) (marked as needed) by calling [ClientScope.AddRef](#) for each of them.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Function ExecuteIn(Of T)( _     ByVal query As System.Collections.Generic.IEnumerable(Of T), _     ByVal scope As ClientScope _ ) As System.Collections.Generic.IEnumerable(Of T)</pre>	
C#	

```
[System.Runtime.CompilerServices.Extension()]
public static System.Collections.Generic.IEnumerable<T> ExecuteIn<T>(
    System.Collections.Generic.IEnumerable<T> query,
    ClientScope scope
)
```

## Parameters

*query*

The query to execute inside the [client scope](#).

*scope*

The [client scope](#) to execute the query in.

## Type Parameters

*T*

The type of items returned by the query.

## Return Value

The query that will be executed inside the given [scope](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[DataExtensions Class](#)

[DataExtensions Members](#)

## FilteredView<T>

[C1.Data Namespace](#) : FilteredView<T> Class

The type of the entities in this client view.

Represents a [client view](#) filtered by a filter key function, an [operator](#) and a [filter key value](#).

## Object Model

FilteredView<T>

## Syntax

Visual Basic (Declaration)

```
Public Class FilteredView(Of T)
    Inherits ClientView(Of T)
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T),
C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class FilteredView<T> : ClientView<T>,
C1.LiveLinq.Indexing.IIndexedSource<T>, C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

The type of the entities in this client view.

## Remarks

Filtering is performed on the server.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

[C1.Data.ClientView<T>](#)

**C1.Data.FilteredView<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilteredView<T> Members](#)

[C1.Data Namespace](#)

## Overview

[C1.Data Namespace](#) : [FilteredView<T>](#) Class

The type of the entities in this client view.

Represents a [client view](#) filtered by a filter key function, an [operator](#) and a [filter key value](#).

## Object Model

[FilteredView<T>](#)

## Syntax

Visual Basic (Declaration)

```
Public Class FilteredView(Of T)
    Inherits ClientView(Of T)
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T),
C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class FilteredView<T> : ClientView<T>,
C1.LiveLinq.Indexing.IIndexedSource<T>, C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

The type of the entities in this client view.

## Remarks

Filtering is performed on the server.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

[C1.Data.ClientView<T>](#)

**[C1.Data.FilteredView<T>](#)**

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilteredView<T> Members](#)  
[C1.Data Namespace](#)


## Members

[Fields](#) [Properties](#) [Methods](#) [Events](#)

[C1.Data Namespace](#) : [FilteredView<T>](#) Class





The following tables list the members exposed by [FilteredView<T>](#).












## Public Fields




	Name	Description
 <b>S</b>	<a href="#">Unfiltered</a>	A special value indicating that filtering must not be performed. To disable filtering, assign the value of this field to the <a href="#">FilterKey</a> property.

[Top](#)

## Public Properties





	Name	Description
	<a href="#">AutoLoad</a>	Gets or sets a boolean value indicating whether the <a href="#">client view</a> must be loaded automatically when its data is accessed. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )













		<a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">FilterKey</a>	Gets or sets a value that is used to filter items by comparing this value to the result of the filter key function applied to an item.
	<a href="#">FilterKeyType</a>	Gets the filter key type. It is determined by the filter key function.
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">IsLoaded</a>	Gets a value indicating whether the <a href="#">client view</a> is <a href="#">loaded</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsLoading</a>	Gets a value indicating whether the <a href="#">client view</a> is being <a href="#">loaded</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Operator</a>	Gets a <a href="#">C1.Data.DataSource.FilterOperator</a> used to compare filter keys.

	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Scope</a>	Gets the <a href="#">client scope</a> to which this client view belongs. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )







[Top](#)

## Public Methods

	Name	Description
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">C1.LiveLinq.LiveViews.View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsFiltered</a>	Filters the view on the server side using a <a href="#">predicate</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">AsFilteredBound&lt;TKey&gt;</a>	Filters the view on the server using a key selector function and configurable <a href="#">value</a> and <a href="#">operator</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )



⇒ 	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">BindFilterKey</a>	Overloaded. Binds the <a href="#">FilterKey</a> property using the specified <b>System.Windows.Data.Binding</b> object.
⇒ 	<a href="#">CancelLoad</a>	Cancels the current <a href="#">loading operation</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ 	<a href="#">Concat</a>	Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ 	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">Include</a>	Specifies related objects to include while loading the <a href="#">client view</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ 	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of

		its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Join&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Load</a>	Loads the entities of the <a href="#">client view</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Paging</a>	Overloaded. Applies paging to this <a href="#">client view</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒	<a href="#">ProgressiveLoading</a>	Overloaded. Specifies that the <a href="#">client view</a> loading is performed not in a single trip to the server but in multiple batches, each loading a page of a limited size so the user sees the result and can interact with it before all pages are loaded. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Refresh</a>	Loads the entities of the <a href="#">client view</a> ignoring the client-side cache. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )

 <a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">SetTransaction</a>	Sets the value of the <a href="#">C1.LiveLinq.LiveViews.View.Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">Union</a>	Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## Public Events

	Name	Description
 <a href="#">Changed</a>		Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">Loaded</a>		Occurs when the <a href="#">client view</a> has finished <a href="#">loading</a> successfully, and also when an exception has been thrown during <a href="#">loading</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[FilteredView<T> Class](#)

[C1.Data Namespace](#)







## Methods

[C1.Data Namespace](#) : [FilteredView<T> Class](#)

For a list of all members of this type, see [FilteredView<T> members](#).

## Public Methods

	Name	Description
≡	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">C1.LiveLinq.LiveViews.View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">AsFiltered</a>	Filters the view on the server side using a <a href="#">predicate</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
≡	<a href="#">AsFilteredBound&lt;TKey&gt;</a>	Filters the view on the server using a key selector function and configurable <a href="#">value</a> and <a href="#">operator</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
≡	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">BindFilterKey</a>	Overloaded. Binds the <a href="#">FilterKey</a> property using the specified <b>System.Windows.Data.Binding</b> object.

⇒  CancelLoad	Cancels the current <a href="#">loading operation</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒  Concat	Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  Contains	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  DeferMaintenance	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒  GetEnumerator	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  GroupBy	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  GroupJoin<TInner,TKey,TResult>	Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  Include	Specifies related objects to include while loading the <a href="#">client view</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒  IndexOf	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  Join<TInner,TKey,TResult>	Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  Load	Loads the entities of the <a href="#">client view</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )



⇒	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Paging</a>	Overloaded. Applies paging to this <a href="#">client view</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒	<a href="#">ProgressiveLoading</a>	Overloaded. Specifies that the <a href="#">client view</a> loading is performed not in a single trip to the server but in multiple batches, each loading a page of a limited size so the user sees the result and can interact with it before all pages are loaded. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Refresh</a>	Loads the entities of the <a href="#">client view</a> ignoring the client-side cache. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

≡	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">C1.LiveLinq.LiveViews.View.Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">Union</a>	Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[FilteredView<T> Class](#)

[C1.Data Namespace](#)

### BindFilterKey Method

[C1.Data Namespace](#) > [FilteredView<T> Class](#) : BindFilterKey Method

Binds the [FilterKey](#) property using the specified **System.Windows.Data.Binding** object.

## Overload List

Overload	Description
<a href="#">BindFilterKey(Binding)</a>	Binds the <a href="#">FilterKey</a> property using the specified <b>System.Windows.Data.Binding</b> object.
<a href="#">BindFilterKey(Object,String)</a>	Binds the <a href="#">FilterKey</a> property to a given <a href="#">path</a> on a given <a href="#">source</a> .

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[FilteredView<T> Class](#)

[FilteredView<T> Members](#)

BindFilterKey(Binding) Method

[C1.Data Namespace](#) > [FilteredView<T> Class](#) > [BindFilterKey Method](#) : BindFilterKey(Binding) Method

The **System.Windows.Data.Binding** object used to bind the [FilterKey](#). Use null to unbind the previously bound [FilterKey](#).

Binds the [FilterKey](#) property using the specified **System.Windows.Data.Binding** object.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function BindFilterKey( _  
    ByVal binding As System.Windows.Data.Binding _  
) As FilteredView(Of T)
```

C#

```
public FilteredView<T> BindFilterKey(  
    System.Windows.Data.Binding binding  
)
```

### Parameters

*binding*

The **System.Windows.Data.Binding** object used to bind the [FilterKey](#). Use null to unbind the previously bound [FilterKey](#).

### Return Value

The same [FilteredView<T>](#) on which this method was called.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilteredView<T> Class](#)  
[FilteredView<T> Members](#)  
[Overload List](#)

## BindFilterKey(Object,String) Method

[C1.Data Namespace](#) > [FilteredView<T> Class](#) > [BindFilterKey Method](#) : BindFilterKey(Object,String)  
 Method

The object to bind to. Cannot be null.

The property path to bind to.

Binds the [FilterKey](#) property to a given [path](#) on a given [source](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function BindFilterKey( _     ByVal source As System.Object, _     ByVal path As System.String _ ) As FilteredView(Of T)</pre>	
C#	
<pre>public FilteredView&lt;T&gt; BindFilterKey(     System.object source,     System.string path )</pre>	

## Parameters

*source*

The object to bind to. Cannot be null.

*path*

The property path to bind to.

## Return Value

The same [FilteredView<T>](#) on which this method was called.

## Exceptions

Exception	Description

<b>System.NullReferenceException</b>	source is null.
--------------------------------------	-----------------

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference






[FilteredView<T> Class](#)  
[FilteredView<T> Members](#)  
[Overload List](#)







## Properties


[C1.Data Namespace](#) : [FilteredView<T> Class](#)

For a list of all members of this type, see [FilteredView<T> members](#).

## Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	Gets or sets a boolean value indicating whether the <a href="#">client view</a> must be loaded automatically when its data is accessed. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

	<a href="#">FilterKey</a>	Gets or sets a value that is used to filter items by comparing this value to the result of the filter key function applied to an item.
	<a href="#">FilterKeyType</a>	Gets the filter key type. It is determined by the filter key function.
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">IsLoaded</a>	Gets a value indicating whether the <a href="#">client view</a> is <a href="#">loaded</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsLoading</a>	Gets a value indicating whether the <a href="#">client view</a> is being <a href="#">loaded</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Operator</a>	Gets a <a href="#">C1.Data.DataSource.FilterOperator</a> used to compare filter keys.
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Scope</a>	Gets the <a href="#">client scope</a> to which this client view belongs. (Inherited

		from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## See Also

### Reference

[FilteredView<T> Class](#)

[C1.Data Namespace](#)

### FilterKey Property

[C1.Data Namespace](#) > [FilteredView<T> Class](#) : FilterKey Property

Gets or sets a value that is used to filter items by comparing this value to the result of the filter key function applied to an item.

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> FilterKey <b>As</b> System.Object	
C#	
<b>public</b> System.object FilterKey { <b>get</b> ; <b>set</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilteredView<T> Class](#)

[FilteredView<T> Members](#)

[Operator Property](#)

## FilterKeyType Property

[C1.Data Namespace](#) > [FilteredView<T> Class](#) : FilterKeyType Property

Gets the filter key type. It is determined by the filter key function.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> FilterKeyType <b>As</b> System.Type	
C#	
<b>public</b> System.Type FilterKeyType { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilteredView<T> Class](#)

[FilteredView<T> Members](#)

## Operator Property

[C1.Data Namespace](#) > [FilteredView<T> Class](#) : Operator Property

Gets a [C1.Data.DataSource.FilterOperator](#) used to compare filter keys.

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> Operator <b>As</b> FilterOperator	
C#	
<b>public</b> FilterOperator Operator { <b>get</b> ; <b>set</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2




# See Also

## Reference

- [FilteredView<T> Class](#)
- [FilteredView<T> Members](#)
- [FilterOperator Enumeration](#)

## Fields

>

Name	Description
 <b>Unfiltered</b>	A special value indicating that filtering must not be performed. To disable filtering, assign the value of this field to the <a href="#">FilterKey</a> property.

[Top](#)

# See Also

## Reference

- [FilteredView<T> Class](#)
- [C1.Data Namespace](#)

## Unfiltered Field

[C1.Data Namespace](#) > [FilteredView<T> Class](#) : Unfiltered Field

A special value indicating that filtering must not be performed. To disable filtering, assign the value of this field to the [FilterKey](#) property.

# Syntax

Visual Basic (Declaration)	
<code>Public Shared ReadOnly Unfiltered As System.Object</code>	
C#	
<code>public static readonly System.object Unfiltered</code>	

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilteredView<T> Class](#)

[FilteredView<T> Members](#)

## PageChangingEventArgs

[C1.Data Namespace](#) : PageChangingEventArgs Class

Provides data for the [C1.Data.DataSource.ClientCollectionView.PageChanging](#) event.

## Object Model

PageChangingEventArgs

## Syntax

Visual Basic (Declaration)

```
Public NotInheritable Class PageChangingEventArgs  
    Inherits System.ComponentModel.CancelEventArgs
```

C#

```
public sealed class PageChangingEventArgs :  
    System.ComponentModel.CancelEventArgs
```

## Inheritance Hierarchy

```
System.Object  
    System.EventArgs  
        System.ComponentModel.CancelEventArgs  
            C1.Data.PageChangingEventArgs
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

## Overview

[C1.Data Namespace](#) : PageChangingEventArgs Class

Provides data for the [C1.Data.DataSource.ClientCollectionView.PageChanging](#) event.

## Object Model

PageChangingEventArgs

## Syntax

Visual Basic (Declaration)	
<pre>Public NotInheritable Class PageChangingEventArgs     Inherits System.ComponentModel.CancelEventArgs</pre>	
C#	
<pre>public sealed class PageChangingEventArgs :     System.ComponentModel.CancelEventArgs</pre>	

## Inheritance Hierarchy

System.Object  
    System.EventArgs  
        System.ComponentModel.CancelEventArgs  
            **C1.Data.PageChangingEventArgs**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

## Members

[Properties](#)

[C1.Data Namespace](#) : [PageChangingEventArgs Class](#)



The following tables list the members exposed by [PageChangingEventArgs](#).

## Public Constructors

	Name	Description
	<a href="#">PageChangingEventArgs Constructor</a>	Initializes a new instance of the <a href="#">PageChangingEventArgs</a> class.

[Top](#)

## Public Properties

	Name	Description
	<a href="#">Cancel</a>	(Inherited from <a href="#">System.ComponentModel.CancelEventArgs</a> )
	<a href="#">NewPageIndex</a>	Gets the index of the requested page.

[Top](#)

## See Also

### Reference

[PageChangingEventArgs Class](#)  
[C1.Data Namespace](#)

## PageChangingEventArgs Constructor

[C1.Data Namespace](#) > [PageChangingEventArgs Class](#) : [PageChangingEventArgs Constructor](#)

The index of the requested page.

Initializes a new instance of the [PageChangingEventArgs](#) class.

## Syntax

Visual Basic (Declaration)

```
Public Function New( _  
    ByVal newPageIndex As System.Integer _  
)
```

C#

```
public PageChangingEventArgs(  
    System.int newPageIndex  
)
```

## Parameters

*newPageIndex*

The index of the requested page.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



### Reference

[PageChangingEventArgs Class](#)

[PageChangingEventArgs Members](#)

## Properties

>

Name	Description
 <a href="#">Cancel</a>	(Inherited from System.ComponentModel.CancelEventArgs)
 <a href="#">NewPageIndex</a>	Gets the index of the requested page.

[Top](#)

## See Also

### Reference

[PageChangingEventArgs Class](#)

[C1.Data Namespace](#)

### NewPageIndex Property

[C1.Data Namespace](#) > [PageChangingEventArgs Class](#) : NewPageIndex Property

Gets the index of the requested page.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property NewPageIndex As System.Integer
```

C#

```
public System.int NewPageIndex {get;}
```

### Property Value

The index of the requested page.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PageChangingEventArgs Class](#)

[PageChangingEventArgs Members](#)

## PagingView<T>

[C1.Data Namespace](#) : PagingView<T> Class

The type of the entities in the [client view](#).

Represents a paged [client view](#).

## Object Model

PagingView<T>

## Syntax

Visual Basic (Declaration)

```
Public Class PagingView(Of T)  
    Inherits ClientView(Of T)  
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T),  
    C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class PagingView<T> : ClientView<T>,  
C1.LiveLinq.Indexing.IIndexedSource<T>, C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

The type of the entities in the [client view](#).

## Remarks

Paging is performed on the server. It is controlled by the [PageSize](#) and [PageIndex](#) properties.

Sorting is required, paging entities is impossible without sort.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

[C1.Data.ClientView<T>](#)

**C1.Data.PagingView<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PagingView<T> Members](#)

[C1.Data Namespace](#)

### Overview

[C1.Data Namespace](#) : [PagingView<T>](#) Class

The type of the entities in the [client view](#).

Represents a paged [client view](#).

## Object Model

## Syntax

Visual Basic (Declaration)

```
Public Class PagingView(Of T)
    Inherits ClientView(Of T)
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T),
C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class PagingView<T> : ClientView<T>,
C1.LiveLinq.Indexing.IIndexedSource<T>, C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

The type of the entities in the [client view](#).

## Remarks

Paging is performed on the server. It is controlled by the [PageSize](#) and [PageIndex](#) properties.

Sorting is required, paging entities is impossible without sort.

## Inheritance Hierarchy

```
System.Object
    C1.LiveLinq.LiveViews.View
        C1.LiveLinq.LiveViews.View<T>
            C1.Data.ClientView<T>
                C1.Data.PagingView<T>
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



## Members












[Properties](#) [Methods](#) [Events](#)

[C1.Data Namespace](#) : [PagingView<T>](#) Class

The following tables list the members exposed by [PagingView<T>](#).

### Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	Gets or sets a boolean value indicating whether the <a href="#">client view</a> must be loaded automatically when its data is accessed. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">IsLoaded</a>	Gets a value indicating whether the <a href="#">client view</a> is <a href="#">loaded</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsLoading</a>	Gets a value indicating whether the <a href="#">client view</a> is being <a href="#">loaded</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not

		updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">LoadSize</a>	Gets or sets a value controlling the number of entities to load in one batch.
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">PageCount</a>	Gets the number of pages in this <a href="#">view</a> .
	<a href="#">PageIndex</a>	Gets or sets the index of the current page.
	<a href="#">PageSize</a>	Gets or sets the number of items in a page.
	<a href="#">Scope</a>	Gets the <a href="#">client scope</a> to which this client view belongs. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">TotalItemCount</a>	Gets the total number of entities in the view before paging is applied.
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited

		from <a href="#">C1.LiveLinq.LiveViews.View</a> )
--	--	---



[Top](#)

## Public Methods

	Name	Description
≡	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">C1.LiveLinq.LiveViews.View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">AsFiltered</a>	Filters the view on the server side using a <a href="#">predicate</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
≡	<a href="#">AsFilteredBound&lt;TKey&gt;</a>	Filters the view on the server using a key selector function and configurable <a href="#">value</a> and <a href="#">operator</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
≡	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">CancelLoad</a>	Cancels the current <a href="#">loading operation</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
≡	<a href="#">Concat</a>	Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )



≡💎	<a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡💎	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡💎	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡💎	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡💎	<a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡💎	<a href="#">Include</a>	Specifies related objects to include while loading the <a href="#">client view</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
≡💎	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡💎	<a href="#">Join&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡💎	<a href="#">Load</a>	Loads the entities of the <a href="#">client view</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
≡💎	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡💎	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

⇒	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Paging</a>	Overloaded. Applies paging to this <a href="#">client view</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒	<a href="#">ProgressiveLoading</a>	Overloaded. Specifies that the <a href="#">client view</a> loading is performed not in a single trip to the server but in multiple batches, each loading a page of a limited size so the user sees the result and can interact with it before all pages are loaded. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Refresh</a>	Loads the entities of the <a href="#">client view</a> ignoring the client-side cache. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">C1.LiveLinq.LiveViews.View.Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

	<a href="#">Union</a>	Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">Loaded</a>	Occurs when the <a href="#">client view</a> has finished <a href="#">loading</a> succesfully, and also when an exception has been thrown during <a href="#">loading</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[PagingView<T> Class](#)


[C1.Data Namespace](#)










## Properties








[C1.Data Namespace](#) : [PagingView<T> Class](#)

For a list of all members of this type, see [PagingView<T> members](#).

## Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	Gets or sets a boolean value indicating whether the <a href="#">client view</a> must be loaded automatically when its data is accessed. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )

	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">IsLoaded</a>	Gets a value indicating whether the <a href="#">client view</a> is <a href="#">loaded</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsLoading</a>	Gets a value indicating whether the <a href="#">client view</a> is being <a href="#">loaded</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">LoadSize</a>	Gets or sets a value controlling the number of entities to load in one batch.
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from

		<a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">PageCount</a>	Gets the number of pages in this <a href="#">view</a> .
	<a href="#">PageIndex</a>	Gets or sets the index of the current page.
	<a href="#">PageSize</a>	Gets or sets the number of items in a page.
	<a href="#">Scope</a>	Gets the <a href="#">client scope</a> to which this client view belongs. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">TotalItemCount</a>	Gets the total number of entities in the view before paging is applied.
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## See Also

### Reference

[PagingView<T> Class](#)  
[C1.Data Namespace](#)

### LoadSize Property

[C1.Data Namespace](#) > [PagingView<T> Class](#) : LoadSize Property

Gets or sets a value controlling the number of entities to load in one batch.

## Syntax

Visual Basic (Declaration)



<b>Public Property</b> LoadSize <b>As</b> System.Integer
C#
<b>public</b> System. <b>int</b> LoadSize { <b>get</b> ; <b>set</b> ;}

## Remarks

Entities will be loaded using the multiple of [PageSize](#) nearest [LoadSize](#). This allows multiple pages to be loaded at once without loading partial pages.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PagingView<T> Class](#)

[PagingView<T> Members](#)

### PageCount Property

[C1.Data Namespace](#) > [PagingView<T> Class](#) : PageCount Property

Gets the number of pages in this [view](#).

## Syntax

Visual Basic (Declaration)
<b>Public ReadOnly Property</b> PageCount <b>As</b> System.Integer
C#
<b>public</b> System. <b>int</b> PageCount { <b>get</b> ;}

## Remarks

If [PageSize](#) is 0, [PageCount](#) is also 0.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PagingView<T> Class](#)

[PagingView<T> Members](#)

### PageIndex Property

[C1.Data Namespace](#) > [PagingView<T> Class](#) : PageIndex Property

Gets or sets the index of the current page.

## Syntax

Visual Basic (Declaration)

```
Public Property PageIndex As System.Integer
```

C#

```
public System.int PageIndex {get; set;}
```

## Remarks

Setting this property value to an invalid value is ignored. A value is invalid if it is less than 0 or greater or equal to [PageCount](#). If there are no items in this [view](#), the only valid value for this property is 0.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PagingView<T> Class](#)

[PagingView<T> Members](#)

### PageSize Property

[C1.Data Namespace](#) > [PagingView<T> Class](#) : PageSize Property

Gets or sets the number of items in a page.

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> PageSize <b>As</b> System.Integer	
C#	
<b>public</b> System.int PageSize { <b>get</b> ; <b>set</b> ;}	

## Remarks

To disable paging, set this property to 0.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PagingView<T> Class](#)

[PagingView<T> Members](#)

### TotalItemCount Property

[C1.Data Namespace](#) > [PagingView<T> Class](#) : TotalItemCount Property

Gets the total number of entities in the view before paging is applied.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> TotalItemCount <b>As</b> System.Integer	
C#	
<b>public</b> System.int TotalItemCount { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[PagingView<T> Class](#)

[PagingView<T> Members](#)

## ProgressiveView<T>

[C1.Data Namespace](#) : [ProgressiveView<T> Class](#)

The type of the entities in this [view](#).

Represents a [client view](#) that loads entities sequentially (progressively) page by page. The user sees the result and can interact with it before all pages are loaded.

## Object Model

ProgressiveView<T>

## Syntax

Visual Basic (Declaration)

```
Public Class ProgressiveView(Of T)
    Inherits ClientView(Of T)
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T),
C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class ProgressiveView<T> : ClientView<T>,
C1.LiveLinq.Indexing.IIndexedSource<T>, C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

The type of the entities in this [view](#).

## Remarks

Sorting is required, loading entities progressively is impossible without sort.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

[C1.Data.ClientView<T>](#)  
**C1.Data.ProgressiveView<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ProgressiveView<T> Members](#)  
[C1.Data Namespace](#)

### Overview

[C1.Data Namespace](#) : [ProgressiveView<T>](#) Class

The type of the entities in this [view](#).

Represents a [client view](#) that loads entities sequentially (progressively) page by page. The user sees the result and can interact with it before all pages are loaded.

## Object Model

[ProgressiveView<T>](#)

## Syntax

Visual Basic (Declaration)

```
Public Class ProgressiveView(Of T)
    Inherits ClientView(Of T)
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T),
C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class ProgressiveView<T> : ClientView<T>,
C1.LiveLinq.Indexing.IIndexedSource<T>, C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

The type of the entities in this [view](#).

## Remarks

Sorting is required, loading entities progressively is impossible without sort.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

[C1.Data.ClientView<T>](#)

**C1.Data.ProgressiveView<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ProgressiveView<T> Members](#)

[C1.Data Namespace](#)




## Members












[Properties](#) [Methods](#) [Events](#)



[C1.Data Namespace](#) : [ProgressiveView<T>](#) Class

The following tables list the members exposed by [ProgressiveView<T>](#).

## Public Properties






	Name	Description
	<a href="#">AutoLoad</a>	Gets or sets a boolean value indicating whether the <a href="#">client view</a> must be loaded automatically when its data is accessed. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from

		<a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">IsLoaded</a>	Gets a value indicating whether the <a href="#">client view</a> is <a href="#">loaded</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsLoading</a>	Gets a value indicating whether the <a href="#">client view</a> is being <a href="#">loaded</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">LoadSize</a>	Gets or sets the size of a page. To disable progressive loading, set this property to 0.
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from

		<a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Scope</a>	Gets the <a href="#">client scope</a> to which this client view belongs. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )












[Top](#)

## Public Methods

	Name	Description
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">C1.LiveLinq.LiveViews.View</a> to a <a href="#">View&lt;dynamic&gt;</a> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsFiltered</a>	Filters the view on the server side using a <a href="#">predicate</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">AsFilteredBound&lt;TKey&gt;</a>	Filters the view on the server using a key selector function and configurable <a href="#">value</a> and <a href="#">operator</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )



⇒	<a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">CancelLoad</a>	Cancels the current <a href="#">loading operation</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒	<a href="#">Concat</a>	Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Include</a>	Specifies related objects to include while loading the <a href="#">client view</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Join&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

⇒  Load	Loads the entities of the <a href="#">client view</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒  Maintain	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒  OrderBy<TKey>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  OrderByDescending<TKey>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  Paging	Overloaded. Applies paging to this <a href="#">client view</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒  ProgressiveLoading	Overloaded. Specifies that the <a href="#">client view</a> loading is performed not in a single trip to the server but in multiple batches, each loading a page of a limited size so the user sees the result and can interact with it before all pages are loaded. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒  PurgeEmptyGroups	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒  Rebuild	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒  Refresh	Loads the entities of the <a href="#">client view</a> ignoring the client-side cache. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒  Select<TResult>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  SelectMany	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from

		<a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">C1.LiveLinq.LiveViews.View.Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Union</a>	Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## Public Events

	Name	Description
⚡	<a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⚡	<a href="#">Loaded</a>	Occurs when the <a href="#">client view</a> has finished <a href="#">loading</a> successfully, and also when an exception has been thrown during <a href="#">loading</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[ProgressiveView<T> Class](#)












[C1.Data Namespace](#)




## Properties

[C1.Data Namespace](#) : [ProgressiveView<T> Class](#)

For a list of all members of this type, see [ProgressiveView<T> members](#).

## Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	Gets or sets a boolean value indicating whether the <a href="#">client view</a> must be loaded automatically when its data is accessed. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">IsLoaded</a>	Gets a value indicating whether the <a href="#">client view</a> is <a href="#">loaded</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsLoading</a>	Gets a value indicating whether the <a href="#">client view</a> is being <a href="#">loaded</a> . (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">LoadSize</a>	Gets or sets the size of a page. To disable progressive loading, set this property to 0.

	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Scope</a>	Gets the <a href="#">client scope</a> to which this client view belongs. (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## See Also

### Reference

[ProgressiveView<T> Class](#)

[C1.Data Namespace](#)

### LoadSize Property

[C1.Data Namespace](#) > [ProgressiveView<T> Class](#) : LoadSize Property

Gets or sets the size of a page. To disable progressive loading, set this property to 0.

## Syntax

Visual Basic (Declaration)

```
Public Property LoadSize As System.Integer
```

C#	
<code>public System.int LoadSize {get; set;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ProgressiveView<T> Class](#)  
[ProgressiveView<T> Members](#)

## SavedChangesEventArgs

[C1.Data Namespace](#) : SavedChangesEventArgs Class

Provides data for the C1DataSource.SavedChanges event.

## Object Model

SavedChangesEventArgs

## Syntax

Visual Basic (Declaration)	
<code>Public Class SavedChangesEventArgs</code> <code>Inherits System.EventArgs</code>	
C#	
<code>public class SavedChangesEventArgs : System.EventArgs</code>	

## Inheritance Hierarchy

System.Object  
     System.EventArgs  
         **C1.Data.SavedChangesEventArgs**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[SavedChangesEventArgs Members](#)  
[C1.Data Namespace](#)

Overview

[C1.Data Namespace](#) : SavedChangesEventArgs Class

Provides data for the C1DataSource.SavedChanges event.

Object Model

SavedChangesEventArgs

Syntax

Visual Basic (Declaration)	
Public Class SavedChangesEventArgs Inherits System.EventArgs	
C#	
public class SavedChangesEventArgs : System.EventArgs	

Inheritance Hierarchy

System.Object  
System.EventArgs  
C1.Data.SavedChangesEventArgs

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

## [SavedChangesEventArgs Members](#)

### [C1.Data Namespace](#)




## Members

[Properties](#) [Methods](#)

[C1.Data Namespace](#) : [SavedChangesEventArgs](#) Class


The following tables list the members exposed by [SavedChangesEventArgs](#).

## Public Properties

	Name	Description
	<a href="#">Error</a>	Gets a value showing the error that occurred during a save operation.
	<a href="#">HasError</a>	Gets a value indicating whether the save operation has failed. If true, inspect the <a href="#">Error</a> property for details.
	<a href="#">IsErrorHandled</a>	Gets a value indicating whether the error has been marked as handled by calling <a href="#">MarkErrorAsHandled</a> .

[Top](#)

## Public Methods

	Name	Description
	<a href="#">MarkErrorAsHandled</a>	If this method is called for a failed operation (if <a href="#">HasError</a> is true), it marks the error as handled. Otherwise, an exception is thrown.

[Top](#)

## See Also

### Reference

[SavedChangesEventArgs Class](#)

[C1.Data Namespace](#)


## Methods

[C1.Data Namespace](#) : [SavedChangesEventArgs](#) Class

For a list of all members of this type, see [SavedChangesEventArgs members](#).



## Public Methods

	Name	Description
	<a href="#">MarkErrorAsHandled</a>	If this method is called for a failed operation (if <a href="#">HasError</a> is true), it marks the error as handled. Otherwise, an exception is thrown.

[Top](#)

## See Also

### Reference

[SavedChangesEventArgs Class](#)

[C1.Data Namespace](#)

### MarkErrorAsHandled Method

[C1.Data Namespace](#) > [SavedChangesEventArgs Class](#) : MarkErrorAsHandled Method

If this method is called for a failed operation (if [HasError](#) is true), it marks the error as handled. Otherwise, an exception is thrown.

## Syntax

Visual Basic (Declaration)	
<pre>Public Sub MarkErrorAsHandled()</pre>	
C#	
<pre>public void MarkErrorAsHandled()</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SavedChangesEventArgs Class](#)




[SavedChangesEventArgs Members](#)

# Properties

C1.Data Namespace : SavedChangesEventArgs Class

For a list of all members of this type, see [SavedChangesEventArgs members](#).

## Public Properties

	Name	Description
	<a href="#">Error</a>	Gets a value showing the error that occurred during a save operation.
	<a href="#">HasError</a>	Gets a value indicating whether the save operation has failed. If true, inspect the <a href="#">Error</a> property for details.
	<a href="#">IsErrorHandled</a>	Gets a value indicating whether the error has been marked as handled by calling <a href="#">MarkErrorAsHandled</a> .

[Top](#)

## See Also

### Reference

[SavedChangesEventArgs Class](#)

[C1.Data Namespace](#)

### Error Property

[C1.Data Namespace](#) > [SavedChangesEventArgs Class](#) : Error Property

Gets a value showing the error that occurred during a save operation.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Error As System.Exception</code>	
C#	
<code>public System.Exception Error {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SavedChangesEventArgs Class](#)  
[SavedChangesEventArgs Members](#)

### HasError Property

[C1.Data Namespace](#) > [SavedChangesEventArgs Class](#) : HasError Property

Gets a value indicating whether the save operation has failed. If true, inspect the [Error](#) property for details.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property HasError As System.Boolean</code>	
C#	
<code>public System.bool HasError {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SavedChangesEventArgs Class](#)  
[SavedChangesEventArgs Members](#)

### IsErrorHandled Property

[C1.Data Namespace](#) > [SavedChangesEventArgs Class](#) : IsErrorHandled Property

Gets a value indicating whether the error has been marked as handled by calling [MarkErrorAsHandled](#).

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property IsErrorHandled As System.Boolean</code>	
C#	
<code>public System.bool IsErrorHandled {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference




[SavedChangesEventArgs Class](#)







[SavedChangesEventArgs Members](#)

# C1.Data.DataSource Namespace




## Overview

### Classes

	Class	Description
	<a href="#">BaseControlHandler</a>	A base class for control handlers that connect GUI controls of supported types to a <a href="#">C1DataSource</a> so that those controls can be given additional functionality such as <a href="#">auto-lookup columns</a> and <a href="#">virtual mode</a> .
	<a href="#">ClientCollectionView</a>	The collection view implementation used by a <a href="#">ClientViewSource</a> and other Studio for Entity Framework data sources.
	<a href="#">ClientViewSource</a>	Data source object exposing data from <a href="#">C1.Data.ClientCacheBase</a> to which GUI controls can bind. Using a <a href="#">ClientViewSource</a> , you can <a href="#">load</a> , <a href="#">filter</a> , <a href="#">group</a> , and <a href="#">sort</a> data easily.

	<a href="#">ClientViewSourceException</a>	This exception indicates that a <a href="#">ClientViewSource</a> is misconfigured or an error has occurred during the <a href="#">ClientViewSource.Load</a> operation.
	<a href="#">DependencyObjectCollection&lt;T&gt;</a>	An observable collection of <b>dependency objects</b> .
	<a href="#">FilterDescriptor</a>	Descriptor used by the <a href="#">ClientViewSource</a> to filter data in queries.
	<a href="#">FilterDescriptorCollection</a>	Collection of <a href="#">FilterDescriptor</a> dependency objects.
	<a href="#">GroupDescriptor</a>	Descriptor used by the <a href="#">ClientViewSource</a> to group data returned from server-side queries.
	<a href="#">GroupDescriptorCollection</a>	Collection of <a href="#">GroupDescriptor</a> dependency objects.
	<a href="#">SortDescriptor</a>	Descriptor used by the <a href="#">ClientViewSource</a> to sort data returned from queries.
	<a href="#">SortDescriptorCollection</a>	Collection of <a href="#">SortDescriptor</a> dependency objects.

## Enumerations

	Enumeration	Description
	<a href="#">FilterDescriptorLogicalOperator</a>	Enumeration of logical operators for filter collections.
	<a href="#">FilterOperator</a>	Operator used in <a href="#">FilterDescriptor</a> class.
	<a href="#">VirtualModeKind</a>	Enumeration of possible virtual modes a <a href="#">ClientViewSource</a> can be in. Used in the <a href="#">ClientViewSource.VirtualMode</a> property.

## See Also

### Reference

[C1.Data.Entity.4 Assembly](#)

# Classes

## BaseControlHandler

[C1.Data.DataSource Namespace](#) : BaseControlHandler Class

A base class for control handlers that connect GUI controls of supported types to a C1DataSource so that those controls can be given additional functionality such as [auto-lookup columns](#) and [virtual mode](#).

## Object Model

BaseControlHandler

## Syntax

Visual Basic (Declaration)

```
Public MustInherit Class BaseControlHandler
    Inherits System.Windows.DependencyObject
```

C#

```
public abstract class BaseControlHandler : System.Windows.DependencyObject
```

## Remarks

Use platform-specific control handlers for your controls: C1.Win.Data.ControlHandler, C1.WPF.Data.ControlHandler, and C1.Silverlight.Data.ControlHandler.

The list of supported GUI controls for each platform can be found in the reference of that platform's ControlHandler.

## Inheritance Hierarchy

System.Object

System.Windows.Threading.DispatcherObject

System.Windows.DependencyObject

**C1.Data.DataSource.BaseControlHandler**

[C1.Silverlight.Data.ControlHandler](#)

[C1.WPF.Data.ControlHandler](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[BaseControlHandler Members](#)  
[C1.Data.DataSource Namespace](#)

## Overview

[C1.Data.DataSource Namespace](#) : BaseControlHandler Class

A base class for control handlers that connect GUI controls of supported types to a C1DataSource so that those controls can be given additional functionality such as [auto-lookup columns](#) and [virtual mode](#).

# Object Model

BaseControlHandler

## Syntax

Visual Basic (Declaration)	
<pre>Public MustInherit Class BaseControlHandler     Inherits System.Windows.DependencyObject</pre>	
C#	
<pre>public abstract class BaseControlHandler : System.Windows.DependencyObject</pre>	

## Remarks

Use platform-specific control handlers for your controls: C1.Win.Data.ControlHandler, C1.WPF.Data.ControlHandler, and C1.Silverlight.Data.ControlHandler.

The list of supported GUI controls for each platform can be found in the reference of that platform's ControlHandler.

## Inheritance Hierarchy

System.Object  
  System.Windows.Threading.DispatcherObject  
    System.Windows.DependencyObject  
      **C1.Data.DataSource.BaseControlHandler**  
        [C1.Silverlight.Data.ControlHandler](#)  
        [C1.WPF.Data.ControlHandler](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Members](#)  
[C1.Data.DataSource Namespace](#)



## Members

[Fields](#) [Properties](#) [Methods](#)

[C1.Data.DataSource Namespace](#) : [BaseControlHandler Class](#)




The following tables list the members exposed by [BaseControlHandler](#).

## Public Fields




	Name	Description
 <b>S</b>	<a href="#">AutoLookupProperty</a>	The DependencyProperty for the <a href="#">AutoLookup</a> property.
 <b>S</b>	<a href="#">VirtualModeProperty</a>	The DependencyProperty for the <a href="#">VirtualMode</a> property.

[Top](#)

## Public Properties












	Name	Description
	<a href="#">AutoLookup</a>	Gets or sets a value indicating whether data grid columns bound to navigation (foreign key, lookup) properties must be converted to combo box columns, so the user can see the right value and edit it by choosing a value from a drop-down list. The default value is False.
	<a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)



	<a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SupportsVirtualMode</a>	Gets a value indicating whether this <a href="#">control handler</a> supports Virtual Mode.
	<a href="#">VirtualMode</a>	Gets or sets a value indicating whether virtual mode specified in a <a href="#">ClientViewSource</a> is managed by this control handler.


[Top](#)

## Public Methods

	Name	Description
	<a href="#">Apply</a>	Forces this <a href="#">control handler</a> to apply its settings to the current control.
	<a href="#">ClearValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)
	<a href="#">CoerceValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Equals</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetHashCode</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetLocalValueEnumerator</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">InvalidateProperty</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SetCurrentValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SetValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)

[Top](#)

## Protected Methods

	Name	Description
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Windows.DependencyObject)

[Top](#)

## See Also

### Reference

[BaseControlHandler Class](#)









[C1.Data.DataSource Namespace](#)




## Methods

[C1.Data.DataSource Namespace](#) : [BaseControlHandler Class](#)

For a list of all members of this type, see [BaseControlHandler members](#).


## Public Methods

	Name	Description
	<a href="#">Apply</a>	Forces this <a href="#">control handler</a> to apply its settings to the current control.
	<a href="#">ClearValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)
	<a href="#">CoerceValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Equals</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetHashCode</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetLocalValueEnumerator</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">InvalidateProperty</a>	(Inherited from System.Windows.DependencyObject)

	<a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SetCurrentValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SetValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)

[Top](#)

## Protected Methods

	Name	Description
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Windows.DependencyObject)

[Top](#)

## See Also

### Reference

[BaseControlHandler Class](#)

[C1.Data.DataSource Namespace](#)

### Apply Method

[C1.Data.DataSource Namespace](#) > [BaseControlHandler Class](#) : Apply Method

Forces this [control handler](#) to apply its settings to the current control.

## Syntax

Visual Basic (Declaration)	
<code>Public Overridable Sub Apply()</code>	
C#	
<code>public virtual void Apply()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Class](#)







[BaseControlHandler Members](#)

## Properties

[C1.Data.DataSource Namespace](#) : [BaseControlHandler Class](#)

For a list of all members of this type, see [BaseControlHandler members](#).

## Public Properties

	Name	Description
	<a href="#">AutoLookup</a>	Gets or sets a value indicating whether data grid columns bound to navigation (foreign key, lookup) properties must be converted to combo box columns, so the user can see the right value and edit it by choosing a value from a drop-down list. The default value is False.
	<a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)
	<a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SupportsVirtualMode</a>	Gets a value indicating whether this <a href="#">control handler</a> supports Virtual Mode.
	<a href="#">VirtualMode</a>	Gets or sets a value indicating whether virtual mode specified in a <a href="#">ClientViewSource</a> is managed by this control handler.

[Top](#)

## See Also

### Reference

[BaseControlHandler Class](#)

[C1.Data.DataSource Namespace](#)

## AutoLookup Property

[C1.Data.DataSource Namespace](#) > [BaseControlHandler Class](#) : AutoLookup Property

Gets or sets a value indicating whether data grid columns bound to navigation (foreign key, lookup) properties must be converted to combo box columns, so the user can see the right value and edit it by choosing a value from a drop-down list. The default value is False.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DescriptionAttribute("Gets or sets a value indicating whether data grid columns bound to navigation (foreign key, lookup) properties must be converted to combo box columns.")&gt; &lt;System.ComponentModel.NotifyParentPropertyAttribute(True)&gt; &lt;System.ComponentModel.DefaultValueAttribute()&gt; Public Property AutoLookup As System.Boolean</pre>	
C#	
<pre>[System.ComponentModel.Description("Gets or sets a value indicating whether data grid columns bound to navigation (foreign key, lookup) properties must be converted to combo box columns.")] [System.ComponentModel.NotifyParentProperty(true)] [System.ComponentModel.DefaultValue()] public System.bool AutoLookup {get; set;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Class](#)

[BaseControlHandler Members](#)

## SupportsVirtualMode Property

[C1.Data.DataSource Namespace](#) > [BaseControlHandler Class](#) : SupportsVirtualMode Property

Gets a value indicating whether this [control handler](#) supports Virtual Mode.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DesignerSerializationVisibilityAttribute(DesignerSerializationVisibility.Hidden)&gt; &lt;System.ComponentModel.BrowsableAttribute(False)&gt; Public Overridable ReadOnly Property SupportsVirtualMode As System.Boolean</pre>	
C#	
<pre>[System.ComponentModel.DesignerSerializationVisibility(DesignerSerializationVisibility.Hidden)] [System.ComponentModel.Browsable(false)] public virtual System.bool SupportsVirtualMode {get;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Class](#)

[BaseControlHandler Members](#)

[VirtualMode Property](#)

### VirtualMode Property

[C1.Data.DataSource Namespace](#) > [BaseControlHandler Class](#) : VirtualMode Property

Gets or sets a value indicating whether virtual mode specified in a [ClientViewSource](#) is managed by this control handler.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.NotifyParentPropertyAttribute(True)&gt; &lt;System.ComponentModel.DescriptionAttribute("Gets or sets a value indicating whether virtual mode specified in a ClientViewSource is managed by this control.")&gt; &lt;System.ComponentModel.DefaultValueAttribute()&gt; Public Property VirtualMode As System.Boolean</pre>	

C#

```
[System.ComponentModel.NotifyParentProperty(true)]  
[System.ComponentModel.Description("Gets or sets a value indicating whether  
virtual mode specified in a ClientViewSource is managed by this control.")]  
[System.ComponentModel.DefaultValue()]  
public System.bool VirtualMode {get; set;}
```

## Remarks

Setting this property to True has effect only if [ClientViewSource.VirtualMode](#) of the associated [ClientViewSource](#) is set to [Managed](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Class](#)



[BaseControlHandler Members](#)

### Fields

[C1.Data.DataSource Namespace](#) : [BaseControlHandler Class](#)

For a list of all members of this type, see [BaseControlHandler members](#).

## Public Fields

	Name	Description
 S	<a href="#">AutoLookupProperty</a>	The <a href="#">DependencyProperty</a> for the <a href="#">AutoLookup</a> property.
 S	<a href="#">VirtualModeProperty</a>	The <a href="#">DependencyProperty</a> for the <a href="#">VirtualMode</a> property.

[Top](#)

## See Also

### Reference

[BaseControlHandler Class](#)  
[C1.Data.DataSource Namespace](#)

## AutoLookupProperty Field

[C1.Data.DataSource Namespace](#) > [BaseControlHandler Class](#) : AutoLookupProperty Field

The DependencyProperty for the [AutoLookup](#) property.

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared ReadOnly AutoLookupProperty As System.Windows.DependencyProperty</pre>	
C#	
<pre>public static readonly System.Windows.DependencyProperty AutoLookupProperty</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Class](#)  
[BaseControlHandler Members](#)

## VirtualModeProperty Field

[C1.Data.DataSource Namespace](#) > [BaseControlHandler Class](#) : VirtualModeProperty Field

The DependencyProperty for the [VirtualMode](#) property.

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared ReadOnly VirtualModeProperty As System.Windows.DependencyProperty</pre>	
C#	
<pre>public static readonly System.Windows.DependencyProperty VirtualModeProperty</pre>	



## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Class](#)

[BaseControlHandler Members](#)

## ClientCollectionView

[C1.Data.DataSource Namespace](#) : ClientCollectionView Class

The collection view implementation used by a [ClientViewSource](#) and other Studio for Entity Framework data sources.

## Object Model

ClientCollectionView

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>  
Public Class ClientCollectionView
```

C#

```
[System.Reflection.DefaultMember("Item")]  
public class ClientCollectionView
```

## Inheritance Hierarchy

System.Object

**C1.Data.DataSource.ClientCollectionView**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Members](#)  
[C1.Data.DataSource Namespace](#)

## Overview

[C1.Data.DataSource Namespace](#) : ClientCollectionView Class

The collection view implementation used by a [ClientViewSource](#) and other Studio for Entity Framework data sources.

## Object Model

ClientCollectionView

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>  
Public Class ClientCollectionView
```

C#

```
[System.Reflection.DefaultMember("Item")]  
public class ClientCollectionView
```

## Inheritance Hierarchy

System.Object

**C1.Data.DataSource.ClientCollectionView**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Members](#)  
[C1.Data.DataSource Namespace](#)












## Members





[Properties](#) [Methods](#) [Events](#)

[C1.Data.DataSource Namespace](#) : [ClientCollectionView](#) Class

The following tables list the members exposed by [ClientCollectionView](#).










### Public Properties

	Name	Description
	<a href="#">CanAdd</a>	Gets a value indicating whether the <a href="#">Add</a> method is supported.
	<a href="#">CanChangePage</a>	Gets a value that indicates whether the <a href="#">PageIndex</a> value can change.
	<a href="#">CanRemove</a>	Gets a value that indicates whether an item can be removed from the collection.
	<a href="#">CollectionViewFactory</a>	Gets an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> .
	<a href="#">Count</a>	Gets the number of elements contained in the <a href="#">ClientCollectionView</a> .
	<a href="#">CurrentItem</a>	Gets the current item in the view.
	<a href="#">CurrentPosition</a>	Gets the ordinal position of the <a href="#">CurrentItem</a> within the collection view.
	<a href="#">IsEmpty</a>	Returns a value that indicates whether the resulting view is empty.
	<a href="#">IsPageChanging</a>	Gets a value that indicates whether the page index is changing.
	<a href="#">Item</a>	Gets the element at the specified <a href="#">index</a> .
	<a href="#">ItemProperties</a>	Gets a collection that contains information about the properties

		that are available on the items in this <a href="#">ClientCollectionView</a> .
	<a href="#">PageCount</a>	Gets the count of the pages in this view.
	<a href="#">PageIndex</a>	Gets the zero-based index of the current page.
	<a href="#">PageSize</a>	Gets or sets the number of items to display on a page.
	<a href="#">TotalItemCount</a>	Gets the total number of items in the view before paging is applied.

[Top](#)

## Public Methods

	Name	Description
	<a href="#">Add</a>	Adds a new <a href="#">entity</a> to the client-side cache and to the associated context. The <a href="#">entity</a> will appear in this <a href="#">collection view</a> if it matches the underlying query.
	<a href="#">AsLive&lt;T&gt;</a>	Converts this <a href="#">ClientCollectionView</a> to a <a href="#">live view</a> .
	<a href="#">Contains</a>	Returns a value that indicates whether a given item belongs to this collection view.
	<a href="#">IndexOf</a>	Determines the index of a specific <a href="#">item</a> in the <a href="#">ClientCollectionView</a> .
	<a href="#">MoveCurrentTo</a>	Sets the specified item to be the <a href="#">CurrentItem</a> in the view.
	<a href="#">MoveCurrentToFirst</a>	Sets the first item in the view as the <a href="#">CurrentItem</a> .
	<a href="#">MoveCurrentToLast</a>	Sets the last item in the view as the <a href="#">CurrentItem</a> .
	<a href="#">MoveCurrentToNext</a>	Sets the item after the <a href="#">CurrentItem</a> in the view as the <b>System.ComponentModel.ICollectionView.CurrentItem</b> .
	<a href="#">MoveCurrentToPosition</a>	Sets the item at the specified index to be the <a href="#">CurrentItem</a> in the

		view.
≡💡	<a href="#">MoveCurrentToPrevious</a>	Sets the item before the <a href="#">CurrentItem</a> in the view as the <a href="#">CurrentItem</a> .
≡💡	<a href="#">MoveToFirstPage</a>	Sets the first page as the current page.
≡💡	<a href="#">MoveToLastPage</a>	Sets the last page as the current page.
≡💡	<a href="#">MoveToNextPage</a>	Moves to the page after the current page.
≡💡	<a href="#">MoveToPage</a>	Sets the first page as the current page.
≡💡	<a href="#">MoveToPreviousPage</a>	Moves to the page before the current page.
≡💡	<a href="#">Remove</a>	Removes the specified item from the collection.
≡💡	<a href="#">RemoveAt</a>	Removes the item at the specified position from the collection.

[Top](#)

## Public Events

	Name	Description
⚡	<a href="#">CurrentChanged</a>	When implementing this interface, raise this event after the current item has been changed.
⚡	<a href="#">CurrentChanging</a>	When implementing this interface, raise this event before changing the current item. Event handler can cancel this event.
⚡	<a href="#">PageChanged</a>	Occurs after the <a href="#">PageIndex</a> has changed.
⚡	<a href="#">PageChanging</a>	Occurs before changing the <a href="#">PageIndex</a> .
⚡	<a href="#">PropertyChanged</a>	Occurs when a property value changes.

[Top](#)

## See Also

## Reference

[ClientCollectionView Class](#)

[C1.Data.DataSource Namespace](#)








## Methods

[C1.Data.DataSource Namespace](#) : [ClientCollectionView Class](#)

For a list of all members of this type, see [ClientCollectionView members](#).

## Public Methods

	Name	Description
≡	<a href="#">Add</a>	Adds a new <a href="#">entity</a> to the client-side cache and to the associated context. The <a href="#">entity</a> will appear in this <a href="#">collection view</a> if it matches the underlying query.
≡	<a href="#">AsLive&lt;T&gt;</a>	Converts this <a href="#">ClientCollectionView</a> to a <a href="#">live view</a> .
≡	<a href="#">Contains</a>	Returns a value that indicates whether a given item belongs to this collection view.
≡	<a href="#">IndexOf</a>	Determines the index of a specific <a href="#">item</a> in the <a href="#">ClientCollectionView</a> .
≡	<a href="#">MoveCurrentTo</a>	Sets the specified item to be the <a href="#">CurrentItem</a> in the view.
≡	<a href="#">MoveCurrentToFirst</a>	Sets the first item in the view as the <a href="#">CurrentItem</a> .
≡	<a href="#">MoveCurrentToLast</a>	Sets the last item in the view as the <a href="#">CurrentItem</a> .
≡	<a href="#">MoveCurrentToNext</a>	Sets the item after the <a href="#">CurrentItem</a> in the view as the <b>System.ComponentModel.ICollectionView.CurrentItem</b> .
≡	<a href="#">MoveCurrentToPosition</a>	Sets the item at the specified index to be the <a href="#">CurrentItem</a> in the view.
≡	<a href="#">MoveCurrentToPrevious</a>	Sets the item before the <a href="#">CurrentItem</a> in the view as the <a href="#">CurrentItem</a> .

	<a href="#">MoveToFirstPage</a>	Sets the first page as the current page.
	<a href="#">MoveToLastPage</a>	Sets the last page as the current page.
	<a href="#">MoveToNextPage</a>	Moves to the page after the current page.
	<a href="#">MoveToPage</a>	Sets the first page as the current page.
	<a href="#">MoveToPreviousPage</a>	Moves to the page before the current page.
	<a href="#">Remove</a>	Removes the specified item from the collection.
	<a href="#">RemoveAt</a>	Removes the item at the specified position from the collection.

[Top](#)

## See Also

### Reference

[ClientCollectionView Class](#)  
[C1.Data.DataSource Namespace](#)

### Add Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : Add Method

The new entity to add.

Adds a new [entity](#) to the client-side cache and to the associated context. The [entity](#) will appear in this [collection view](#) if it matches the underlying query.

## Syntax

Visual Basic (Declaration)

```
Public Sub Add( _
    ByVal entity As System.Object _
)
```

C#

```
public void Add(
    System.object entity
)
```

## Parameters

*entity*

The new entity to add.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### AsLive<T> Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : AsLive<T> Method

The type of the elements in this collection view.

Converts this [ClientCollectionView](#) to a [live view](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Function AsLive(Of T)() As View(Of T)</pre>	
C#	
<pre>public View&lt;T&gt; AsLive&lt;T&gt;()</pre>	

### Type Parameters

*T*

The type of the elements in this collection view.

### Return Value

The resulting [live view](#).

## Exceptions



Exception	Description
<b>System.NotSupportedException</b>	The <a href="#">ClientViewSource</a> is in <a href="#">virtual mode</a> .

## Remarks

This method does not change the [ClientCollectionView](#) in any way, it just exposes its live view functionality.

This method is not supported for a [ClientViewSource](#) in [virtual mode](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### Contains Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : Contains Method

The object to check.

Returns a value that indicates whether a given item belongs to this collection view.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function Contains( _     ByVal item As System.Object _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool Contains(     System.object item )</pre>	

### Parameters

*item*

The object to check.

## Return Value

true if the item belongs to this collection view; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## IndexOf Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : IndexOf Method

The item to locate in the [ClientCollectionView](#).

Determines the index of a specific *item* in the [ClientCollectionView](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Function IndexOf( _     ByVal item As System.Object _ ) As System.Integer</pre>	
C#	
<pre>public System.int IndexOf(     System.object item )</pre>	

## Parameters

*item*

The item to locate in the [ClientCollectionView](#).

## Return Value

The index of the [item](#) if found in the list; otherwise, -1.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveCurrentTo Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveCurrentTo Method

The item to set as the [CurrentItem](#).

Sets the specified item to be the [CurrentItem](#) in the view.

## Syntax

Visual Basic (Declaration)

```
Public Function MoveCurrentTo( _  
    ByVal item As System.Object _  
) As System.Boolean
```

C#

```
public System.bool MoveCurrentTo(  
    System.object item  
)
```

### Parameters

*item*

The item to set as the [CurrentItem](#).

### Return Value

true if the resulting [CurrentItem](#) is within the view; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveCurrentToFirst Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveCurrentToFirst Method

Sets the first item in the view as the [CurrentItem](#).

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> MoveCurrentToFirst() <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool MoveCurrentToFirst()	

### Return Value

true if the resulting [CurrentItem](#) is an item within the view; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveCurrentToLast Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveCurrentToLast Method

Sets the last item in the view as the [CurrentItem](#).

## Syntax

Visual Basic (Declaration)

```
Public Function MoveCurrentToLast() As System.Boolean
```

C#

```
public System.bool MoveCurrentToLast()
```

### Return Value

true if the resulting [CurrentItem](#) is an item within the view; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveCurrentToNext Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveCurrentToNext Method

Sets the item after the [CurrentItem](#) in the view as the

**System.ComponentModel.ICollectionView.CurrentItem.**

## Syntax

Visual Basic (Declaration)

```
Public Function MoveCurrentToNext() As System.Boolean
```

C#

```
public System.bool MoveCurrentToNext()
```

### Return Value

true if the resulting [CurrentItem](#) is an item within the view; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveCurrentToPosition Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveCurrentToPosition Method

The index to set the **System.ComponentModel.ICollectionView.CurrentItem** to.

Sets the item at the specified index to be the [CurrentItem](#) in the view.

## Syntax

Visual Basic (Declaration)

```
Public Function MoveCurrentToPosition( _  
    ByVal position As System.Integer _  
) As System.Boolean
```

C#

```
public System.bool MoveCurrentToPosition(  
    System.int position  
)
```

### Parameters

*position*

The index to set the **System.ComponentModel.ICollectionView.CurrentItem** to.

### Return Value

true if the resulting [CurrentItem](#) is an item within the view; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveCurrentToPrevious Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveCurrentToPrevious Method

Sets the item before the [CurrentItem](#) in the view as the [CurrentItem](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Function MoveCurrentToPrevious() As System.Boolean</pre>	
C#	
<pre>public System.bool MoveCurrentToPrevious()</pre>	

### Return Value

true if the resulting [CurrentItem](#) is an item within the view; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveToFirstPage Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveToFirstPage Method

Sets the first page as the current page.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<b>Public Function</b> MoveToFirstPage() <b>As</b> System.Boolean
---

C#
----

<b>public</b> System.bool MoveToFirstPage()
---

### Return Value

true if the operation was successful; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveToLastPage Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveToLastPage Method

Sets the last page as the current page.

## Syntax

Visual Basic (Declaration)
----------------------------

<b>Public Function</b> MoveToLastPage() <b>As</b> System.Boolean
--

C#
----

<b>public</b> System.bool MoveToLastPage()
--

### Return Value

true if the operation was successful; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



## Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## MoveToNextPage Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveToNextPage Method

Moves to the page after the current page.

## Syntax

Visual Basic (Declaration)

```
Public Function MoveToNextPage() As System.Boolean
```

C#

```
public System.bool MoveToNextPage()
```

## Return Value

true if the operation was successful; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## MoveToPage Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveToPage Method

The index of the page to move to.

Sets the first page as the current page.

## Syntax

Visual Basic (Declaration)

```
Public Function MoveToPage( _
    ByVal pageIndex As System.Integer _
) As System.Boolean
```

C#

```
public System.bool MoveToPage(
    System.int pageIndex
)
```

## Parameters

*pageIndex*

The index of the page to move to.

## Return Value

True if the operation was successful; otherwise, False.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## MoveToPreviousPage Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveToPreviousPage Method

Moves to the page before the current page.

## Syntax

Visual Basic (Declaration)

```
Public Function MoveToPreviousPage() As System.Boolean
```

C#

```
public System.bool MoveToPreviousPage()
```

## Return Value

true if the operation was successful; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### Remove Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : Remove Method

The item to remove.

Removes the specified item from the collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public Sub Remove( _     ByVal item As System.Object _ )</pre>	
C#	
<pre>public void Remove(     System.object item )</pre>	

### Parameters

*item*

The item to remove.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### RemoveAt Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : RemoveAt Method

The position of the item to remove.

Removes the item at the specified position from the collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public Sub RemoveAt( _     ByVal index As System.Integer _ )</pre>	
C#	
<pre>public void RemoveAt(     System.int index )</pre>	

### Parameters

*index*

The position of the item to remove.

## Exceptions

Exception	Description
<b>System.ArgumentOutOfRangeException</b>	index is less than 0 or greater than the number of items in the collection view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)










[ClientCollectionView Members](#)







### Properties

[C1.Data.DataSource Namespace](#) : [ClientCollectionView Class](#)

For a list of all members of this type, see [ClientCollectionView members](#).

### Public Properties

	Name	Description
	<a href="#">CanAdd</a>	Gets a value indicating whether the <a href="#">Add</a> method is supported.
	<a href="#">CanChangePage</a>	Gets a value that indicates whether the <a href="#">PageIndex</a> value can change.
	<a href="#">CanRemove</a>	Gets a value that indicates whether an item can be removed from the collection.
	<a href="#">CollectionViewFactory</a>	Gets an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> .
	<a href="#">Count</a>	Gets the number of elements contained in the <a href="#">ClientCollectionView</a> .
	<a href="#">CurrentItem</a>	Gets the current item in the view.
	<a href="#">CurrentPosition</a>	Gets the ordinal position of the <a href="#">CurrentItem</a> within the collection view.
	<a href="#">IsEmpty</a>	Returns a value that indicates whether the resulting view is empty.
	<a href="#">IsPageChanging</a>	Gets a value that indicates whether the page index is changing.

	<a href="#">Item</a>	Gets the element at the specified <a href="#">index</a> .
	<a href="#">ItemProperties</a>	Gets a collection that contains information about the properties that are available on the items in this <a href="#">ClientCollectionView</a> .
	<a href="#">PageCount</a>	Gets the count of the pages in this view.
	<a href="#">PageIndex</a>	Gets the zero-based index of the current page.
	<a href="#">PageSize</a>	Gets or sets the number of items to display on a page.
	<a href="#">TotalItemCount</a>	Gets the total number of items in the view before paging is applied.

[Top](#)

## See Also

### Reference

[ClientCollectionView Class](#)

[C1.Data.DataSource Namespace](#)

### CanAdd Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CanAdd Property

Gets a value indicating whether the [Add](#) method is supported.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> CanAdd <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool CanAdd { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### CanChangePage Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CanChangePage Property

Gets a value that indicates whether the [PageIndex](#) value can change.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> CanChangePage <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool CanChangePage { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### CanRemove Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CanRemove Property

Gets a value that indicates whether an item can be removed from the collection.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> CanRemove <b>As</b> System.Boolean	
C#	

```
public System.bool CanRemove {get;}
```

### Property Value

true if an item can be removed from the collection; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### CollectionViewFactory Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CollectionViewFactory Property

Gets an instance of **System.ComponentModel.ICollectionViewFactory** that can be used as a source of a **System.Windows.Data.CollectionViewSource**.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property CollectionViewFactory As  
System.ComponentModel.ICollectionViewFactory
```

C#

```
public System.ComponentModel.ICollectionViewFactory CollectionViewFactory  
{get;}
```

### Property Value

A factory that returns the same [ClientCollectionView](#) as an **System.ComponentModel.ICollectionView**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2



## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### Count Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : Count Property

Gets the number of elements contained in the [ClientCollectionView](#).

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> Count <b>As</b> System.Integer	
C#	
<b>public</b> System.int Count { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### CurrentItem Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CurrentItem Property

Gets the current item in the view.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> CurrentItem <b>As</b> System.Object	
C#	

```
public System.object CurrentItem {get;}
```

### Property Value

The current item of the view or null if there is no current item.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### CurrentPosition Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CurrentPosition Property

Gets the ordinal position of the [CurrentItem](#) within the collection view.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property CurrentPosition As System.Integer
```

C#

```
public System.int CurrentPosition {get;}
```

### Property Value

The ordinal position of the [CurrentItem](#) within the collection view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

## IsEmpty Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : IsEmpty Property

Returns a value that indicates whether the resulting view is empty.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property IsEmpty As System.Boolean</code>	
C#	
<code>public System.bool IsEmpty {get;}</code>	

### Property Value

true if the resulting view is empty; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

## IsPageChanging Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : IsPageChanging Property

Gets a value that indicates whether the page index is changing.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property IsPageChanging As System.Boolean</code>	
C#	

```
public System.bool IsPageChanging {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### Item Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : Item Property

The zero-based index of the element to get.

Gets the element at the specified [index](#).

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Default Property Item( _  
    ByVal index As System.Integer _  
) As System.Object
```

C#

```
public System.object this[  
    System.int index  
]; {get;}
```

### Parameters

*index*

The zero-based index of the element to get.

### Property Value

The element at the specified index.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

### ItemProperties Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : ItemProperties Property

Gets a collection that contains information about the properties that are available on the items in this [ClientCollectionView](#).

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property ItemProperties As  
System.Collections.ObjectModel.ReadOnlyCollection(Of ItemPropertyInfo)
```

C#

```
public System.Collections.ObjectModel.ReadOnlyCollection<ItemPropertyInfo>  
ItemProperties {get;}
```

### Property Value

A collection that contains information about the properties that are available on the items in this [ClientCollectionView](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

## PageCount Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : PageCount Property

Gets the count of the pages in this view.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property PageCount As System.Integer</code>	
C#	
<code>public System.int PageCount {get;}</code>	

## Remarks

When [PageSize](#) is 0, the [PageIndex](#) will also be 0.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## PageIndex Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : PageIndex Property

Gets the zero-based index of the current page.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property PageIndex As System.Integer</code>	
C#	
<code>public System.int PageIndex {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### PageSize Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : PageSize Property

Gets or sets the number of items to display on a page.

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> PageSize <b>As</b> System.Integer	
C#	
<b>public</b> System.int PageSize { <b>get</b> ; <b>set</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### TotalItemCount Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : TotalItemCount Property

Gets the total number of items in the view before paging is applied.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property TotalItemCount As System.Integer</code>	
C#	
<code>public System.int TotalItemCount {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2






## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

## Events

>

Name	Description
 <a href="#">CurrentChanged</a>	When implementing this interface, raise this event after the current item has been changed.
 <a href="#">CurrentChanging</a>	When implementing this interface, raise this event before changing the current item. Event handler can cancel this event.
 <a href="#">PageChanged</a>	Occurs after the <a href="#">PageIndex</a> has changed.
 <a href="#">PageChanging</a>	Occurs before changing the <a href="#">PageIndex</a> .
 <a href="#">PropertyChanged</a>	Occurs when a property value changes.

[Top](#)

## See Also

### Reference

[ClientCollectionView Class](#)  
[C1.Data.DataSource Namespace](#)



## CurrentChanged Event

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CurrentChanged Event

When implementing this interface, raise this event after the current item has been changed.

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> CurrentChanged <b>As</b> System.EventHandler	
C#	
<b>public event</b> System.EventHandler CurrentChanged	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

## CurrentChanging Event

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CurrentChanging Event

When implementing this interface, raise this event before changing the current item. Event handler can cancel this event.

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> CurrentChanging <b>As</b> System.ComponentModel.CurrentChangingEventHandler	
C#	
<b>public event</b> System.ComponentModel.CurrentChangingEventHandler CurrentChanging	

## Event Data

The event handler receives an argument of type `System.ComponentModel.CurrentChangingEventArgs` containing data related to this event. The following **CurrentChangingEventArgs** properties provide information specific to this event.

Property	Description
<b>Cancel</b>	Gets or sets a value that indicates whether to cancel the event.
<b>IsCancelable</b>	Gets a value that indicates whether the event is cancelable.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

### PageChanged Event

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : PageChanged Event

Occurs after the [PageIndex](#) has changed.

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> PageChanged <b>As</b> System.EventHandler(Of EventArgs)	
C#	
<b>public event</b> System.EventHandler<EventArgs> PageChanged	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## PageChanging Event

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : PageChanging Event

Occurs before changing the [PageIndex](#).

## Syntax

Visual Basic (Declaration)	
<code>Public Event PageChanging As System.EventHandler(Of PageChangingEventArgs)</code>	
C#	
<code>public event System.EventHandler&lt;PageChangingEventArgs&gt; PageChanging</code>	

## Event Data

The event handler receives an argument of type [PageChangingEventArgs](#) containing data related to this event. The following **PageChangingEventArgs** properties provide information specific to this event.

Property	Description
<a href="#">Cancel</a>	(Inherited from <a href="#">System.ComponentModel.CancelEventArgs</a> )
<a href="#">NewPageIndex</a>	Gets the index of the requested page.

## Remarks

This event allows to cancel the ongoing [PageIndex](#) change by setting **System.ComponentModel.CancelEventArgs.Cancel** to true.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

## PropertyChanged Event

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : PropertyChanged Event

Occurs when a property value changes.

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> PropertyChanged <b>As</b> System.ComponentModel.PropertyChangedEventHandler	
C#	
<b>public event</b> System.ComponentModel.PropertyChangedEventHandler PropertyChanged	

## Event Data

The event handler receives an argument of type `System.ComponentModel.PropertyChangedEventArgs` containing data related to this event. The following **PropertyChangedEventArgs** properties provide information specific to this event.

Property	Description
<b>PropertyName</b>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

## ClientViewSource

[C1.Data.DataSource Namespace](#) : ClientViewSource Class

Data source object exposing data from [C1.Data.ClientCacheBase](#) to which GUI controls can bind. Using a [ClientViewSource](#), you can [load](#), [filter](#), [group](#), and [sort](#) data easily.

## Object Model

[ClientViewSource](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Class ClientViewSource     Inherits System.Windows.DependencyObject</pre>	
C#	
<pre>public class ClientViewSource : System.Windows.DependencyObject</pre>	

## Inheritance Hierarchy

System.Object  
    System.Windows.Threading.DispatcherObject  
        System.Windows.DependencyObject  
            **C1.Data.DataSource.ClientViewSource**  
                [C1.Data.Entities.EntityViewSource](#)  
                [C1.Silverlight.Data.RiaServices.RiaViewSource](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Members](#)  
[C1.Data.DataSource Namespace](#)

### Overview

[C1.Data.DataSource Namespace](#) : [ClientViewSource Class](#)

Data source object exposing data from [C1.Data.ClientCacheBase](#) to which GUI controls can bind. Using a [ClientViewSource](#), you can [load](#), [filter](#), [group](#), and [sort](#) data easily.

# Object Model

ClientViewSource

## Syntax

Visual Basic (Declaration)	
<pre>Public Class ClientViewSource     Inherits System.Windows.DependencyObject</pre>	
C#	
<pre>public class ClientViewSource : System.Windows.DependencyObject</pre>	

## Inheritance Hierarchy

- System.Object
  - System.Windows.Threading.DispatcherObject
    - System.Windows.DependencyObject
      - C1.Data.DataSource.ClientViewSource**
        - [C1.Data.Entities.EntityViewSource](#)
        - [C1.Silverlight.Data.RiaServices.RiaViewSource](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

- [ClientViewSource Members](#)
- [C1.Data.DataSource Namespace](#)


## Members

- [Properties](#)
- [Methods](#)
- [Events](#)

[C1.Data.DataSource Namespace](#) : ClientViewSource Class










The following tables list the members exposed by [ClientViewSource](#).











## Public Constructors

	Name	Description
	<a href="#">ClientViewSource Constructor</a>	Initializes a new instance of the <a href="#">ClientViewSource</a> class.




[Top](#)

## Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	Gets or sets a value indicating whether <a href="#">Load</a> is automatically invoked on startup and when a change occurs that impacts the query composed by the <a href="#">ClientViewSource</a> . The default is True.
	<a href="#">BaseView</a>	Gets or sets an instance of <a href="#">C1.Data.ClientView&lt;T&gt;</a> that the <a href="#">ClientViewSource</a> uses as the base for composing queries.
	<a href="#">CacheTimeout</a>	Gets or sets the period of time entities loaded in virtual mode are kept in the cache without checking whether they are needed or not. If an entity was neither used nor considered needed for a period of time longer than <a href="#">CacheTimeout</a> , <a href="#">ClientViewSource</a> may evict it from the cache.
	<a href="#">CurrentClientView</a>	Gets the current <a href="#">client view</a> used to load entities, or null in <a href="#">virtual mode</a> .
	<a href="#">DataView</a>	Gets the current view of entities resulting from the last load operation.
	<a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)
	<a href="#">FilterDescriptors</a>	Gets the collection of <a href="#">FilterDescriptor</a> objects used when performing loads.
	<a href="#">FilterOperator</a>	Gets or sets the logical operator used for combining <a href="#">FilterDescriptors</a> in the filter collection. The default value is








		<a href="#">FilterDescriptorLogicalOperator.And</a> .
	<a href="#">GroupDescriptors</a>	Gets the collection of <a href="#">GroupDescriptor</a> objects used to organize the loaded entities into groups.
	<a href="#">Include</a>	Gets or sets a comma-separated list of property paths that specify related objects to include during the <a href="#">Load</a> operation.
	<a href="#">IsLoadingData</a>	Gets a value indicating whether the <a href="#">ClientViewSource</a> is currently loading data.
	<a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">LoadCommand</a>	Gets an <b>System.Windows.Input.ICommand</b> that invokes <a href="#">Load</a> on this <a href="#">ClientViewSource</a> .
	<a href="#">LoadDelay</a>	Gets or sets the delay before an automatic data loading operation is started. It is the delay from the time a change prompting automatic load occurs until the time the resulting <a href="#">Load</a> is started. The default delay is 25 milliseconds.
	<a href="#">LoadSize</a>	Gets or sets the maximum number of items to load each time a <a href="#">Load</a> is executed. When equal to 0, all requested entities will be loaded. The default is 0.
	<a href="#">MoveToFirstOnLoad</a>	Gets or sets a value indicating that the first item must be made current after <a href="#">Load</a> operation is completed if current item was not set by other means.
	<a href="#">Name</a>	Gets a name of this <a href="#">ClientViewSource</a> to reference it in a C1DataSource.ViewSources collection. By default it is determined by the EntitySetName (for Entity Framework) or QueryName (for RIA Services), but it can be overridden by <a href="#">NameOverride</a> .
	<a href="#">NameOverride</a>	Gets or sets a value that overrides the value of the <a href="#">Name</a> property.










	<a href="#">PageSize</a>	Gets or sets the number of items displayed on each page of the <a href="#">DataView</a> , or the number of items to fetch in each query in <a href="#">virtual mode</a> , or 0 to disable paging.
	<a href="#">SortDescriptors</a>	Gets the collection of <a href="#">SortDescriptor</a> objects used to sort the data.
	<a href="#">VirtualMode</a>	Gets or sets a value indicating whether the <a href="#">ClientViewSource</a> is in virtual mode. Virtual mode is an innovative technology allowing to bind GUI controls directly to very large data sets without delays and performance degradation and without inconvenience of paging. By default, virtual mode is disabled (the default value is <a href="#">VirtualModeKind.None</a> ).

[Top](#)


## Public Methods

	Name	Description
	<a href="#">ClearValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)
	<a href="#">CoerceValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">DeferLoad</a>	Used to group changes to multiple load-affecting properties together, deferring the resulting load operations so a single load operation is performed in the end, that is, when the object returned from this method is disposed.
	<a href="#">Equals</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetHashCode</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetLocalValueEnumerator</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)

 <a href="#">InvalidateProperty</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">Load</a>	Starts a load operation. Any pending load will be implicitly canceled.
 <a href="#">LoadRange</a>	If in <a href="#">virtual mode</a> , loads a specific range of entities.
 <a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">Refresh</a>	Starts a load operation ignoring the client-side cache. Any pending load will be implicitly canceled.
 <a href="#">SetCurrentValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">SetValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)



[Top](#)

## Protected Methods

	Name	Description
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Windows.DependencyObject)

[Top](#)

## Public Events

	Name	Description
	<a href="#">LoadedData</a>	Occurs when a load operation is completed, or when an exception was thrown during the load operation.
	<a href="#">PropertyChanged</a>	Occurs when a property value changes.

[Top](#)

## See Also

### Reference

[ClientViewSource Class](#)  
[C1.Data.DataSource Namespace](#)

## ClientViewSource Constructor

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : ClientViewSource Constructor

Initializes a new instance of the [ClientViewSource](#) class.

## Syntax

Visual Basic (Declaration)

```
Public Function New()
```

C#

```
public ClientViewSource()
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



[ClientViewSource Class](#)  
[ClientViewSource Members](#)













## Methods

[C1.Data.DataSource Namespace](#) : [ClientViewSource Class](#)

For a list of all members of this type, see [ClientViewSource members](#).


## Public Methods

	Name	Description
	<a href="#">ClearValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)
	<a href="#">CoerceValue</a>	(Inherited from System.Windows.DependencyObject)

 <a href="#">DeferLoad</a>	Used to group changes to multiple load-affecting properties together, deferring the resulting load operations so a single load operation is performed in the end, that is, when the object returned from this method is disposed.
 <a href="#">Equals</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">GetHashCode</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">GetLocalValueEnumerator</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">InvalidateProperty</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">Load</a>	Starts a load operation. Any pending load will be implicitly canceled.
 <a href="#">LoadRange</a>	If in <a href="#">virtual mode</a> , loads a specific range of entities.
 <a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">Refresh</a>	Starts a load operation ignoring the client-side cache. Any pending load will be implicitly canceled.
 <a href="#">SetCurrentValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">SetValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)

[Top](#)

## Protected Methods

	Name	Description
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Windows.DependencyObject)

[Top](#)

## See Also

### Reference

[ClientViewSource Class](#)

[C1.Data.DataSource Namespace](#)

### DeferLoad Method

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : DeferLoad Method

Used to group changes to multiple load-affecting properties together, deferring the resulting load operations so a single load operation is performed in the end, that is, when the object returned from this method is disposed.

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> DeferLoad() <b>As</b> System.IDisposable	
C#	
<b>public</b> System.IDisposable DeferLoad()	

### Return Value

An **System.IDisposable** object that will trigger a [Load](#) operation when disposed using the **System.IDisposable.Dispose** method.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### Load Method

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : Load() Method

Starts a load operation. Any pending load will be implicitly canceled.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Load()	
C#	
<b>public void</b> Load()	

## Remarks

If the entities are already in the cache, there will be no roundtrip to the server.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### LoadRange Method

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : LoadRange Method

The index of the first item to load.

The number of entities to load.

If in [virtual mode](#), loads a specific range of entities.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> LoadRange( _ <b>ByVal</b> start <b>As</b> System.Integer, _ <b>ByVal</b> length <b>As</b> System.Integer _ )	
C#	
<b>public void</b> LoadRange( System.int start,	

```
System.int Length
)
```

## Parameters

*start*

The index of the first item to load.

*length*

The number of entities to load.

## Exceptions

Exception	Description
<b>System.InvalidOperationException</b>	<a href="#">VirtualMode</a> is <a href="#">VirtualModeKind.None</a> .

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)  
[ClientViewSource Members](#)

### Refresh Method

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : Refresh() Method

Starts a load operation ignoring the client-side cache. Any pending load will be implicitly canceled.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Refresh()	
C#	
<b>public void</b> Refresh()	

## Remarks

Use this method to refresh data with any changes that may have occurred on the server

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)





[ClientViewSource Members](#)

## Properties













[C1.Data.DataSource Namespace](#) : [ClientViewSource Class](#)







For a list of all members of this type, see [ClientViewSource members](#).

## Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	Gets or sets a value indicating whether <a href="#">Load</a> is automatically invoked on startup and when a change occurs that impacts the query composed by the <a href="#">ClientViewSource</a> . The default is True.
	<a href="#">BaseView</a>	Gets or sets an instance of <a href="#">C1.Data.ClientView&lt;T&gt;</a> that the <a href="#">ClientViewSource</a> uses as the base for composing queries.
	<a href="#">CacheTimeout</a>	Gets or sets the period of time entities loaded in virtual mode are kept in the cache without checking whether they are needed or not. If an entity was neither used nor considered needed for a period of time longer than <a href="#">CacheTimeout</a> , <a href="#">ClientViewSource</a> may evict it from the cache.
	<a href="#">CurrentClientView</a>	Gets the current <a href="#">client view</a> used to load entities, or null in <a href="#">virtual mode</a> .



	<a href="#">DataView</a>	Gets the current view of entities resulting from the last load operation.
	<a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)
	<a href="#">FilterDescriptors</a>	Gets the collection of <a href="#">FilterDescriptor</a> objects used when performing loads.
	<a href="#">FilterOperator</a>	Gets or sets the logical operator used for combining <a href="#">FilterDescriptors</a> in the filter collection. The default value is <a href="#">FilterDescriptorLogicalOperator.And</a> .
	<a href="#">GroupDescriptors</a>	Gets the collection of <a href="#">GroupDescriptor</a> objects used to organize the loaded entities into groups.
	<a href="#">Include</a>	Gets or sets a comma-separated list of property paths that specify related objects to include during the <a href="#">Load</a> operation.
	<a href="#">IsLoadingData</a>	Gets a value indicating whether the <a href="#">ClientViewSource</a> is currently loading data.
	<a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">LoadCommand</a>	Gets an <b>System.Windows.Input.ICommand</b> that invokes <a href="#">Load</a> on this <a href="#">ClientViewSource</a> .
	<a href="#">LoadDelay</a>	Gets or sets the delay before an automatic data loading operation is started. It is the delay from the time a change prompting automatic load occurs until the time the resulting <a href="#">Load</a> is started. The default delay is 25 milliseconds.
	<a href="#">LoadSize</a>	Gets or sets the maximum number of items to load each time a <a href="#">Load</a> is executed. When equal to 0, all requested entities will be loaded. The default is 0.

	<a href="#">MoveToFirstOnLoad</a>	Gets or sets a value indicating that the first item must be made current after <a href="#">Load</a> operation is completed if current item was not set by other means.
	<a href="#">Name</a>	Gets a name of this <a href="#">ClientViewSource</a> to reference it in a <a href="#">C1DataSource.ViewSources</a> collection. By default it is determined by the <a href="#">EntitySetName</a> (for Entity Framework) or <a href="#">QueryName</a> (for RIA Services), but it can be overridden by <a href="#">NameOverride</a> .
	<a href="#">NameOverride</a>	Gets or sets a value that overrides the value of the <a href="#">Name</a> property.
	<a href="#">PageSize</a>	Gets or sets the number of items displayed on each page of the <a href="#">DataView</a> , or the number of items to fetch in each query in <a href="#">virtual mode</a> , or 0 to disable paging.
	<a href="#">SortDescriptors</a>	Gets the collection of <a href="#">SortDescriptor</a> objects used to sort the data.
	<a href="#">VirtualMode</a>	Gets or sets a value indicating whether the <a href="#">ClientViewSource</a> is in virtual mode. Virtual mode is an innovative technology allowing to bind GUI controls directly to very large data sets without delays and performance degradation and without inconvenience of paging. By default, virtual mode is disabled (the default value is <a href="#">VirtualModeKind.None</a> ).

[Top](#)

## See Also

### Reference

[ClientViewSource Class](#)

[C1.Data.DataSource Namespace](#)

### AutoLoad Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : AutoLoad Property

Gets or sets a value indicating whether [Load](#) is automatically invoked on startup and when a change occurs that impacts the query composed by the [ClientViewSource](#). The default is True.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute()&gt; &lt;System.ComponentModel.CategoryAttribute("Common")&gt; Public Property AutoLoad As System.Boolean</pre>	
C#	
<pre>[System.ComponentModel.DefaultValue()] [System.ComponentModel.Category("Common")] public System.bool AutoLoad {get; set;}</pre>	

## Remarks

When [AutoLoad](#) is True, any property change affecting the load query will automatically invoke a [Load](#) after the specified [LoadDelay](#). Examples of properties that impact the query are [PageSize](#) and [FilterOperator](#). Also, changes to dependency object collections like [FilterDescriptors](#) and changes to the dependency properties on elements contained in those collections will affect the query and prompt an automatic [Load](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### BaseView Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : BaseView Property

Gets or sets an instance of [C1.Data.ClientView<T>](#) that the [ClientViewSource](#) uses as the base for composing queries.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute()&gt;</pre>	

<b>Public Property</b> BaseView <b>As</b> View
C#
[System.ComponentModel.DefaultValue()] <b>public</b> View BaseView { <b>get</b> ; <b>set</b> ;}

## Exceptions

Exception	Description
<b>System.ArgumentException</b>	The value is not null and not an instance of <a href="#">C1.Data.ClientView&lt;T&gt;</a> .

## Remarks

The [ClientViewSource](#) applies filtering, sorting, grouping, and paging to its BaseView.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### CacheTimeout Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : CacheTimeout Property

Gets or sets the period of time entities loaded in virtual mode are kept in the cache without checking whether they are needed or not. If an entity was neither used nor considered needed for a period of time longer than [CacheTimeout](#), [ClientViewSource](#) may evict it from the cache.

## Syntax

Visual Basic (Declaration)
<System.ComponentModel.DefaultValueAttribute(> <b>Public Property</b> CacheTimeout <b>As</b> System.TimeSpan

C#	
[System.ComponentModel.DefaultValue()] public System.TimeSpan CacheTimeout {get; set;}	

## Remarks

This property is not used if [VirtualMode](#) is [VirtualModeKind.None](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### CurrentClientView Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : CurrentClientView Property

Gets the current [client view](#) used to load entities, or null in [virtual mode](#).

## Syntax

Visual Basic (Declaration)	
<System.ComponentModel.BrowsableAttribute( <a href="#">False</a> )> Public Property CurrentClientView As View	
C#	
[System.ComponentModel.Browsable( <a href="#">false</a> )] public View CurrentClientView {get; set;}	

## Remarks

Using [CurrentClientView](#), you can build client views on top of the [ClientViewSource](#) by applying live view operators to the [CurrentClientView](#).

The value of this property changes and the [PropertyChanged](#) event is raised whenever the query used to load entities changes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### DataView Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : DataView Property

Gets the current view of entities resulting from the last load operation.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.BrowsableAttribute(False)&gt; &lt;System.ComponentModel.DesignerSerializationVisibilityAttribute(DesignerSerializationVisibility.Hidden)&gt; Public ReadOnly Property DataView As ClientCollectionView</pre>	
C#	
<pre>[System.ComponentModel.Browsable(false)] [System.ComponentModel.DesignerSerializationVisibility(DesignerSerializationV isibility.Hidden)] public ClientCollectionView DataView {get;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

## FilterDescriptors Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : FilterDescriptors Property

Gets the collection of [FilterDescriptor](#) objects used when performing loads.

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.CategoryAttribute("Common")>  
Public ReadOnly Property FilterDescriptors As FilterDescriptorCollection
```

C#

```
[System.ComponentModel.Category("Common")]  
public FilterDescriptorCollection FilterDescriptors {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

## FilterOperator Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : FilterOperator Property

Gets or sets the logical operator used for combining [FilterDescriptors](#) in the filter collection. The default value is [FilterDescriptorLogicalOperator.And](#).

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.CategoryAttribute("Common")>  
Public Property FilterOperator As FilterDescriptorLogicalOperator
```

C#

```
[System.ComponentModel.Category("Common")]
```

```
public FilterDescriptorLogicalOperator FilterOperator {get; set;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### GroupDescriptors Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : GroupDescriptors Property

Gets the collection of [GroupDescriptor](#) objects used to organize the loaded entities into groups.

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.CategoryAttribute("Common")>  
Public ReadOnly Property GroupDescriptors As GroupDescriptorCollection
```

C#

```
[System.ComponentModel.Category("Common")]  
public GroupDescriptorCollection GroupDescriptors {get;}
```

## Remarks

Grouping only works in WPF and Silverlight. It is ignored in WinForms because WinForms data binding does not support grouping.

When a [GroupDescriptor](#) is applied, the data will inherently be sorted by the grouped property. To force a grouped property to be sorted in **descending** order, add a [SortDescriptor](#) to the [SortDescriptors](#) collection for that property using the **Descending** direction.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2



## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### Include Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : Include Property

Gets or sets a comma-separated list of property paths that specify related objects to include during the [Load](#) operation.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.CategoryAttribute("Common")&gt; &lt;System.ComponentModel.DefaultValueAttribute()&gt; Public Property Include As System.String</pre>	
C#	
<pre>[System.ComponentModel.Category("Common")] [System.ComponentModel.DefaultValue()] public System.string Include {get; set;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### IsLoadingData Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : IsLoadingData Property

Gets a value indicating whether the [ClientViewSource](#) is currently loading data.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property IsLoadingData As System.Boolean</code>	
C#	
<code>public System.bool IsLoadingData {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### LoadCommand Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : LoadCommand Property

Gets an **System.Windows.Input.ICommand** that invokes [Load](#) on this [ClientViewSource](#).

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property LoadCommand As System.Windows.Input.ICommand</code>	
C#	
<code>public System.Windows.Input.ICommand LoadCommand {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

## LoadDelay Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : LoadDelay Property

Gets or sets the delay before an automatic data loading operation is started. It is the delay from the time a change prompting automatic load occurs until the time the resulting [Load](#) is started. The default delay is 25 milliseconds.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute(&gt;&gt; Public Property LoadDelay As System.TimeSpan</pre>	
C#	
<pre>[System.ComponentModel.DefaultValue()] public System.TimeSpan LoadDelay {get; set;}</pre>	

## Remarks

Multiple changes that occur within the specified time span are aggregated into a single [Load](#) operation. For every change that occurs, the delay timer is reset. This allows many changes to be combined into a single call as long as each change occurs within the specified delay from the last. Once the delay timer is allowed to elapse without a change occurring, [Load](#) will be invoked.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

## LoadSize Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : LoadSize Property

Gets or sets the maximum number of items to load each time a [Load](#) is executed. When equal to 0, all requested entities will be loaded. The default is 0.

## Syntax

Visual Basic (Declaration)	
<code>&lt;System.ComponentModel.DefaultValueAttribute(&gt;  Public Property LoadSize As System.Integer</code>	
C#	
<code>[System.ComponentModel.DefaultValue()]  public System.int LoadSize {get; set;}</code>	

## Remarks

When [PageSize](#) and [LoadSize](#) are both non-zero, entities will be loaded using the multiple of [PageSize](#) nearest [LoadSize](#). This allows multiple pages to be loaded at once without loading partial pages.

This property is ignored when the [ClientViewSource](#) is in [virtual mode](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### MoveToFirstOnLoad Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : MoveToFirstOnLoad Property

Gets or sets a value indicating that the first item must be made current after [Load](#) operation is completed if current item was not set by other means.

## Syntax

Visual Basic (Declaration)	
<code>&lt;System.ComponentModel.DefaultValueAttribute(&gt;  Public Property MoveToFirstOnLoad As System.Boolean</code>	
C#	
<code>[System.ComponentModel.DefaultValue()]</code>	

```
public System.bool MoveToFirstOnLoad {get; set;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### Name Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : Name Property

Gets a name of this [ClientViewSource](#) to reference it in a [C1DataSource.ViewSources](#) collection. By default it is determined by the [EntitySetName](#) (for Entity Framework) or [QueryName](#) (for RIA Services), but it can be overridden by [NameOverride](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Overridable ReadOnly Property Name As System.String</pre>	
C#	
<pre>public virtual System.string Name {get;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### NameOverride Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : NameOverride Property

Gets or sets a value that overrides the value of the [Name](#) property.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute()&gt; Public Property NameOverride As System.String</pre>	
C#	
<pre>[System.ComponentModel.DefaultValue()] public System.string NameOverride {get; set;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### PageSize Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : PageSize Property

Gets or sets the number of items displayed on each page of the [DataView](#), or the number of items to fetch in each query in [virtual mode](#), or 0 to disable paging.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.CategoryAttribute("Common")&gt; &lt;System.ComponentModel.DefaultValueAttribute()&gt; Public Property PageSize As System.Integer</pre>	
C#	
<pre>[System.ComponentModel.Category("Common")] [System.ComponentModel.DefaultValue()] public System.int PageSize {get; set;}</pre>	

## Remarks

If not in the [virtual mode](#), a non-zero page size will cause the number of entities loaded with each [Load](#) operation to be limited, using server-side paging.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### SortDescriptors Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : SortDescriptors Property

Gets the collection of [SortDescriptor](#) objects used to sort the data.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.CategoryAttribute("Common")&gt; Public ReadOnly Property SortDescriptors As SortDescriptorCollection</pre>	
C#	
<pre>[System.ComponentModel.Category("Common")] public SortDescriptorCollection SortDescriptors {get;}</pre>	

## Remarks

In a [Load](#) operation, the [SortDescriptors](#) are used to perform server-side sorting. The specified sorting is also applied on the client side when changes are made on the client to the loaded entities, with the [DataView](#) reflecting the changes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### VirtualMode Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : VirtualMode Property

Gets or sets a value indicating whether the [ClientViewSource](#) is in virtual mode. Virtual mode is an innovative technology allowing to bind GUI controls directly to very large data sets without delays and performance degradation and without inconvenience of paging. By default, virtual mode is disabled (the default value is [VirtualModeKind.None](#)).

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.DefaultValueAttribute()>  
<System.ComponentModel.CategoryAttribute("Common")>  
Public Property VirtualMode As VirtualModeKind
```

C#

```
[System.ComponentModel.DefaultValue()]  
[System.ComponentModel.Category("Common")]  
public VirtualModeKind VirtualMode {get; set;}
```

## Remarks

[ClientViewSource](#) in virtual mode intelligently and transparently both for the end user and for the developer loads only the entities that need to be displayed on the screen.

To enable virtual mode, set this property to [Managed](#) if you have a control handler with [VirtualMode](#) set to True; otherwise, set it to [VirtualModeKind.Unmanaged](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



### Reference



[ClientViewSource Class](#)  
[ClientViewSource Members](#)  
[VirtualModeKind Enumeration](#)

## Events

>

Name	Description
 <a href="#">LoadedData</a>	Occurs when a load operation is completed, or when an exception was thrown during the load operation.
 <a href="#">PropertyChanged</a>	Occurs when a property value changes.

[Top](#)

## See Also

### Reference

[ClientViewSource Class](#)  
[C1.Data.DataSource Namespace](#)

### LoadedData Event

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : LoadedData Event

Occurs when a load operation is completed, or when an exception was thrown during the load operation.

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> LoadedData <b>As</b> System.EventHandler(Of ClientViewLoadedEventArgs)	
C#	
<b>public event</b> System.EventHandler<ClientViewLoadedEventArgs> LoadedData	

## Event Data

The event handler receives an argument of type [ClientViewLoadedEventArgs](#) containing data related to this event. The following **ClientViewLoadedEventArgs** properties provide information specific to this event.

Property	Description
----------	-------------

<a href="#">Error</a>	Gets the loading error if the loading failed.
<a href="#">HasError</a>	Gets a value indicating whether the loading has failed. If true, inspect the <a href="#">Error</a> property for details.
<a href="#">IsErrorHandled</a>	Gets a value indicating whether the loading error has been marked as handled by calling <a href="#">MarkErrorAsHandled</a> .
<a href="#">Items</a>	Gets all entities loaded by a <a href="#">client view</a> .
<a href="#">TotalItemCount</a>	Gets the total number of rows for the original query without any paging applied to it.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### PropertyChanged Event

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : PropertyChanged Event

Occurs when a property value changes.

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> PropertyChanged <b>As</b> System.ComponentModel.PropertyChangedEventHandler	
C#	
<b>public event</b> System.ComponentModel.PropertyChangedEventHandler PropertyChanged	

## Event Data

The event handler receives an argument of type `System.ComponentModel.PropertyChangedEventArgs` containing data related to this event. The following **PropertyChangedEventArgs** properties provide information specific to this event.

Property	Description
<b>PropertyName</b>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)  
[ClientViewSource Members](#)

## ClientViewSourceException

[C1.Data.DataSource Namespace](#) : ClientViewSourceException Class

This exception indicates that a [ClientViewSource](#) is miconfigured or an error has occurred during the [ClientViewSource.Load](#) operation.

## Object Model

ClientViewSourceException

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.SerializableAttribute(&gt;&gt; Public Class ClientViewSourceException     Inherits System.Exception</pre>	
C#	
<pre>[System.Serializable()] public class ClientViewSourceException : System.Exception</pre>	

## Inheritance Hierarchy

System.Object

System.Exception

**C1.Data.DataSource.ClientViewSourceException**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSourceException Members](#)

[C1.Data.DataSource Namespace](#)

## Overview

[C1.Data.DataSource Namespace](#) : ClientViewSourceException Class

This exception indicates that a [ClientViewSource](#) is misconfigured or an error has occurred during the [ClientViewSource.Load](#) operation.

## Object Model

ClientViewSourceException

## Syntax

Visual Basic (Declaration)

```
<System.SerializableAttribute(>>  
Public Class ClientViewSourceException  
    Inherits System.Exception
```

C#

```
[System.Serializable()]  
public class ClientViewSourceException : System.Exception
```

## Inheritance Hierarchy

System.Object

System.Exception

**C1.Data.DataSource.ClientViewSourceException**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSourceException Members](#)

[C1.Data.DataSource Namespace](#)


## Members

[Properties](#) [Methods](#) [Events](#)

[C1.Data.DataSource Namespace](#) : [ClientViewSourceException Class](#)







The following tables list the members exposed by [ClientViewSourceException](#).



## Public Constructors

	Name	Description
	<a href="#">ClientViewSourceException Constructor</a>	Overloaded.

[Top](#)





## Public Properties

	Name	Description
	<a href="#">Data</a>	(Inherited from System.Exception)
	<a href="#">HelpLink</a>	(Inherited from System.Exception)
	<a href="#">HResult</a>	(Inherited from System.Exception)
	<a href="#">InnerException</a>	(Inherited from System.Exception)
	<a href="#">Message</a>	(Inherited from System.Exception)
	<a href="#">Source</a>	(Inherited from System.Exception)

	<a href="#">StackTrace</a>	(Inherited from System.Exception)
	<a href="#">TargetSite</a>	(Inherited from System.Exception)


[Top](#)

## Public Methods

	Name	Description
	<a href="#">GetBaseException</a>	(Inherited from System.Exception)
	<a href="#">GetObjectData</a>	(Inherited from System.Exception)
	<a href="#">GetType</a>	(Inherited from System.Exception)
	<a href="#">ToString</a>	(Inherited from System.Exception)

[Top](#)

## Protected Events

	Name	Description
	<a href="#">SerializeObjectState</a>	(Inherited from System.Exception)

[Top](#)

## See Also

### Reference

[ClientViewSourceException Class](#)

[C1.Data.DataSource Namespace](#)

## ClientViewSourceException Constructor

[C1.Data.DataSource Namespace](#) > [ClientViewSourceException Class](#) : ClientViewSourceException Constructor

## Overload List

Overload	Description
----------	-------------

<a href="#">ClientViewSourceException Constructor()</a>	Initializes a new instance of the <a href="#">ClientViewSourceException</a> class.
<a href="#">ClientViewSourceException Constructor(String)</a>	Initializes a new instance of the <a href="#">ClientViewSourceException</a> class with a specified error message.
<a href="#">ClientViewSourceException Constructor(String,Exception)</a>	Initializes a new instance of the <a href="#">ClientViewSourceException</a> class with a specified error message, and a reference to the inner exception that is the cause of this exception.
<a href="#">ClientViewSourceException Constructor(SerializationInfo,StreamingContext)</a>	Initializes a new instance of the <a href="#">ClientViewSourceException</a> class with serialized data.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSourceException Class](#)

[ClientViewSourceException Members](#)

## ClientViewSourceException Constructor()

[C1.Data.DataSource Namespace](#) > [ClientViewSourceException Class](#) > [ClientViewSourceException Constructor](#) : ClientViewSourceException Constructor()

Initializes a new instance of the [ClientViewSourceException](#) class.

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public ClientViewSourceException()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSourceException Class](#)

[ClientViewSourceException Members](#)

[Overload List](#)

## ClientViewSourceException Constructor(String)

[C1.Data.DataSource Namespace](#) > [ClientViewSourceException Class](#) > [ClientViewSourceException Constructor](#) : ClientViewSourceException Constructor(String)

The error message that explains the reason for the exception.

Initializes a new instance of the [ClientViewSourceException](#) class with a specified error message.

## Syntax

Visual Basic (Declaration)	
<code>Public Function New( _     ByVal message As System.String _ )</code>	
C#	



```
public ClientViewSourceException(
    System.string message
)
```

## Parameters

*message*

The error message that explains the reason for the exception.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSourceException Class](#)  
[ClientViewSourceException Members](#)  
[Overload List](#)

### ClientViewSourceException Constructor(String,Exception)

[C1.Data.DataSource Namespace](#) > [ClientViewSourceException Class](#) > [ClientViewSourceException Constructor](#) : ClientViewSourceException Constructor(String,Exception)

The error message that explains the reason for the exception.

The exception that is the cause of the current exception.

Initializes a new instance of the [ClientViewSourceException](#) class with a specified error message, and a reference to the inner exception that is the cause of this exception.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function New( _     ByVal message As System.String, _     ByVal inner As System.Exception _ )</pre>	
C#	
<pre>public ClientViewSourceException(</pre>	

```
System.string message,  
System.Exception inner  
)
```

## Parameters

*message*

The error message that explains the reason for the exception.

*inner*

The exception that is the cause of the current exception.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSourceException Class](#)  
[ClientViewSourceException Members](#)  
[Overload List](#)

### ClientViewSourceException Constructor(SerializationInfo,StreamingContext)

[C1.Data.DataSource Namespace](#) > [ClientViewSourceException Class](#) > [ClientViewSourceException Constructor](#) : ClientViewSourceException Constructor(SerializationInfo,StreamingContext)

The **System.Runtime.Serialization.SerializationInfo** object that holds the serialized object data.

The contextual information about the source or destination.

Initializes a new instance of the [ClientViewSourceException](#) class with serialized data.

## Syntax

Visual Basic (Declaration)

```
Protected Function New( _  
    ByVal info As System.Runtime.Serialization.SerializationInfo, _  
    ByVal context As System.Runtime.Serialization.StreamingContext _  
)
```

C#

```
protected ClientViewSourceException(  
    System.Runtime.Serialization.SerializationInfo info,  
    System.Runtime.Serialization.StreamingContext context  
)
```

### Parameters

*info*

The **System.Runtime.Serialization.SerializationInfo** object that holds the serialized object data.

*context*

The contextual information about the source or destination.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSourceException Class](#)  
[ClientViewSourceException Members](#)  
[Overload List](#)

## DependencyObjectCollection<T>

[C1.Data.DataSource Namespace](#) : DependencyObjectCollection<T> Class

The type of objects in the collection. Must be derived from **System.Windows.DependencyObject**.

An observable collection of **dependency objects**.

## Object Model

DependencyObjectCollection<T>

## Syntax

Visual Basic (Declaration)	
<pre>Public Class DependencyObjectCollection(Of T As System.Windows.DependencyObject)     Inherits System.Collections.ObjectModel.ObservableCollection(Of T)</pre>	
C#	
<pre>public class DependencyObjectCollection&lt;T&gt; : System.Collections.ObjectModel.ObservableCollection&lt;T&gt; where T: System.Windows.DependencyObject</pre>	

## Type Parameters

*T*

The type of objects in the collection. Must be derived from **System.Windows.DependencyObject**.

## Inheritance Hierarchy

```
System.Object
  System.Collections.ObjectModel.Collection<T>
    System.Collections.ObjectModel.ObservableCollection<T>
      C1.Data.DataSource.DependencyObjectCollection<T>
        C1.Data.DataSource.FilterDescriptorCollection
        C1.Data.DataSource.GroupDescriptorCollection
        C1.Data.DataSource.SortDescriptorCollection
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DependencyObjectCollection<T> Members](#)  
[C1.Data.DataSource Namespace](#)

### Overview

[C1.Data.DataSource Namespace](#) : [DependencyObjectCollection<T> Class](#)

The type of objects in the collection. Must be derived from **System.Windows.DependencyObject**.

An observable collection of **dependency objects**.

## Object Model

DependencyObjectCollection<T>

## Syntax

Visual Basic (Declaration)

```
Public Class DependencyObjectCollection(Of T As
System.Windows.DependencyObject)
    Inherits System.Collections.ObjectModel.ObservableCollection(Of T)
```

C#

```
public class DependencyObjectCollection<T> :
System.Collections.ObjectModel.ObservableCollection<T>
where T: System.Windows.DependencyObject
```

## Type Parameters

*T*

The type of objects in the collection. Must be derived from **System.Windows.DependencyObject**.

## Inheritance Hierarchy

System.Object

System.Collections.ObjectModel.Collection<T>

System.Collections.ObjectModel.ObservableCollection<T>

**C1.Data.DataSource.DependencyObjectCollection<T>**

[C1.Data.DataSource.FilterDescriptorCollection](#)

[C1.Data.DataSource.GroupDescriptorCollection](#)

[C1.Data.DataSource.SortDescriptorCollection](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

## Members

[Properties](#) [Methods](#) [Events](#)

[C1.Data.DataSource Namespace](#) : [DependencyObjectCollection<T>](#) Class



The following tables list the members exposed by [DependencyObjectCollection<T>](#).

### Public Constructors

	Name	Description
	<a href="#">DependencyObjectCollection&lt;T&gt; Constructor</a>	


[Top](#)

### Public Properties

	Name	Description
	<a href="#">Count</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
	<a href="#">Item</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)



[Top](#)









### Protected Properties

	Name	Description
	<a href="#">Items</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)

[Top](#)







### Public Methods




	Name	Description
	<a href="#">Add</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
	<a href="#">Clear</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)

≡ 	<a href="#">Contains</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
≡ 	<a href="#">CopyTo</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
≡ 	<a href="#">GetEnumerator</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
≡ 	<a href="#">IndexOf</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
≡ 	<a href="#">Insert</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
≡ 	<a href="#">Move</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
≡ 	<a href="#">Remove</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
≡ 	<a href="#">RemoveAt</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)

[Top](#)


## Protected Methods

	Name	Description
	<a href="#">BlockReentrancy</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">CheckReentrancy</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">ClearItems</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">InsertItem</a>	Overridden. Inserts an item into the collection at the specified index.
	<a href="#">MoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">OnCollectionChanged</a>	(Inherited from

		System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">RemoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">SetItem</a>	Overridden. Replaces the element at the specified index.


[Top](#)

## Public Events

	Name	Description
	<a href="#">CollectionChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)

[Top](#)

## Protected Events

	Name	Description
	<a href="#">PropertyChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)

[Top](#)

## See Also

### Reference

[DependencyObjectCollection<T> Class](#)

[C1.Data.DataSource Namespace](#)

## DependencyObjectCollection<T> Constructor

[C1.Data.DataSource Namespace](#) > [DependencyObjectCollection<T> Class](#) :

DependencyObjectCollection<T> Constructor

## Syntax



Visual Basic (Declaration)	
<b>Public Function New()</b>	
C#	
<b>public</b> DependencyObjectCollection<T>()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DependencyObjectCollection<T> Class](#)

[DependencyObjectCollection<T> Members](#)




## Methods

[C1.Data.DataSource Namespace](#) : [DependencyObjectCollection<T> Class](#)

For a list of all members of this type, see [DependencyObjectCollection<T> members](#).










## Public Methods

	Name	Description
⇒	<a href="#">Add</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">Clear</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">Contains</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">CopyTo</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">GetEnumerator</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">IndexOf</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">Insert</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)

	<a href="#">Move</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">Remove</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
	<a href="#">RemoveAt</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)

[Top](#)

## Protected Methods

	Name	Description
	<a href="#">BlockReentrancy</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">CheckReentrancy</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">ClearItems</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">InsertItem</a>	Overridden. Inserts an item into the collection at the specified index.
	<a href="#">MoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">OnCollectionChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">RemoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<T>)
	<a href="#">SetItem</a>	Overridden. Replaces the element at the specified index.

[Top](#)

## See Also

### Reference

[DependencyObjectCollection<T> Class](#)  
[C1.Data.DataSource Namespace](#)

### InsertItem Method

[C1.Data.DataSource Namespace](#) > [DependencyObjectCollection<T> Class](#) : InsertItem Method

The zero-based index at which *item* should be inserted.

The object to insert.

Inserts an item into the collection at the specified index.

## Syntax

Visual Basic (Declaration)

```
Protected Overrides Sub InsertItem( _  
    ByVal index As System.Integer, _  
    ByVal item As T _  
)
```

C#

```
protected override void InsertItem(  
    System.int index,  
    T item  
)
```

### Parameters

*index*

The zero-based index at which *item* should be inserted.

*item*

The object to insert.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DependencyObjectCollection<T> Class](#)

[DependencyObjectCollection<T> Members](#)

### SetItem Method

[C1.Data.DataSource Namespace](#) > [DependencyObjectCollection<T> Class](#) : SetItem Method

The zero-based index of the element to replace.

The new value for the element at the specified index.

Replaces the element at the specified index.

## Syntax

Visual Basic (Declaration)

```
Protected Overrides Sub SetItem( _  
    ByVal index As System.Integer, _  
    ByVal item As T _  
)
```

C#

```
protected override void SetItem(  
    System.int index,  
    T item  
)
```

### Parameters

*index*

The zero-based index of the element to replace.

*item*

The new value for the element at the specified index.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DependencyObjectCollection<T> Class](#)

[DependencyObjectCollection<T> Members](#)

## FilterDescriptor

[C1.Data.DataSource Namespace](#) : FilterDescriptor Class

Descriptor used by the [ClientViewSource](#) to filter data in queries.

## Object Model

FilterDescriptor

## Syntax

Visual Basic (Declaration)

```
Public Class FilterDescriptor
    Inherits System.Windows.DependencyObject
```

C#

```
public class FilterDescriptor : System.Windows.DependencyObject
```

## Inheritance Hierarchy

System.Object

System.Windows.Threading.DispatcherObject

System.Windows.DependencyObject

**C1.Data.DataSource.FilterDescriptor**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptor Members](#)

[C1.Data.DataSource Namespace](#)

# Overview

[C1.Data.DataSource Namespace](#) : FilterDescriptor Class

Descriptor used by the [ClientViewSource](#) to filter data in queries.

# Object Model

FilterDescriptor

# Syntax

Visual Basic (Declaration)	
<pre>Public Class FilterDescriptor     Inherits System.Windows.DependencyObject</pre>	
C#	
<pre>public class FilterDescriptor : System.Windows.DependencyObject</pre>	

# Inheritance Hierarchy

System.Object  
    System.Windows.Threading.DispatcherObject  
        System.Windows.DependencyObject  
            **C1.Data.DataSource.FilterDescriptor**

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[FilterDescriptor Members](#)  
[C1.Data.DataSource Namespace](#)


# Members

[Fields](#) [Properties](#) [Methods](#)

[C1.Data.DataSource Namespace](#) : FilterDescriptor Class







The following tables list the members exposed by [FilterDescriptor](#).

## Public Constructors

	Name	Description
	<a href="#">FilterDescriptor Constructor</a>	Overloaded.





[Top](#)





## Public Fields

	Name	Description
 S	<a href="#">DefaultIgnoredValue</a>	The default value of the <a href="#">IgnoredValue</a> property.
 S	<a href="#">IgnoredValueProperty</a>	The <b>System.Windows.DependencyProperty</b> for the <a href="#">IgnoredValue</a> property.
 S	<a href="#">IsCaseSensitiveProperty</a>	The DependencyProperty for the <a href="#">IsCaseSensitive</a> property.
 S	<a href="#">OperatorProperty</a>	The DependencyProperty for the <a href="#">Operator</a> property.
 S	<a href="#">PropertyPathProperty</a>	The DependencyProperty for the <a href="#">PropertyPath</a> property.
 S	<a href="#">ValueProperty</a>	The DependencyProperty for the <a href="#">Value</a> property.

[Top](#)











## Public Properties

	Name	Description
	<a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)
	<a href="#">IgnoredValue</a>	Gets or sets the value for the right operand for which this filter should be ignored.
	<a href="#">IsCaseSensitive</a>	Gets or sets a value indicating whether the <a href="#">FilterDescriptor</a> is case sensitive for string values.

	<a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Operator</a>	Gets or sets the filter operator.
	<a href="#">PropertyPath</a>	Gets or sets the name of the property path used as the left operand.
	<a href="#">Value</a>	Gets or sets the value of the right operand.

[Top](#)


## Public Methods

	Name	Description
	<a href="#">ClearValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)
	<a href="#">CoerceValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Equals</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetHashCode</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetLocalValueEnumerator</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">InvalidateProperty</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SetCurrentValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SetValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)

[Top](#)

## Protected Methods



	Name	Description
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Windows.DependencyObject)

[Top](#)

## See Also

### Reference

[FilterDescriptor Class](#)

[C1.Data.DataSource Namespace](#)

## FilterDescriptor Constructor

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) : FilterDescriptor Constructor

## Overload List

Overload	Description
<a href="#">FilterDescriptor Constructor(String,FilterOperator,Object)</a>	Initializes a new instance of the <a href="#">FilterDescriptor</a> class with the specified property to use for filtering, the operator to use when evaluating the filtering check, and the filter value.
<a href="#">FilterDescriptor Constructor()</a>	Initializes a new instance of the <a href="#">FilterDescriptor</a> class with default values.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptor Class](#)

[FilterDescriptor Members](#)

## FilterDescriptor Constructor(String,FilterOperator,Object)

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) > [FilterDescriptor Constructor](#) : FilterDescriptor  
Constructor(String,FilterOperator,Object)

The property path to use for filtering.

The kind of comparison to use.

The value to use when filtering.

Initializes a new instance of the [FilterDescriptor](#) class with the specified property to use for filtering, the operator to use when evaluating the filtering check, and the filter value.

## Syntax

Visual Basic (Declaration)

```
Public Function New( _  
    ByVal propertyPath As System.String, _  
    ByVal filterOperator As FilterOperator, _  
    ByVal filterValue As System.Object _  
)
```

C#

```
public FilterDescriptor(  
    System.string propertyPath,  
    FilterOperator filterOperator,  
    System.object filterValue  
)
```

## Parameters

*propertyPath*

The property path to use for filtering.

*filterOperator*

The kind of comparison to use.

*filterValue*

The value to use when filtering.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

- [FilterDescriptor Class](#)
- [FilterDescriptor Members](#)
- [Overload List](#)

### FilterDescriptor Constructor()

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) > [FilterDescriptor Constructor](#) : FilterDescriptor Constructor()

Initializes a new instance of the [FilterDescriptor](#) class with default values.

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public FilterDescriptor()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also









### Reference

- [FilterDescriptor Class](#)
- [FilterDescriptor Members](#)
- [Overload List](#)

### Properties

>

Name	Description
------	-------------

 <a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)
 <a href="#">IgnoredValue</a>	Gets or sets the value for the right operand for which this filter should be ignored.
 <a href="#">IsCaseSensitive</a>	Gets or sets a value indicating whether the <a href="#">FilterDescriptor</a> is case sensitive for string values.
 <a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">Operator</a>	Gets or sets the filter operator.
 <a href="#">PropertyPath</a>	Gets or sets the name of the property path used as the left operand.
 <a href="#">Value</a>	Gets or sets the value of the right operand.

[Top](#)

## See Also

### Reference

[FilterDescriptor Class](#)  
[C1.Data.DataSource Namespace](#)

### IgnoredValue Property

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) : IgnoredValue Property

Gets or sets the value for the right operand for which this filter should be ignored.

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.DefaultValueAttribute(>
Public Property IgnoredValue As System.Object
```

C#

```
[System.ComponentModel.DefaultValue()]
public System.object IgnoredValue {get; set;}
```

## Remarks

If [Value](#) matches [IgnoredValue](#), this filter will not be applied to the load query by the [ClientViewSource](#). The [IgnoredValue](#) is compared to [Value](#) twice in the [ClientViewSource](#). First, it is strictly compared using an **System.Object.Equals(System.Object, System.Object)** comparison. Second, both values are converted to type of the property specified by the [PropertyPath](#) and compared again. If either conversion matches, this filter is ignored.

For example, the following Value/IgnoredValue pairs will all match for an integer property and result in the filter being ignored: 0/0, 0/"0", "0"/"0", and "0"/0.

This property is set to [DefaultIgnoredValue](#) by default. The default value will only match if [Value](#) is also set to [DefaultIgnoredValue](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptor Class](#)

[FilterDescriptor Members](#)

### IsCaseSensitive Property

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) : IsCaseSensitive Property

Gets or sets a value indicating whether the [FilterDescriptor](#) is case sensitive for string values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute(&gt; Public Property IsCaseSensitive As System.Boolean</pre>	
C#	
<pre>[System.ComponentModel.DefaultValue()] public System.bool IsCaseSensitive {get; set;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptor Class](#)

[FilterDescriptor Members](#)

### Operator Property

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) : Operator Property

Gets or sets the filter operator.

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.DefaultValueAttribute(>>  
Public Property Operator As FilterOperator
```

C#

```
[System.ComponentModel.DefaultValue()]  
public FilterOperator Operator {get; set;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptor Class](#)

[FilterDescriptor Members](#)

### PropertyPath Property

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) : PropertyPath Property

Gets or sets the name of the property path used as the left operand.

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.DefaultValueAttribute(>>
```

<b>Public Property</b> PropertyPath <b>As</b> System.String	
C#	
[System.ComponentModel.DefaultValue()] <b>public</b> System. <b>string</b> PropertyPath { <b>get</b> ; <b>set</b> ;} 	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptor Class](#)

[FilterDescriptor Members](#)

### Value Property

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) : Value Property

Gets or sets the value of the right operand.

## Syntax

Visual Basic (Declaration)	
<System.ComponentModel.DefaultValueAttribute(> <b>Public Property</b> Value <b>As</b> System.Object 	
C#	
[System.ComponentModel.DefaultValue()] <b>public</b> System. <b>object</b> Value { <b>get</b> ; <b>set</b> ;} 	

## Remarks

This will be used by the [ClientViewSource](#) to compose a filter for the load query. It will be applied following the pattern [Entity].[PropertyPath] [Operator] [Value]. For example, a query might look like Customer.Name == "CurrentCustomerName".

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also







### Reference

[FilterDescriptor Class](#)

[FilterDescriptor Members](#)

### Fields

>

Name	Description
 <b>S</b> <a href="#">DefaultIgnoredValue</a>	The default value of the <a href="#">IgnoredValue</a> property.
 <b>S</b> <a href="#">IgnoredValueProperty</a>	The <b>System.Windows.DependencyProperty</b> for the <a href="#">IgnoredValue</a> property.
 <b>S</b> <a href="#">IsCaseSensitiveProperty</a>	The DependencyProperty for the <a href="#">IsCaseSensitive</a> property.
 <b>S</b> <a href="#">OperatorProperty</a>	The DependencyProperty for the <a href="#">Operator</a> property.
 <b>S</b> <a href="#">PropertyPathProperty</a>	The DependencyProperty for the <a href="#">PropertyPath</a> property.
 <b>S</b> <a href="#">ValueProperty</a>	The DependencyProperty for the <a href="#">Value</a> property.

[Top](#)

## See Also

### Reference

[FilterDescriptor Class](#)

[C1.Data.DataSource Namespace](#)

### DefaultIgnoredValue Field

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) : DefaultIgnoredValue Field

The default value of the [IgnoredValue](#) property.

## Syntax

Visual Basic (Declaration)



<code>Public Shared ReadOnly DefaultIgnoredValue As System.Object</code>	
C#	
<code>public static readonly System.object DefaultIgnoredValue</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptor Class](#)

[FilterDescriptor Members](#)

### IgnoredValueProperty Field

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) : IgnoredValueProperty Field

The **System.Windows.DependencyProperty** for the [IgnoredValue](#) property.

## Syntax

Visual Basic (Declaration)	
<code>Public Shared ReadOnly IgnoredValueProperty As System.Windows.DependencyProperty</code>	
C#	
<code>public static readonly System.Windows.DependencyProperty IgnoredValueProperty</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptor Class](#)

[FilterDescriptor Members](#)

## IsCaseSensitiveProperty Field

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) : IsCaseSensitiveProperty Field

The DependencyProperty for the [IsCaseSensitive](#) property.

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared ReadOnly IsCaseSensitiveProperty As System.Windows.DependencyProperty</pre>	
C#	
<pre>public static readonly System.Windows.DependencyProperty IsCaseSensitiveProperty</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptor Class](#)

[FilterDescriptor Members](#)

## OperatorProperty Field

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) : OperatorProperty Field

The DependencyProperty for the [Operator](#) property.

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared ReadOnly OperatorProperty As System.Windows.DependencyProperty</pre>	
C#	
<pre>public static readonly System.Windows.DependencyProperty OperatorProperty</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[FilterDescriptor Class](#)  
[FilterDescriptor Members](#)

PropertyPathProperty Field

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) : PropertyPathProperty Field

The DependencyProperty for the [PropertyPath](#) property.

Syntax

Visual Basic (Declaration)	
<pre>Public Shared ReadOnly PropertyPathProperty As System.Windows.DependencyProperty</pre>	
C#	
<pre>public static readonly System.Windows.DependencyProperty PropertyPathProperty</pre>	

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[FilterDescriptor Class](#)  
[FilterDescriptor Members](#)

ValueProperty Field

[C1.Data.DataSource Namespace](#) > [FilterDescriptor Class](#) : ValueProperty Field

The DependencyProperty for the [Value](#) property.

Syntax

Visual Basic (Declaration)	
<code>Public Shared ReadOnly ValueProperty As System.Windows.DependencyProperty</code>	
C#	
<code>public static readonly System.Windows.DependencyProperty ValueProperty</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptor Class](#)

[FilterDescriptor Members](#)

## FilterDescriptorCollection

[C1.Data.DataSource Namespace](#) : FilterDescriptorCollection Class

Collection of [FilterDescriptor](#) dependency objects.

## Object Model

FilterDescriptorCollection

## Syntax

Visual Basic (Declaration)	
<pre>Public Class FilterDescriptorCollection     Inherits C1.Data.DataSource.DependencyObjectCollection(Of FilterDescriptor)</pre>	
C#	
<pre>public class FilterDescriptorCollection : C1.Data.DataSource.DependencyObjectCollection&lt;FilterDescriptor&gt;</pre>	

## Inheritance Hierarchy

System.Object  
System.Collections.ObjectModel.Collection<T>  
System.Collections.ObjectModel.ObservableCollection<T>  
[C1.Data.DataSource.DependencyObjectCollection<T>](#)  
**C1.Data.DataSource.FilterDescriptorCollection**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptorCollection Members](#)  
[C1.Data.DataSource Namespace](#)

## Overview

[C1.Data.DataSource Namespace](#) : FilterDescriptorCollection Class

Collection of [FilterDescriptor](#) dependency objects.

## Object Model

FilterDescriptorCollection

## Syntax

Visual Basic (Declaration)

```
Public Class FilterDescriptorCollection
    Inherits C1.Data.DataSource.DependencyObjectCollection(Of
FilterDescriptor)
```

C#

```
public class FilterDescriptorCollection :
C1.Data.DataSource.DependencyObjectCollection<FilterDescriptor>
```

## Inheritance Hierarchy

System.Object  
System.Collections.ObjectModel.Collection<T>  
System.Collections.ObjectModel.ObservableCollection<T>

[C1.Data.DataSource.DependencyObjectCollection<T>](#)

## **C1.Data.DataSource.FilterDescriptorCollection**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptorCollection Members](#)

[C1.Data.DataSource Namespace](#)


## Members

[Properties](#) [Methods](#) [Events](#)

[C1.Data.DataSource Namespace](#) : [FilterDescriptorCollection Class](#)



The following tables list the members exposed by [FilterDescriptorCollection](#).

## Public Constructors

	Name	Description
	<a href="#">FilterDescriptorCollection Constructor</a>	Initializes a new instance of the <a href="#">FilterDescriptorCollection</a> class.


[Top](#)

## Public Properties

	Name	Description
	<a href="#">Count</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;FilterDescriptor&gt;</a> )
	<a href="#">Item</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;FilterDescriptor&gt;</a> )











[Top](#)

## Protected Properties

	Name	Description
	Items	(Inherited from System.Collections.ObjectModel.Collection<FilterDescriptor>)

[Top](#)







## Public Methods

	Name	Description
	Add	(Inherited from System.Collections.ObjectModel.Collection<FilterDescriptor>)
	Clear	(Inherited from System.Collections.ObjectModel.Collection<FilterDescriptor>)
	Contains	(Inherited from System.Collections.ObjectModel.Collection<FilterDescriptor>)
	CopyTo	(Inherited from System.Collections.ObjectModel.Collection<FilterDescriptor>)
	GetEnumerator	(Inherited from System.Collections.ObjectModel.Collection<FilterDescriptor>)
	IndexOf	(Inherited from System.Collections.ObjectModel.Collection<FilterDescriptor>)
	Insert	(Inherited from System.Collections.ObjectModel.Collection<FilterDescriptor>)
	Move	(Inherited from System.Collections.ObjectModel.ObservableCollection<FilterDescriptor>)
	Remove	(Inherited from System.Collections.ObjectModel.Collection<FilterDescriptor>)
	RemoveAt	(Inherited from


		System.Collections.ObjectModel.Collection<FilterDescriptor>)
--	--	--

[Top](#)

## Protected Methods


	Name	Description
	<a href="#">BlockReentrancy</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<FilterDescriptor>)
	<a href="#">CheckReentrancy</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<FilterDescriptor>)
	<a href="#">ClearItems</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<FilterDescriptor>)
	<a href="#">InsertItem</a>	Inserts an item into the collection at the specified index. (Inherited from <a href="#">C1.Data.DataSource.DependencyObjectCollection&lt;FilterDescriptor&gt;</a> )
	<a href="#">MoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<FilterDescriptor>)
	<a href="#">OnCollectionChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<FilterDescriptor>)
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<FilterDescriptor>)
	<a href="#">RemoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<FilterDescriptor>)



		or>)
	<a href="#">SetItem</a>	Replaces the element at the specified index. (Inherited from <a href="#">C1.Data.DataSource.DependencyObjectCollection&lt;FilterDescriptor&gt;</a> )


[Top](#)

## Public Events

	Name	Description
	<a href="#">CollectionChange</a> <a href="#">d</a>	(Inherited from <a href="#">System.Collections.ObjectModel.ObservableCollection&lt;FilterDescriptor&gt;</a> )

[Top](#)

## Protected Events

	Name	Description
	<a href="#">PropertyChange</a> <a href="#">d</a>	(Inherited from <a href="#">System.Collections.ObjectModel.ObservableCollection&lt;FilterDescriptor&gt;</a> )

[Top](#)

## See Also

### Reference

[FilterDescriptorCollection Class](#)  
[C1.Data.DataSource Namespace](#)

## FilterDescriptorCollection Constructor

[C1.Data.DataSource Namespace](#) > [FilterDescriptorCollection Class](#) : FilterDescriptorCollection Constructor

Initializes a new instance of the [FilterDescriptorCollection](#) class.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

Public Function New()
-----------------------

C#
----

public FilterDescriptorCollection()
-------------------------------------

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilterDescriptorCollection Class](#)

[FilterDescriptorCollection Members](#)

## GroupDescriptor

[C1.Data.DataSource Namespace](#) : GroupDescriptor Class

Descriptor used by the [ClientViewSource](#) to group data returned from server-side queries.

## Object Model

GroupDescriptor

## Syntax

Visual Basic (Declaration)
----------------------------

<pre>Public Class GroupDescriptor     Inherits System.Windows.DependencyObject</pre>
--

C#
----

<pre>public class GroupDescriptor : System.Windows.DependencyObject</pre>
---

## Inheritance Hierarchy

```
System.Object
  System.Windows.Threading.DispatcherObject
    System.Windows.DependencyObject
      C1.Data.DataSource.GroupDescriptor
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupDescriptor Members](#)  
[C1.Data.DataSource Namespace](#)

## Overview

[C1.Data.DataSource Namespace](#) : GroupDescriptor Class

Descriptor used by the [ClientViewSource](#) to group data returned from server-side queries.

## Object Model

GroupDescriptor

## Syntax

Visual Basic (Declaration)	
<pre>Public Class GroupDescriptor     Inherits System.Windows.DependencyObject</pre>	
C#	
<pre>public class GroupDescriptor : System.Windows.DependencyObject</pre>	

## Inheritance Hierarchy

System.Object  
    System.Windows.Threading.DispatcherObject  
        System.Windows.DependencyObject  
            **C1.Data.DataSource.GroupDescriptor**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[GroupDescriptor Members](#)

[C1.Data.DataSource Namespace](#)


## Members

[Fields](#) [Properties](#) [Methods](#)

[C1.Data.DataSource Namespace](#) : [GroupDescriptor Class](#)


The following tables list the members exposed by [GroupDescriptor](#).

### Public Constructors

	Name	Description
	<a href="#">GroupDescriptor Constructor</a>	Overloaded.





[Top](#)

### Public Fields

	Name	Description
	<a href="#">PropertyPathProperty</a>	The <a href="#">DependencyProperty</a> for the <a href="#">PropertyPath</a> property.











[Top](#)

### Public Properties

	Name	Description
	<a href="#">DependencyObjectType</a>	(Inherited from <a href="#">System.Windows.DependencyObject</a> )
	<a href="#">Dispatcher</a>	(Inherited from <a href="#">System.Windows.Threading.DispatcherObject</a> )
	<a href="#">IsSealed</a>	(Inherited from <a href="#">System.Windows.DependencyObject</a> )
	<a href="#">PropertyPath</a>	Gets or sets the name of the property path used to group data.


[Top](#)

### Public Methods

	Name	Description
	<a href="#">ClearValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)
	<a href="#">CoerceValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Equals</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetHashCode</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetLocalValueEnumerator</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">InvalidateProperty</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SetCurrentValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SetValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)

[Top](#)

## Protected Methods

	Name	Description
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Windows.DependencyObject)

[Top](#)

## See Also

### Reference

[GroupDescriptor Class](#)  
[C1.Data.DataSource Namespace](#)

## GroupDescriptor Constructor

[C1.Data.DataSource Namespace](#) > [GroupDescriptor Class](#) : GroupDescriptor Constructor

### Overload List

Overload	Description
<a href="#">GroupDescriptor Constructor()</a>	Initializes a new instance of the <a href="#">GroupDescriptor</a> class.
<a href="#">GroupDescriptor Constructor(String)</a>	Initializes a new instance of the <a href="#">GroupDescriptor</a> class.

### Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

### See Also

#### Reference

[GroupDescriptor Class](#)

[GroupDescriptor Members](#)

#### GroupDescriptor Constructor()

[C1.Data.DataSource Namespace](#) > [GroupDescriptor Class](#) > [GroupDescriptor Constructor](#) :  
GroupDescriptor Constructor()

Initializes a new instance of the [GroupDescriptor](#) class.

### Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public GroupDescriptor()</code>	

### Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupDescriptor Class](#)

[GroupDescriptor Members](#)

[Overload List](#)

### GroupDescriptor Constructor(String)

[C1.Data.DataSource Namespace](#) > [GroupDescriptor Class](#) > [GroupDescriptor Constructor](#) :

GroupDescriptor Constructor(String)

The group property path

Initializes a new instance of the [GroupDescriptor](#) class.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function New( _     ByVal propertyPath As System.String _ )</pre>	
C#	
<pre>public GroupDescriptor(     System.string propertyPath )</pre>	

### Parameters

*propertyPath*

The group property path

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2





## See Also

### Reference

[GroupDescriptor Class](#)  
[GroupDescriptor Members](#)  
[Overload List](#)

## Properties

>

Name	Description
 <a href="#">DependencyObjectType</a> (Inherited from System.Windows.DependencyObject)	
 <a href="#">Dispatcher</a> (Inherited from System.Windows.Threading.DispatcherObject)	
 <a href="#">IsSealed</a> (Inherited from System.Windows.DependencyObject)	
 <a href="#">PropertyPath</a> Gets or sets the name of the property path used to group data.	

[Top](#)

## See Also

### Reference

[GroupDescriptor Class](#)  
[C1.Data.DataSource Namespace](#)

### PropertyPath Property

[C1.Data.DataSource Namespace](#) > [GroupDescriptor Class](#) : PropertyPath Property

Gets or sets the name of the property path used to group data.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute(&gt;&gt; Public Property PropertyPath As System.String</pre>	
C#	
<pre>[System.ComponentModel.DefaultValue()] public System.string PropertyPath {get; set;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2



# See Also

## Reference


[GroupDescriptor Class](#)  
[GroupDescriptor Members](#)

## Fields

[C1.Data.DataSource Namespace](#) : [GroupDescriptor Class](#)

For a list of all members of this type, see [GroupDescriptor members](#).

# Public Fields

	Name	Description
 <b>S</b>	<a href="#">PropertyPathProperty</a>	The <a href="#">DependencyProperty</a> for the <a href="#">PropertyPath</a> property.

[Top](#)

# See Also

## Reference

[GroupDescriptor Class](#)  
[C1.Data.DataSource Namespace](#)

## PropertyPathProperty Field

[C1.Data.DataSource Namespace](#) > [GroupDescriptor Class](#) : [PropertyPathProperty Field](#)

The [DependencyProperty](#) for the [PropertyPath](#) property.

# Syntax

Visual Basic (Declaration)	
<b>Public Shared ReadOnly</b> <a href="#">PropertyPathProperty</a> <b>As</b> <a href="#">System.Windows.DependencyProperty</a>	
C#	
<b>public static readonly</b> <a href="#">System.Windows.DependencyProperty</a> <a href="#">PropertyPathProperty</a>	

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupDescriptor Class](#)  
[GroupDescriptor Members](#)

## GroupDescriptorCollection

[C1.Data.DataSource Namespace](#) : GroupDescriptorCollection Class

Collection of [GroupDescriptor](#) dependency objects.

## Object Model

GroupDescriptorCollection

## Syntax

Visual Basic (Declaration)	
<pre>Public Class GroupDescriptorCollection     Inherits C1.Data.DataSource.DependencyObjectCollection(Of GroupDescriptor)</pre>	
C#	
<pre>public class GroupDescriptorCollection :     C1.Data.DataSource.DependencyObjectCollection&lt;GroupDescriptor&gt;</pre>	

## Inheritance Hierarchy

System.Object  
  System.Collections.ObjectModel.Collection<T>  
    System.Collections.ObjectModel.ObservableCollection<T>  
      [C1.Data.DataSource.DependencyObjectCollection<T>](#)  
        **C1.Data.DataSource.GroupDescriptorCollection**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[GroupDescriptorCollection Members](#)  
[C1.Data.DataSource Namespace](#)

## Overview

[C1.Data.DataSource Namespace](#) : GroupDescriptorCollection Class

Collection of [GroupDescriptor](#) dependency objects.

# Object Model

GroupDescriptorCollection

## Syntax

Visual Basic (Declaration)	
<pre>Public Class GroupDescriptorCollection     Inherits C1.Data.DataSource.DependencyObjectCollection(Of GroupDescriptor)</pre>	
C#	
<pre>public class GroupDescriptorCollection :     C1.Data.DataSource.DependencyObjectCollection&lt;GroupDescriptor&gt;</pre>	

## Inheritance Hierarchy

System.Object  
    System.Collections.ObjectModel.Collection<T>  
        System.Collections.ObjectModel.ObservableCollection<T>  
            [C1.Data.DataSource.DependencyObjectCollection<T>](#)  
                **C1.Data.DataSource.GroupDescriptorCollection**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference


## Members

[Properties](#) [Methods](#) [Events](#)

[C1.Data.DataSource Namespace](#) : GroupDescriptorCollection Class



The following tables list the members exposed by [GroupDescriptorCollection](#).

### Public Constructors

	Name	Description
	<a href="#">GroupDescriptorCollection Constructor</a>	Initializes a new instance of the <a href="#">GroupDescriptorCollection</a> class.


[Top](#)

### Public Properties

	Name	Description
	<a href="#">Count</a>	(Inherited from System.Collections.ObjectModel.Collection<GroupDescriptor>)
	<a href="#">Item</a>	(Inherited from System.Collections.ObjectModel.Collection<GroupDescriptor>)

[Top](#)











### Protected Properties

	Name	Description
	<a href="#">Items</a>	(Inherited from System.Collections.ObjectModel.Collection<GroupDescriptor>)

[Top](#)


### Public Methods









	Name	Description
--	------	-------------

	<a href="#">Add</a>	(Inherited from System.Collections.ObjectModel.Collection<GroupDescriptor>)
	<a href="#">Clear</a>	(Inherited from System.Collections.ObjectModel.Collection<GroupDescriptor>)
	<a href="#">Contains</a>	(Inherited from System.Collections.ObjectModel.Collection<GroupDescriptor>)
	<a href="#">CopyTo</a>	(Inherited from System.Collections.ObjectModel.Collection<GroupDescriptor>)
	<a href="#">GetEnumerator</a>	(Inherited from System.Collections.ObjectModel.Collection<GroupDescriptor>)
	<a href="#">IndexOf</a>	(Inherited from System.Collections.ObjectModel.Collection<GroupDescriptor>)
	<a href="#">Insert</a>	(Inherited from System.Collections.ObjectModel.Collection<GroupDescriptor>)
	<a href="#">Move</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<GroupDescriptor>)
	<a href="#">Remove</a>	(Inherited from System.Collections.ObjectModel.Collection<GroupDescriptor>)
	<a href="#">RemoveAt</a>	(Inherited from System.Collections.ObjectModel.Collection<GroupDescriptor>)

[Top](#)


## Protected Methods

	Name	Description
	<a href="#">BlockReentrancy</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<GroupDescrip

		tor>)
	<a href="#">CheckReentrancy</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<GroupDescriptor>)
	<a href="#">ClearItems</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<GroupDescriptor>)
	<a href="#">InsertItem</a>	Inserts an item into the collection at the specified index. (Inherited from <a href="#">C1.Data.DataSource.DependencyObjectCollection&lt;GroupDescriptor&gt;</a> )
	<a href="#">MoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<GroupDescriptor>)
	<a href="#">OnCollectionChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<GroupDescriptor>)
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<GroupDescriptor>)
	<a href="#">RemoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<GroupDescriptor>)
	<a href="#">SetItem</a>	Replaces the element at the specified index. (Inherited from <a href="#">C1.Data.DataSource.DependencyObjectCollection&lt;GroupDescriptor&gt;</a> )


[Top](#)

## Public Events

	Name	Description
	<a href="#">CollectionChange</a> <a href="#">d</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<GroupDescriptor>)

[Top](#)

## Protected Events

	Name	Description
	<a href="#">PropertyChange</a> <a href="#">d</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<GroupDescriptor>)

[Top](#)

## See Also

### Reference

[GroupDescriptorCollection Class](#)  
[C1.Data.DataSource Namespace](#)

## GroupDescriptorCollection Constructor

[C1.Data.DataSource Namespace](#) > [GroupDescriptorCollection Class](#) : GroupDescriptorCollection  
 Constructor

Initializes a new instance of the [GroupDescriptorCollection](#) class.

## Syntax

Visual Basic (Declaration)	
<b>Public Function New()</b>	
C#	
<b>public</b> GroupDescriptorCollection()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[GroupDescriptorCollection Class](#)  
[GroupDescriptorCollection Members](#)

SortDescriptor

[C1.Data.DataSource Namespace](#) : SortDescriptor Class

Descriptor used by the [ClientViewSource](#) to sort data returned from queries.

Object Model

SortDescriptor

Syntax

Visual Basic (Declaration)	
<pre>Public Class SortDescriptor     Inherits System.Windows.DependencyObject</pre>	
C#	
<pre>public class SortDescriptor : System.Windows.DependencyObject</pre>	

Inheritance Hierarchy

System.Object  
    System.Windows.Threading.DispatcherObject  
        System.Windows.DependencyObject  
            **C1.Data.DataSource.SortDescriptor**

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also



## Reference

[SortDescriptor Members](#)

[C1.Data.DataSource Namespace](#)

## Overview

[C1.Data.DataSource Namespace](#) : SortDescriptor Class

Descriptor used by the [ClientViewSource](#) to sort data returned from queries.

## Object Model

SortDescriptor

## Syntax

Visual Basic (Declaration)

```
Public Class SortDescriptor
    Inherits System.Windows.DependencyObject
```

C#

```
public class SortDescriptor : System.Windows.DependencyObject
```

## Inheritance Hierarchy

System.Object

System.Windows.Threading.DispatcherObject

System.Windows.DependencyObject

**C1.Data.DataSource.SortDescriptor**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SortDescriptor Members](#)

[C1.Data.DataSource Namespace](#)


## Members

[Fields](#) [Properties](#) [Methods](#)

[C1.Data.DataSource Namespace](#) : SortDescriptor Class



The following tables list the members exposed by [SortDescriptor](#).

## Public Constructors

	Name	Description
	<a href="#">SortDescriptor Constructor</a>	Overloaded.






[Top](#)

## Public Fields

	Name	Description
 S	<a href="#">DirectionProperty</a>	The DependencyProperty for the <a href="#">Direction</a> property.
 S	<a href="#">PropertyPathProperty</a>	The DependencyProperty for the <a href="#">PropertyPath</a> property.

[Top](#)











## Public Properties

	Name	Description
	<a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Direction</a>	Gets or sets the sort direction: Ascending or Descending.
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)
	<a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">PropertyPath</a>	Gets or sets the name of the property path used to sort data.

[Top](#)


## Public Methods

	Name	Description
--	------	-------------

	<a href="#">ClearValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)
	<a href="#">CoerceValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Equals</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetHashCode</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetLocalValueEnumerator</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">InvalidateProperty</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SetCurrentValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SetValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)

[Top](#)

## Protected Methods

	Name	Description
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Windows.DependencyObject)

[Top](#)

## See Also

### Reference

[SortDescriptor Class](#)

[C1.Data.DataSource Namespace](#)

## SortDescriptor Constructor

[C1.Data.DataSource Namespace](#) > [SortDescriptor Class](#) : SortDescriptor Constructor

## Overload List

Overload	Description
<a href="#">SortDescriptor Constructor()</a>	Initializes a new instance of the <a href="#">SortDescriptor</a> class.
<a href="#">SortDescriptor Constructor(String,ListSortDirection)</a>	Initializes a new instance of the <a href="#">SortDescriptor</a> class.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SortDescriptor Class](#)

[SortDescriptor Members](#)

### SortDescriptor Constructor()

[C1.Data.DataSource Namespace](#) > [SortDescriptor Class](#) > [SortDescriptor Constructor](#) : SortDescriptor Constructor()

Initializes a new instance of the [SortDescriptor](#) class.

## Syntax

Visual Basic (Declaration)	
<b>Public Function New()</b>	
C#	
<b>public</b> SortDescriptor()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SortDescriptor Class](#)

[SortDescriptor Members](#)

[Overload List](#)

### SortDescriptor Constructor(String,ListSortDirection)

[C1.Data.DataSource Namespace](#) > [SortDescriptor Class](#) > [SortDescriptor Constructor](#) : SortDescriptor  
Constructor(String,ListSortDirection)

The sort property path

The sort direction

Initializes a new instance of the [SortDescriptor](#) class.

## Syntax

Visual Basic (Declaration)

```
Public Function New( _  
    ByVal propertyPath As System.String, _  
    ByVal direction As System.ComponentModel.ListSortDirection _  
)
```

C#

```
public SortDescriptor(  
    System.string propertyPath,  
    System.ComponentModel.ListSortDirection direction  
)
```

### Parameters

*propertyPath*

The sort property path

*direction*

The sort direction

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2






# See Also

## Reference

- [SortDescriptor Class](#)
- [SortDescriptor Members](#)
- [Overload List](#)

## Properties

>

Name	Description
 <a href="#">DependencyObjectType</a> (Inherited from System.Windows.DependencyObject)	
 <a href="#">Direction</a>	Gets or sets the sort direction: Ascending or Descending.
 <a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)
 <a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">PropertyPath</a>	Gets or sets the name of the property path used to sort data.

[Top](#)

# See Also

## Reference

- [SortDescriptor Class](#)
- [C1.Data.DataSource Namespace](#)

## Direction Property

[C1.Data.DataSource Namespace](#) > [SortDescriptor Class](#) : Direction Property

Gets or sets the sort direction: Ascending or Descending.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute(&gt;&gt; <b>Public Property</b> Direction <b>As</b> System.ComponentModel.ListSortDirection</pre>	
C#	
<pre>[System.ComponentModel.DefaultValue()]</pre>	

```
public System.ComponentModel.ListSortDirection Direction {get; set;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SortDescriptor Class](#)

[SortDescriptor Members](#)

### PropertyPath Property

[C1.Data.DataSource Namespace](#) > [SortDescriptor Class](#) : PropertyPath Property

Gets or sets the name of the property path used to sort data.

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.DefaultValueAttribute(>>  
Public Property PropertyPath As System.String
```

C#

```
[System.ComponentModel.DefaultValue()]  
public System.string PropertyPath {get; set;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SortDescriptor Class](#)



[SortDescriptor Members](#)

# Fields

[C1.Data.DataSource Namespace](#) : SortDescriptor Class

For a list of all members of this type, see [SortDescriptor members](#).

## Public Fields

	Name	Description
 <b>S</b>	<a href="#">DirectionProperty</a>	The DependencyProperty for the <a href="#">Direction</a> property.
 <b>S</b>	<a href="#">PropertyPathProperty</a>	The DependencyProperty for the <a href="#">PropertyPath</a> property.

[Top](#)

## See Also

### Reference

[SortDescriptor Class](#)  
[C1.Data.DataSource Namespace](#)

### DirectionProperty Field

[C1.Data.DataSource Namespace](#) > [SortDescriptor Class](#) : DirectionProperty Field

The DependencyProperty for the [Direction](#) property.

## Syntax

Visual Basic (Declaration)	
<code>Public Shared ReadOnly DirectionProperty As System.Windows.DependencyProperty</code>	
C#	
<code>public static readonly System.Windows.DependencyProperty DirectionProperty</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



[SortDescriptor Class](#)  
[SortDescriptor Members](#)

## PropertyPathProperty Field

[C1.Data.DataSource Namespace](#) > [SortDescriptor Class](#) : PropertyPathProperty Field

The DependencyProperty for the [PropertyPath](#) property.

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared ReadOnly PropertyPathProperty As System.Windows.DependencyProperty</pre>	
C#	
<pre>public static readonly System.Windows.DependencyProperty PropertyPathProperty</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

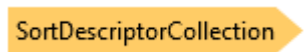
[SortDescriptor Class](#)  
[SortDescriptor Members](#)

## SortDescriptorCollection

[C1.Data.DataSource Namespace](#) : SortDescriptorCollection Class

Collection of [SortDescriptor](#) dependency objects.

## Object Model



## Syntax

Visual Basic (Declaration)	
<pre>Public Class SortDescriptorCollection</pre>	

<a href="#">Inherits C1.Data.DataSource.DependencyObjectCollection(Of SortDescriptor)</a>	
C#	
<pre>public class SortDescriptorCollection : C1.Data.DataSource.DependencyObjectCollection&lt;SortDescriptor&gt;</pre>	

## Inheritance Hierarchy

```
System.Object
  System.Collections.ObjectModel.Collection<T>
    System.Collections.ObjectModel.ObservableCollection<T>
      C1.Data.DataSource.DependencyObjectCollection<T>
        C1.Data.DataSource.SortDescriptorCollection
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SortDescriptorCollection Members](#)  
[C1.Data.DataSource Namespace](#)

### Overview

[C1.Data.DataSource Namespace](#) : SortDescriptorCollection Class

Collection of [SortDescriptor](#) dependency objects.

## Object Model

[SortDescriptorCollection](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Class SortDescriptorCollection   Inherits C1.Data.DataSource.DependencyObjectCollection(Of SortDescriptor)</pre>	
C#	

```
public class SortDescriptorCollection :  
C1.Data.DataSource.DependencyObjectCollection<SortDescriptor>
```

Inheritance Hierarchy

System.Object  
System.Collections.ObjectModel.Collection<T>  
System.Collections.ObjectModel.ObservableCollection<T>  
C1.Data.DataSource.DependencyObjectCollection<T>  
C1.Data.DataSource.SortDescriptorCollection

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

SortDescriptorCollection Members  
C1.Data.DataSource Namespace

Members

Properties Methods Events

C1.Data.DataSource Namespace : SortDescriptorCollection Class


The following tables list the members exposed by SortDescriptorCollection.


Public Constructors

	Name	Description
	<a href="#">SortDescriptorCollection Constructor</a>	Initializes a new instance of the <a href="#">SortDescriptorCollection</a> class.

[Top](#)


Public Properties

	Name	Description
	<a href="#">Count</a>	(Inherited from

		System.Collections.ObjectModel.Collection<SortDescriptor>)
	<a href="#">Item</a>	(Inherited from System.Collections.ObjectModel.Collection<SortDescriptor>)








[Top](#)

## Protected Properties

	Name	Description
	<a href="#">Items</a>	(Inherited from System.Collections.ObjectModel.Collection<SortDescriptor>)

[Top](#)

## Public Methods




	Name	Description
	<a href="#">Add</a>	(Inherited from System.Collections.ObjectModel.Collection<SortDescriptor>)
	<a href="#">Clear</a>	(Inherited from System.Collections.ObjectModel.Collection<SortDescriptor>)
	<a href="#">Contains</a>	(Inherited from System.Collections.ObjectModel.Collection<SortDescriptor>)
	<a href="#">CopyTo</a>	(Inherited from System.Collections.ObjectModel.Collection<SortDescriptor>)
	<a href="#">GetEnumerator</a>	(Inherited from System.Collections.ObjectModel.Collection<SortDescriptor>)
	<a href="#">IndexOf</a>	(Inherited from System.Collections.ObjectModel.Collection<SortDescriptor>)
	<a href="#">Insert</a>	(Inherited from

		System.Collections.ObjectModel.Collection<SortDescriptor>)
≡	<a href="#">Move</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<SortDescriptor>)
≡	<a href="#">Remove</a>	(Inherited from System.Collections.ObjectModel.Collection<SortDescriptor>)
≡	<a href="#">RemoveAt</a>	(Inherited from System.Collections.ObjectModel.Collection<SortDescriptor>)

[Top](#)


## Protected Methods

	Name	Description
🔒	<a href="#">BlockReentrancy</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<SortDescriptor>)
🔒	<a href="#">CheckReentrancy</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<SortDescriptor>)
🔒	<a href="#">ClearItems</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<SortDescriptor>)
🔒	<a href="#">InsertItem</a>	Inserts an item into the collection at the specified index. (Inherited from <a href="#">C1.Data.DataSource.DependencyObjectCollection&lt;SortDescriptor&gt;</a> )
🔒	<a href="#">MoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<SortDescriptor>)
🔒	<a href="#">OnCollectionChange</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<SortDescriptor>)

	d	r>)
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<SortDescriptor> r>)
	<a href="#">RemoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<SortDescriptor> r>)
	<a href="#">SetItem</a>	Replaces the element at the specified index. (Inherited from <a href="#">C1.Data.DataSource.DependencyObjectCollection&lt;SortDescriptor&gt;</a> )


[Top](#)

## Public Events

	Name	Description
	<a href="#">CollectionChange</a> d	(Inherited from System.Collections.ObjectModel.ObservableCollection<SortDescriptor> >)

[Top](#)

## Protected Events

	Name	Description
	<a href="#">PropertyChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<SortDescriptor>)

[Top](#)

## See Also

### Reference

[SortDescriptorCollection Class](#)  
[C1.Data.DataSource Namespace](#)

## SortDescriptorCollection Constructor

[C1.Data.DataSource Namespace](#) > [SortDescriptorCollection Class](#) : SortDescriptorCollection Constructor

Initializes a new instance of the [SortDescriptorCollection](#) class.

### Syntax

Visual Basic (Declaration)	
<b>Public Function</b> New()	
C#	
<b>public</b> SortDescriptorCollection()	

### Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

### See Also

#### Reference

[SortDescriptorCollection Class](#)

[SortDescriptorCollection Members](#)

## Enumerations

## FilterDescriptorLogicalOperator

[C1.Data.DataSource Namespace](#) : FilterDescriptorLogicalOperator Enumeration

Enumeration of logical operators for filter collections.

### Syntax

Visual Basic (Declaration)	
<b>Public Enum</b> FilterDescriptorLogicalOperator <b>Inherits</b> System.Enum	
C#	
<b>public enum</b> FilterDescriptorLogicalOperator : System.Enum	

## Members

Member	Description
<b>And</b>	Filters are AND'ed.
<b>Or</b>	Filters are OR'ed.

## Inheritance Hierarchy

System.Object

System.ValueType

System.Enum

**C1.Data.DataSource.FilterDescriptorLogicalOperator**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.Data.DataSource Namespace](#)

## FilterOperator

[C1.Data.DataSource Namespace](#) : FilterOperator Enumeration

Operator used in FilterDescriptor class.

## Syntax

Visual Basic (Declaration)	
<pre>Public Enum FilterOperator     Inherits System.Enum</pre>	
C#	
<pre>public enum FilterOperator : System.Enum</pre>	

## Members



Member	Description
<b>Contains</b>	Left operand must contain the right one.
<b>EndsWith</b>	Left operand must end with the right one.
<b>IsContainedIn</b>	Left operand must be contained in the right one.
<b>IsEqualTo</b>	Left operand must be equal to the right one.
<b>IsGreaterThan</b>	Left operand must be larger than or equal to the right one.
<b>IsGreaterThanOrEqualTo</b>	Left operand must be larger than the right one.
<b>IsLessThan</b>	Left operand must be smaller than the right one.
<b>IsLessThanOrEqualTo</b>	Left operand must be smaller than or equal to the right one.
<b>IsNotEqualTo</b>	Left operand must be different from the right one.
<b>StartsWith</b>	// Left operand must start with the right one.

## Inheritance Hierarchy

System.Object

System.ValueType

System.Enum

**C1.Data.DataSource.FilterOperator**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.Data.DataSource Namespace](#)

# VirtualModeKind

[C1.Data.DataSource Namespace](#) : VirtualModeKind Enumeration

Enumeration of possible virtual modes a [ClientViewSource](#) can be in. Used in the [ClientViewSource.VirtualMode](#) property.

## Syntax

Visual Basic (Declaration)	
<pre>Public Enum VirtualModeKind     Inherits System.Enum</pre>	
C#	
<pre>public enum VirtualModeKind : System.Enum</pre>	

## Members

Member	Description
<b>Managed</b>	Virtual mode is managed by a GUI control bound to the <a href="#">ClientViewSource</a> . That GUI control must have a <a href="#">control handler</a> with the <a href="#">BaseControlHandler.VirtualMode</a> property set to True.
<b>None</b>	Virtual mode is disabled.
<b>Unmanaged</b>	Virtual mode is not managed by a <a href="#">control handler</a> , it is managed by the <a href="#">ClientViewSource</a> itself that is unaware of what controls are bound to it. This option should be used only if you don't have a GUI control that supports virtual mode through a control handler. Although it allows to use virtual mode with any GUI bound control (with or without a control handler), it should be used with caution, only if you can't use the <a href="#">Managed</a> option. See the "Programming Guide" in the Studio for Entity Framework documentation for more details.

## Inheritance Hierarchy

System.Object  
  System.ValueType  
    System.Enum  
      **C1.Data.DataSource.VirtualModeKind**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also





### Reference

[C1.Data.DataSource Namespace](#)

# C1.Data.Entities Namespace

## Overview

## Classes

	Class	Description
	<a href="#">EntityClientCache</a>	Represents a client-side cache specific to Entity Framework.
	<a href="#">EntityClientScope</a>	Defines a scope of data access. Provides facilities to create <a href="#">client views</a> .
	<a href="#">EntityFrameworkExtensions</a>	Provides a set of extensions methods for Entity Framework.
	<a href="#">EntityViewSource</a>	An Entity Framework-specific version of the <a href="#">C1.Data.DataSource.ClientViewSource</a> class.

## See Also

### Reference

[C1.Data.Entity.4 Assembly](#)

## Classes

## EntityClientCache

[C1.Data.Entities Namespace](#) : EntityClientCache Class

Represents a client-side cache specific to Entity Framework.

# Object Model

EntityClientCache

## Syntax

Visual Basic (Declaration)	
<pre>Public Class EntityClientCache     Inherits C1.Data.ClientCacheBase</pre>	
C#	
<pre>public class EntityClientCache : C1.Data.ClientCacheBase</pre>	

## Remarks

Usually, a single instance of this class is created on application startup and exists during the entire application lifetime, while each form, window, or user control works with data using a [EntityClientScope](#) created by calling the [CreateScope](#) method.

## Inheritance Hierarchy

System.Object  
    [C1.Data.ClientCacheBase](#)  
        **C1.Data.Entities.EntityClientCache**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientCache Members](#)  
[C1.Data.Entities Namespace](#)

## Overview

[C1.Data.Entities Namespace](#) : EntityClientCache Class

Represents a client-side cache specific to Entity Framework.

## Object Model

## Syntax

Visual Basic (Declaration)

```
Public Class EntityClientCache
    Inherits C1.Data.ClientCacheBase
```

C#

```
public class EntityClientCache : C1.Data.ClientCacheBase
```

## Remarks

Usually, a single instance of this class is created on application startup and exists during the entire application lifetime, while each form, window, or user control works with data using a [EntityClientScope](#) created by calling the [CreateScope](#) method.

## Inheritance Hierarchy

System.Object

[C1.Data.ClientCacheBase](#)

**C1.Data.Entities.EntityClientCache**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientCache Members](#)

[C1.Data.Entities Namespace](#)


## Members

[Properties](#) [Methods](#)

[C1.Data.Entities Namespace](#) : EntityClientCache Class



The following tables list the members exposed by [EntityClientCache](#).

## Public Constructors

	Name	Description
	<a href="#">EntityClientCache Constructor</a>	Overloaded.







[Top](#)






## Public Properties

	Name	Description
	<a href="#">DbContext</a>	
	<a href="#">ObjectContext</a>	The <a href="#">ObjectContext</a> through which <a href="#">EntityClientCache</a> accesses the data.

[Top](#)

## Public Methods

	Name	Description
	<a href="#">AcceptChanges</a>	Accepts all changes made to entities in the <a href="#">ObjectContext</a> .
	<a href="#">BulkChanges</a>	Used to group massive changes to entities and to allow manual explicit changes of entity states. (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
	<a href="#">CleanupCache</a>	Forces unused memory to be released, unused entities to be detached from the context. It is usually done automatically, so programmers rarely need to call this method in code. (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
	<a href="#">Clear</a>	Clears client-side cache entirely. Call this method if you want to make sure that following queries will fetch fresh data from the server. (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
	<a href="#">CreateScope</a>	Creates an <a href="#">Entity Framework</a> -specific client scope.
	<a href="#">CreateTransaction</a>	Creates a <a href="#">C1.Data.Transactions.ClientTransaction</a> that allows you to easily cancel changes made in transaction scope. (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )

	<a href="#">GetDefault</a>	Returns the default <a href="#">EntityClientCache</a> for a given <a href="#">contextType</a> .
	<a href="#">Refresh</a>	Refreshes data in all C1DataSource controls connected to this ClientCacheBase. (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
	<a href="#">RegisterContext</a>	Overloaded. Registers an <a href="#">ObjectContext</a> as a default for C1DataSource controls for a given <b>context type</b> .
	<a href="#">RejectChanges</a>	Reverts all pending changes for this <a href="#">C1.Data.ClientCacheBase</a> . It is recommended to call this method instead of %System.Data.Objects.ObjectContext.Refresh(System.Data.Objects.RefreshMode, object)%. (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
	<a href="#">SaveChanges</a>	Persists all changes to the server. It is recommended to call this method instead of <b>System.Data.Objects.ObjectContext.SaveChanges()</b> . (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )

[Top](#)

## See Also

### Reference

[EntityClientCache Class](#)

[C1.Data.Entities Namespace](#)

## EntityClientCache Constructor

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) : EntityClientCache Constructor

## Overload List

Overload	Description
<a href="#">EntityClientCache Constructor(DbContext)</a>	Initializes a new instance of the <a href="#">EntityClientCache</a> class.
<a href="#">EntityClientCache Constructor(ObjectContext)</a>	Initializes a new instance of the <a href="#">EntityClientCache</a> class.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientCache Class](#)

[EntityClientCache Members](#)

### EntityClientCache Constructor(DbContext)

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) > [EntityClientCache Constructor](#) : EntityClientCache Constructor(DbContext)

The [object context](#) that is used to access the data.

Initializes a new instance of the [EntityClientCache](#) class.

## Syntax

Visual Basic (Declaration)

```
Public Function New( _  
    ByVal baseContext As System.Data.Entity.DbContext _  
)
```

C#

```
public EntityClientCache(  
    System.Data.Entity.DbContext baseContext  
)
```

### Parameters

*baseContext*

The [object context](#) that is used to access the data.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



[EntityClientCache Class](#)  
[EntityClientCache Members](#)  
[Overload List](#)

## EntityClientCache Constructor(ObjectContext)

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) > [EntityClientCache Constructor](#) : EntityClientCache Constructor(ObjectContext)

The [object context](#) that is used to access the data.

Initializes a new instance of the [EntityClientCache](#) class.

## Syntax

Visual Basic (Declaration)

```
Public Function New( _  
    ByVal baseContext As System.Data.Entity.Core.Objects.ObjectContext _  
)
```

C#

```
public EntityClientCache(  
    System.Data.Entity.Core.Objects.ObjectContext baseContext  
)
```

## Parameters

*baseContext*

The [object context](#) that is used to access the data.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[EntityClientCache Class](#)  
[EntityClientCache Members](#)  
[Overload List](#)


## Methods

[C1.Data.Entities Namespace](#) : EntityClientCache Class

For a list of all members of this type, see [EntityClientCache members](#).

## Public Methods

	Name	Description
≡💎	<a href="#">AcceptChanges</a>	Accepts all changes made to entities in the <a href="#">ObjectContext</a> .
≡💎	<a href="#">BulkChanges</a>	Used to group massive changes to entities and to allow manual explicit changes of entity states. (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
≡💎	<a href="#">CleanupCache</a>	Forces unused memory to be released, unused entities to be detached from the context. It is usually done automatically, so programmers rarely need to call this method in code. (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
≡💎	<a href="#">Clear</a>	Clears client-side cache entirely. Call this method if you want to make sure that following queries will fetch fresh data from the server. (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
≡💎	<a href="#">CreateScope</a>	Creates an <a href="#">Entity Framework-specific client scope</a> .
≡💎	<a href="#">CreateTransaction</a>	Creates a <a href="#">C1.Data.Transactions.ClientTransaction</a> that allows you to easily cancel changes made in transaction scope. (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
≡💎 S	<a href="#">GetDefault</a>	Returns the default <a href="#">EntityClientCache</a> for a given <a href="#">contextType</a> .
≡💎	<a href="#">Refresh</a>	Refreshes data in all <a href="#">C1DataSource</a> controls connected to this <a href="#">ClientCacheBase</a> . (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
≡💎 S	<a href="#">RegisterContext</a>	Overloaded. Registers an <a href="#">ObjectContext</a> as a default for <a href="#">C1DataSource</a> controls for a given <b>context type</b> .
≡💎	<a href="#">RejectChanges</a>	Reverts all pending changes for this <a href="#">C1.Data.ClientCacheBase</a> . It is recommended to call this method instead of

		%System.Data.Objects.ObjectContext.Refresh(System.Data.Objects.Refre shMode, object)%. (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
	<a href="#">SaveChanges</a>	Persists all changes to the server. It is recommended to call this method instead of <b>System.Data.Objects.ObjectContext.SaveChanges()</b> . (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )

[Top](#)

## See Also

### Reference

[EntityClientCache Class](#)

[C1.Data.Entities Namespace](#)

### AcceptChanges Method

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) : AcceptChanges Method

Accepts all changes made to entities in the [ObjectContext](#).

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> AcceptChanges()	
C#	
<b>public void</b> AcceptChanges()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientCache Class](#)

[EntityClientCache Members](#)

## CreateScope Method

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) : CreateScope Method

Creates an [Entity Framework-specific client scope](#).

## Syntax

Visual Basic (Declaration)

```
Public Shadows Function CreateScope() As EntityClientScope
```

C#

```
public new EntityClientScope CreateScope()
```

### Return Value

A new [client scope](#).

## Remarks

Usually, each window, form, or user control creates a [C1.Data.ClientScope](#) and uses it to access entities.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientCache Class](#)

[EntityClientCache Members](#)

## GetDefault Method

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) : GetDefault Method

A subclass of [ObjectContext](#) to get the default [EntityClientCache](#) for.

Returns the default [EntityClientCache](#) for a given [contextType](#).

## Syntax

Visual Basic (Declaration)

```
Public Shared Function GetDefault( _
    ByVal contextType As System.Type _
) As EntityClientCache
```

C#

```
public static EntityClientCache GetDefault(
    System.Type contextType
)
```

## Parameters

*contextType*

A subclass of [ObjectContext](#) to get the default [EntityClientCache](#) for.

## Return Value

The default [EntityClientCache](#) for the given *contextType*.

## Remarks

Creates an [EntityClientCache](#) for the specified *contextType* if it does not already exist; otherwise, returns an existing instance.

It is the same default client cache as used by [C1DataSource](#) with specified [C1DataSource.ObjectContextType](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientCache Class](#)

[EntityClientCache Members](#)

### RegisterContext Method

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) : RegisterContext Method

Registers an [ObjectContext](#) as a default for [C1DataSource](#) controls for a given **context type**.

## Overload List

Overload	Description
<a href="#">RegisterContext(ObjectContext,Type)</a>	Registers an <a href="#">ObjectContext</a> as a default for C1DataSource controls for a given <a href="#">contextType</a> .
<a href="#">RegisterContext(ObjectContext)</a>	Registers an <a href="#">ObjectContext</a> as a default for C1DataSource controls.
<a href="#">RegisterContext(DbContext,Type)</a>	Registers an <a href="#">DbContext</a> as a default for C1DataSource controls for a given <a href="#">contextType</a> .
<a href="#">RegisterContext(DbContext)</a>	Registers an <a href="#">DbContext</a> as a default for C1DataSource controls.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientCache Class](#)

[EntityClientCache Members](#)

[RegisterContext\(ObjectContext,Type\) Method](#)

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) > [RegisterContext Method](#) :

[RegisterContext\(ObjectContext,Type\) Method](#)

An [ObjectContext](#) to set as default.

The type (derived from [ObjectContext](#)) to register the [context](#) for.

Registers an [ObjectContext](#) as a default for C1DataSource controls for a given [contextType](#).

## Syntax

Visual Basic (Declaration)

```
Public Overloads Shared Function RegisterContext( _
    ByVal context As System.Data.Entity.Core.Objects.ObjectContext, _
    ByVal contextType As System.Type _
```

```
) As System.IDisposable
```

```
C#
```

```
public static System.IDisposable RegisterContext(  
    System.Data.Entity.Core.Objects.ObjectContext context,  
    System.Type contextType  
)
```

## Parameters

*context*

An [ObjectContext](#) to set as default.

*contextType*

The type (derived from [ObjectContext](#)) to register the [context](#) for.

## Return Value

An **System.IDisposable** to unregister the [context](#).

## Exceptions

Exception	Description
<b>System.InvalidOperationException</b>	Another context is already registered for the given <a href="#">contextType</a> .

## Remarks

Use this method when you need to customize the default [ObjectContext](#) used in C1DataSource controls. Register a custom [ObjectContext](#) on startup before any C1DataSource instances are created.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientCache Class](#)  
[EntityClientCache Members](#)

[Overload List](#)  
[GetDefault Method](#)

### RegisterContext(ObjectContext) Method

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) > [RegisterContext Method](#) :

RegisterContext(ObjectContext) Method

An [ObjectContext](#) to set as default.

Registers an [ObjectContext](#) as a default for C1DataSource controls.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Shared Function RegisterContext( _  
    ByVal context As System.Data.Entity.Core.Objects.ObjectContext _  
) As System.IDisposable
```

C#

```
public static System.IDisposable RegisterContext(  
    System.Data.Entity.Core.Objects.ObjectContext context  
)
```

### Parameters

*context*

An [ObjectContext](#) to set as default.

### Return Value

An **System.IDisposable** to unregister the [context](#).

## Exceptions

Exception	Description
<b>System.InvalidOperationException</b>	Another context is already registered.

## Remarks

Use this method when you need to customize the default [ObjectContext](#) used in C1DataSource controls. Register a custom [ObjectContext](#) on startup before any C1DataSource instances are created.

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

- [EntityClientCache Class](#)
- [EntityClientCache Members](#)
- [Overload List](#)
- [GetDefault Method](#)

RegisterContext(DbContext,Type) Method

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) > [RegisterContext Method](#) :

RegisterContext(DbContext,Type) Method

An [DbContext](#) to set as default.

The type (derived from [DbContext](#)) to register the [context](#) for.

Registers an [DbContext](#) as a default for C1DataSource controls for a given [contextType](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Shared Function RegisterContext( _     ByVal context As System.Data.Entity.DbContext, _     ByVal contextType As System.Type _ ) As System.IDisposable</pre>	
C#	
<pre>public static System.IDisposable RegisterContext(     System.Data.Entity.DbContext context,     System.Type contextType )</pre>	

### Parameters

*context*

An [DbContext](#) to set as default.

*contextType*

The type (derived from [DbContext](#)) to register the [context](#) for.

## Return Value

An **System.IDisposable** to unregister the [context](#).

## Exceptions

Exception	Description
<b>System.InvalidOperationException</b>	Another context is already registered for the given contextType.

## Remarks

Use this method when you need to customize the default [DbContext](#) used in C1DataSource controls. Register a custom [DbContext](#) on startup before any C1DataSource instances are created.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientCache Class](#)  
[EntityClientCache Members](#)  
[Overload List](#)  
[GetDefault Method](#)

RegisterContext(DbContext) Method

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) > [RegisterContext Method](#) :

RegisterContext(DbContext) Method

An [DbContext](#) to set as default.

Registers an [DbContext](#) as a default for C1DataSource controls.

## Syntax

Visual Basic (Declaration)	
<b>Public Overloads Shared Function</b> RegisterContext( <b>ByVal</b> context <b>As</b> System.Data.Entity.DbContext 	

```
) As System.IDisposable
```

```
C#
```

```
public static System.IDisposable RegisterContext(  
    System.Data.Entity.DbContext context  
)
```

### Parameters

*context*

An [DbContext](#) to set as default.

### Return Value

An **System.IDisposable** to unregister the [context](#).

## Exceptions

Exception	Description
<b>System.InvalidOperationException</b>	Another context is already registered.

## Remarks

Use this method when you need to customize the default [DbContext](#) used in C1DataSource controls. Register a custom [DbContext](#) on startup before any C1DataSource instances are created.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientCache Class](#)

[EntityClientCache Members](#)

[Overload List](#)



[GetDefault Method](#)

## Properties

[C1.Data.Entities Namespace](#) : EntityClientCache Class

For a list of all members of this type, see [EntityClientCache members](#).

## Public Properties

	Name	Description
	<a href="#">DbContext</a>	
	<a href="#">ObjectContext</a>	The <a href="#">ObjectContext</a> through which <a href="#">EntityClientCache</a> accesses the data.

[Top](#)

## See Also

### Reference

[EntityClientCache Class](#)

[C1.Data.Entities Namespace](#)

### DbContext Property

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) : DbContext Property

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property DbContext As System.Data.Entity.DbContext</code>	
C#	
<code>public System.Data.Entity.DbContext DbContext {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientCache Class](#)

[EntityClientCache Members](#)

## ObjectContext Property

[C1.Data.Entities Namespace](#) > [EntityClientCache Class](#) : ObjectContext Property

The [ObjectContext](#) through which [EntityClientCache](#) accesses the data.

## Syntax

Visual Basic (Declaration)	
<pre>Public ReadOnly Property ObjectContext As System.Data.Entity.Core.Objects.ObjectContext</pre>	
C#	
<pre>public System.Data.Entity.Core.Objects.ObjectContext ObjectContext {get;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientCache Class](#)

[EntityClientCache Members](#)

## EntityClientScope

[C1.Data.Entities Namespace](#) : EntityClientScope Class

Defines a scope of data access. Provides facilities to create [client views](#).

## Object Model

EntityClientScope

## Syntax

Visual Basic (Declaration)	
<pre>Public Class EntityClientScope</pre>	

Inherits <a href="#">C1.Data.ClientScope</a>	
C#	
<code>public class EntityClientScope : <a href="#">C1.Data.ClientScope</a></code>	

## Remarks

Usually, one scope is created for each window/user control, and disposed at the end of its lifetime. Entities [pinned to the scope \(marked as needed\)](#) are not evicted from the cache until the scope is [disposed](#) or collected by the GC.

## Inheritance Hierarchy

```

System.Object
  C1.Data.ClientScope
    C1.Data.Entities.EntityClientScope
  
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientScope Members](#)  
[C1.Data.Entities Namespace](#)

## Overview

[C1.Data.Entities Namespace](#) : EntityClientScope Class

Defines a scope of data access. Provides facilities to create [client views](#).

## Object Model

EntityClientScope

## Syntax

Visual Basic (Declaration)	
<pre>Public Class EntityClientScope     Inherits <a href="#">C1.Data.ClientScope</a></pre>	

C#

```
public class EntityClientScope : C1.Data.ClientScope
```

## Remarks

Usually, one scope is created for each window/user control, and disposed at the end of its lifetime. Entities [pinned to the scope \(marked as needed\)](#) are not evicted from the cache until the scope is [disposed](#) or collected by the GC.

## Inheritance Hierarchy

System.Object

[C1.Data.ClientScope](#)

**C1.Data.Entities.EntityClientScope**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientScope Members](#)

[C1.Data.Entities Namespace](#)


## Members

[Properties](#) [Methods](#)

[C1.Data.Entities Namespace](#) : EntityClientScope Class


The following tables list the members exposed by [EntityClientScope](#).

## Public Constructors

	Name	Description
	<a href="#">EntityClientScope Constructor</a>	Initializes a new instance of the <a href="#">EntityClientScope</a> class with the specified <a href="#">EntityClientCache</a> .





[Top](#)

## Public Properties

	Name	Description
	<a href="#">ClientCache</a>	Gets the <a href="#">EntityClientCache</a> to which this <a href="#">client scope</a> is connected.

[Top](#)

## Public Methods

	Name	Description
	<a href="#">AddRef</a>	Overloaded. Marks an entity as needed. Needed entities are not detached/released from the context until the client scope is disposed. (Inherited from <a href="#">C1.Data.ClientScope</a> )
	<a href="#">Dispose</a>	Marks the scope as disposed. Entities that were marked needed by a disposed scope may be disposed of (evicted from the cache, detached from context) unless they are needed by other active scopes. (Inherited from <a href="#">C1.Data.ClientScope</a> )
	<a href="#">GetItems</a>	Overloaded. Gets a <a href="#">client view</a> of entities of a given type.
	<a href="#">Release</a>	Overloaded. Unmark a needed entity. (Inherited from <a href="#">C1.Data.ClientScope</a> )

[Top](#)

## See Also

### Reference

[EntityClientScope Class](#)

[C1.Data.Entities Namespace](#)

## EntityClientScope Constructor

[C1.Data.Entities Namespace](#) > [EntityClientScope Class](#) : EntityClientScope Constructor

An instance of the [EntityClientCache](#) class to which the new [client scope](#) is connected.

Initializes a new instance of the [EntityClientScope](#) class with the specified [EntityClientCache](#).

## Syntax



Visual Basic (Declaration)	
<pre>Public Function New( _     ByVal <i>clientCache</i> As EntityClientCache _ )</pre>	
C#	
<pre>public EntityClientScope(     EntityClientCache <i>clientCache</i> )</pre>	

## Parameters

*clientCache*

An instance of the [EntityClientCache](#) class to which the new [client scope](#) is connected.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[EntityClientScope Class](#)

[EntityClientScope Members](#)

## Methods

[C1.Data.Entities Namespace](#) : EntityClientScope Class

For a list of all members of this type, see [EntityClientScope members](#).

## Public Methods

	Name	Description
⇒💎	<a href="#">AddRef</a>	Overloaded. Marks an entity as needed. Needed entities are not detached/released from the context until the client scope is disposed. (Inherited from <a href="#">C1.Data.ClientScope</a> )
⇒💎	<a href="#">Dispose</a>	Marks the scope as disposed. Entities that were marked needed by a

		disposed scope may be disposed of (evicted from the cache, detached from context) unless they are needed by other active scopes. (Inherited from <a href="#">C1.Data.ClientScope</a> )
≡	<a href="#">GetItems</a>	Overloaded. Gets a <a href="#">client view</a> of entities of a given type.
≡	<a href="#">Release</a>	Overloaded. Unmark a needed entity. (Inherited from <a href="#">C1.Data.ClientScope</a> )

[Top](#)

## See Also

### Reference

[EntityClientScope Class](#)

[C1.Data.Entities Namespace](#)

### GetItems Method

[C1.Data.Entities Namespace](#) > [EntityClientScope Class](#) : [GetItems Method](#)

Gets a [client view](#) of entities of a given type.

## Overload List

Overload	Description
<a href="#">GetItems&lt;T&gt;()</a>	Gets a <a href="#">client view</a> of entities of a given type.
<a href="#">GetItems&lt;T&gt;(String)</a>	Gets a <a href="#">client view</a> of entities from the specified <a href="#">entitySetName</a> .

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientScope Class](#)

[EntityClientScope Members](#)

## GetItems<T>() Method

[C1.Data.Entities Namespace](#) > [EntityClientScope Class](#) > [GetItems Method](#) : GetItems<T>() Method

The type of entities to load.

Gets a [client view](#) of entities of a given type.

## Syntax

Visual Basic (Declaration)	
<b>Public Overloads Function</b> GetItems( <b>Of</b> T)() <b>As</b> ClientView(Of T)	
C#	
<b>public</b> ClientView<T> GetItems<T>()	

### Type Parameters

*T*

The type of entities to load.

### Return Value

A [client view](#) of entities of the specified type.

## Remarks

Entities are loaded using the **entity set** of the matching entity type from the **default entity container**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientScope Class](#)  
[EntityClientScope Members](#)  
[Overload List](#)

## GetItems<T>(String) Method

[C1.Data.Entities Namespace](#) > [EntityClientScope Class](#) > [GetItems Method](#) : GetItems<T>(String) Method

The type of entities in the [entitySetName](#).

The name of the entity set to load entities from.

Gets a [client view](#) of entities from the specified [entitySetName](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function GetItems(Of T)( _     ByVal entitySetName As System.String _ ) As ClientView(Of T)</pre>	
C#	
<pre>public ClientView&lt;T&gt; GetItems&lt;T&gt;(     System.string entitySetName )</pre>	

### Parameters

*entitySetName*

The name of the entity set to load entities from.

### Type Parameters

*T*

The type of entities in the [entitySetName](#).

### Return Value

A [client view](#) of entities from the specified [entitySetName](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference


[EntityClientScope Class](#)  
[EntityClientScope Members](#)  
[Overload List](#)

## Properties

[C1.Data.Entities Namespace](#) : EntityClientScope Class

For a list of all members of this type, see [EntityClientScope members](#).

## Public Properties

	Name	Description
	<a href="#">ClientCache</a>	Gets the <a href="#">EntityClientCache</a> to which this <a href="#">client scope</a> is connected.

[Top](#)

## See Also

### Reference

[EntityClientScope Class](#)

[C1.Data.Entities Namespace](#)

### ClientCache Property

[C1.Data.Entities Namespace](#) > [EntityClientScope Class](#) : ClientCache Property

Gets the [EntityClientCache](#) to which this [client scope](#) is connected.

## Syntax

Visual Basic (Declaration)	
<b>Public Shadows ReadOnly Property</b> ClientCache <b>As</b> <a href="#">EntityClientCache</a>	
C#	
<b>public new</b> <a href="#">EntityClientCache</a> ClientCache { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityClientScope Class](#)

[EntityClientScope Members](#)

# EntityFrameworkExtensions

[C1.Data.Entities Namespace](#) : EntityFrameworkExtensions Class

Provides a set of extensions methods for Entity Framework.

## Object Model

EntityFrameworkExtensions

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public MustInherit NotInheritable Class EntityFrameworkExtensions
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static class EntityFrameworkExtensions
```

## Inheritance Hierarchy

System.Object

**C1.Data.Entities.EntityFrameworkExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityFrameworkExtensions Members](#)

[C1.Data.Entities Namespace](#)

## Overview

[C1.Data.Entities Namespace](#) : EntityFrameworkExtensions Class

Provides a set of extensions methods for Entity Framework.

## Object Model

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public MustInherit NotInheritable Class EntityFrameworkExtensions</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static class EntityFrameworkExtensions</pre>	

## Inheritance Hierarchy

System.Object

**C1.Data.Entities.EntityFrameworkExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityFrameworkExtensions Members](#)

[C1.Data.Entities Namespace](#)


## Members


[Methods](#)

[C1.Data.Entities Namespace](#) : EntityFrameworkExtensions Class

The following tables list the members exposed by [EntityFrameworkExtensions](#).

## Public Methods

	Name	Description
	<a href="#">AsCollectionView&lt;T&gt;</a> >	Converts an <b>System.Data.Entity.Core.Objects.DataClasses.EntityCollection`</b>

		<b>1</b> to an editable <b>System.ComponentModel.ICollectionView</b> .
	<a href="#">AsLive</a>	Overloaded. Converts an <b>System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1</b> to an editable <a href="#">live view</a> .

[Top](#)

## See Also



### Reference

[EntityFrameworkExtensions Class](#)

[C1.Data.Entities Namespace](#)

### Methods

>

Name	Description
 <a href="#">AsCollectionView&lt;T&gt;</a>	Converts an <b>System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1</b> to an editable <b>System.ComponentModel.ICollectionView</b> .
 <a href="#">AsLive</a>	Overloaded. Converts an <b>System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1</b> to an editable <a href="#">live view</a> .

[Top](#)

## See Also

### Reference

[EntityFrameworkExtensions Class](#)

[C1.Data.Entities Namespace](#)

### AsCollectionView<T> Method

[C1.Data.Entities Namespace](#) > [EntityFrameworkExtensions Class](#) : AsCollectionView<T> Method

The type of the entities in the [entities](#).

The **System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1** to convert.

Converts an **System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1** to an editable **System.ComponentModel.ICollectionView**.



## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Shared Function AsCollectionView(Of T As {Class,
System.Data.Entity.Core.Objects.DataClasses.IEntityWithRelationships}))( _
    ByVal entities As
System.Data.Entity.Core.Objects.DataClasses.EntityCollection(Of T) _
) As System.ComponentModel.ICollectionView
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static System.ComponentModel.ICollectionView AsCollectionView<T>(
    System.Data.Entity.Core.Objects.DataClasses.EntityCollection<T> entities
)
where T: class,
System.Data.Entity.Core.Objects.DataClasses.IEntityWithRelationships
```

### Parameters

*entities*

The **System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1** to convert.

### Type Parameters

*T*

The type of the entities in the [entities](#).

### Return Value

The resulting **System.ComponentModel.ICollectionView**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityFrameworkExtensions Class](#)

[EntityFrameworkExtensions Members](#)

## AsLive Method

[C1.Data.Entities Namespace](#) > [EntityFrameworkExtensions Class](#) : AsLive Method

Converts an **System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1** to an editable [live view](#).

## Overload List

Overload	Description
<a href="#">AsLive&lt;T&gt;(EntityCollection&lt;T&gt;)</a>	Converts an <b>System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1</b> to an editable <a href="#">live view</a> .
<a href="#">AsLive&lt;T&gt;(ICollection&lt;T&gt;,EntityClientScope)</a>	Converts a POCO <b>System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1</b> to an editable <a href="#">live view</a> .

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityFrameworkExtensions Class](#)

[EntityFrameworkExtensions Members](#)

[AsLive<T>\(EntityCollection<T>\) Method](#)

[C1.Data.Entities Namespace](#) > [EntityFrameworkExtensions Class](#) > [AsLive Method](#) :

[AsLive<T>\(EntityCollection<T>\) Method](#)

The type of the entities in the [entities](#).

The entity collection to convert.

Converts an **System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1** to an editable [live view](#).

## Syntax

Visual Basic (Declaration)	
<pre> &lt;System.Runtime.CompilerServices.ExtensionAttribute()&gt; Public Overloads Shared Function AsLive(Of T As {Class, System.Data.Entity.Core.Objects.DataClasses.IEntityWithRelationships}))( _     ByVal entities As System.Data.Entity.Core.Objects.DataClasses.EntityCollection(Of T) _ ) As View(Of T) </pre>	
C#	
<pre> [System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsLive&lt;T&gt;(     System.Data.Entity.Core.Objects.DataClasses.EntityCollection&lt;T&gt; entities ) where T: class, System.Data.Entity.Core.Objects.DataClasses.IEntityWithRelationships </pre>	

### Parameters

*entities*

The entity collection to convert.

### Type Parameters

*T*

The type of the entities in the [entities](#).

### Return Value

The resulting [live view](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityFrameworkExtensions Class](#)  
[EntityFrameworkExtensions Members](#)  
[Overload List](#)

AsLive<T>(ICollection<T>,EntityClientScope) Method

[C1.Data.Entities Namespace](#) > [EntityFrameworkExtensions Class](#) > [AsLive Method](#) :

AsLive<T>(ICollection<T>,EntityClientScope) Method

The type of the entities in the [pocoCollection](#).

The entity collection to convert. It must be of type

**System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1**.

The [EntityClientScope](#) to which the entity owning this collection belongs (in which it was fetched by a query or created).

Converts a POCO **System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1** to an editable [live view](#).

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function AsLive(Of T As Class)( _
    ByVal pocoCollection As System.Collections.Generic.ICollection(Of T), _
    ByVal scope As EntityClientScope _
) As View(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<T> AsLive<T>(
    System.Collections.Generic.ICollection<T> pocoCollection,
    EntityClientScope scope
)
where T: class
```

### Parameters

*pocoCollection*

The entity collection to convert. It must be of type

**System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1**.

*scope*

The [EntityClientScope](#) to which the entity owning this collection belongs (in which it was fetched by a query or created).

### Type Parameters

*T*

The type of the entities in the [pocoCollection](#).

## Return Value

The resulting [live view](#).

## Exceptions

Exception	Description
<b>System.ArgumentException</b>	The pocoCollection is not of type <b>System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1</b> .

## Remarks

When POCO objects are used (with proxies), navigation collection properties are typed as **System.Collections.Generic.ICollection`1**, not **System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1** (although they are **System.Data.Entity.Core.Objects.DataClasses.EntityCollection`1** at run time). That is why a special AsLive extension method must be used.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityFrameworkExtensions Class](#)  
[EntityFrameworkExtensions Members](#)  
[Overload List](#)

## EntityViewSource

[C1.Data.Entities Namespace](#) : EntityViewSource Class

An Entity Framework-specific version of the [C1.Data.DataSource.ClientViewSource](#) class.

## Object Model

EntityViewSource

## Syntax

Visual Basic (Declaration)	
<pre>Public Class EntityViewSource     Inherits C1.Data.DataSource.ClientViewSource</pre>	
C#	
<pre>public class EntityViewSource : C1.Data.DataSource.ClientViewSource</pre>	

## Remarks

To load data, specify the [name](#) of an **entity set** to load the data from.

## Inheritance Hierarchy

```
System.Object
  System.Windows.Threading.DispatcherObject
    System.Windows.DependencyObject
      C1.Data.DataSource.ClientViewSource
        C1.Data.Entities.EntityViewSource
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityViewSource Members](#)  
[C1.Data.Entities Namespace](#)

### Overview

[C1.Data.Entities Namespace](#) : EntityViewSource Class

An Entity Framework-specific version of the [C1.Data.DataSource.ClientViewSource](#) class.

## Object Model

EntityViewSource

## Syntax

Visual Basic (Declaration)	
<pre>Public Class EntityViewSource     Inherits C1.Data.DataSource.ClientViewSource</pre>	
C#	
<pre>public class EntityViewSource : C1.Data.DataSource.ClientViewSource</pre>	

## Remarks

To load data, specify the [name](#) of an **entity set** to load the data from.

## Inheritance Hierarchy

```
System.Object
  System.Windows.Threading.DispatcherObject
    System.Windows.DependencyObject
      C1.Data.DataSource.ClientViewSource
        C1.Data.Entities.EntityViewSource
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityViewSource Members](#)  
[C1.Data.Entities Namespace](#)


## Members

[Properties](#) [Methods](#) [Events](#)

[C1.Data.Entities Namespace](#) : EntityViewSource Class










The following tables list the members exposed by [EntityViewSource](#).

## Public Constructors










	Name	Description
	<a href="#">EntityViewSource Constructor</a>	






[Top](#)

## Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	Gets or sets a value indicating whether <a href="#">Load</a> is automatically invoked on startup and when a change occurs that impacts the query composed by the <a href="#">C1.Data.DataSource.ClientViewSource</a> . The default is True. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">BaseView</a>	Gets or sets an instance of <a href="#">C1.Data.ClientView&lt;T&gt;</a> that the <a href="#">C1.Data.DataSource.ClientViewSource</a> uses as the base for composing queries. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">CacheTimeout</a>	Gets or sets the period of time entities loaded in virtual mode are kept in the cache without checking whether they are needed or not. If an entity was neither used nor considered needed for a period of time longer than <a href="#">CacheTimeout</a> , <a href="#">C1.Data.DataSource.ClientViewSource</a> may evict it from the cache. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">CurrentClientView</a>	Gets the current <a href="#">client view</a> used to load entities, or null in <a href="#">virtual mode</a> . (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">DataView</a>	Gets the current view of entities resulting from the last load operation. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)
	<a href="#">EntitySetName</a>	Gets or sets the name of the <b>entity set</b> to load entities from.
	<a href="#">FilterDescriptors</a>	Gets the collection of <a href="#">C1.Data.DataSource.FilterDescriptor</a> objects used when performing loads. (Inherited from

















		<a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">FilterOperator</a>	Gets or sets the logical operator used for combining <a href="#">FilterDescriptors</a> in the filter collection. The default value is <a href="#">C1.Data.DataSource.FilterDescriptorLogicalOperator.And</a> . (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">GroupDescriptors</a>	Gets the collection of <a href="#">C1.Data.DataSource.GroupDescriptor</a> objects used to organize the loaded entities into groups. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">Include</a>	Gets or sets a comma-separated list of property paths that specify related objects to include during the <a href="#">Load</a> operation. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">IsLoadingData</a>	Gets a value indicating whether the <a href="#">C1.Data.DataSource.ClientViewSource</a> is currently loading data. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">LoadCommand</a>	Gets an <b>System.Windows.Input.ICommand</b> that invokes <a href="#">Load</a> on this <a href="#">C1.Data.DataSource.ClientViewSource</a> . (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">LoadDelay</a>	Gets or sets the delay before an automatic data loading operation is started. It is the delay from the time a change prompting automatic load occurs until the time the resulting <a href="#">Load</a> is started. The default delay is 25 milliseconds. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">LoadSize</a>	Gets or sets the maximum number of items to load each time a <a href="#">Load</a> is executed. When equal to 0, all requested entities will be loaded. The default is 0. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">MoveToFirstOnLoad</a>	Gets or sets a value indicating that the first item must be made current after <a href="#">Load</a> operation is completed if current item was not

		set by other means. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">Name</a>	Overridden. Gets a name of this <a href="#">EntityViewSource</a> to reference it in a <a href="#">C1DataSource.ViewSources</a> collection. It is determined by the <a href="#">EntitySetName</a> but can be overridden by the <a href="#">NameOverride</a> .
	<a href="#">NameOverride</a>	Gets or sets a value that overrides the value of the <a href="#">Name</a> property. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">PageSize</a>	Gets or sets the number of items displayed on each page of the <a href="#">DataView</a> , or the number of items to fetch in each query in <a href="#">virtual mode</a> , or 0 to disable paging. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">SortDescriptors</a>	Gets the collection of <a href="#">C1.Data.DataSource.SortDescriptor</a> objects used to sort the data. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">VirtualMode</a>	Gets or sets a value indicating whether the <a href="#">C1.Data.DataSource.ClientViewSource</a> is in virtual mode. Virtual mode is an innovative technology allowing to bind GUI controls directly to very large data sets without delays and performance degradation and without inconvenience of paging. By default, virtual mode is disabled (the default value is <a href="#">C1.Data.DataSource.VirtualModeKind.None</a> ). (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )

[Top](#)

## Public Methods


	Name	Description
	<a href="#">ClearValue</a>	Overloaded. (Inherited from <a href="#">System.Windows.DependencyObject</a> )
	<a href="#">CoerceValue</a>	(Inherited from <a href="#">System.Windows.DependencyObject</a> )

 <a href="#">DeferLoad</a>	Used to group changes to multiple load-affecting properties together, deferring the resulting load operations so a single load operation is performed in the end, that is, when the object returned from this method is disposed. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <a href="#">Equals</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">GetHashCode</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">GetLocalValueEnumerator</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">InvalidateProperty</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">Load</a>	Starts a load operation. Any pending load will be implicitly canceled. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <a href="#">LoadRange</a>	If in <a href="#">virtual mode</a> , loads a specific range of entities. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">Refresh</a>	Starts a load operation ignoring the client-side cache. Any pending load will be implicitly canceled. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <a href="#">SetCurrentValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">SetValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)

[Top](#)



## Protected Methods

Name	Description
------	-------------

	<a href="#">OnPropertyChanged</a>	(Inherited from <a href="#">System.Windows.DependencyObject</a> )
---	-----------------------------------	---

[Top](#)

## Public Events

	Name	Description
	<a href="#">LoadedData</a>	Occurs when a load operation is completed, or when an exception was thrown during the load operation. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">PropertyChanged</a>	Occurs when a property value changes. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )

[Top](#)

## See Also

### Reference

[EntityViewSource Class](#)

[C1.Data.Entities Namespace](#)

## EntityViewSource Constructor

[C1.Data.Entities Namespace](#) > [EntityViewSource Class](#) : EntityViewSource Constructor

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public EntityViewSource()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[EntityViewSource Class](#)







[EntityViewSource Members](#)











## Properties








[C1.Data.Entities Namespace](#) : EntityViewSource Class

For a list of all members of this type, see [EntityViewSource members](#).

## Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	Gets or sets a value indicating whether <a href="#">Load</a> is automatically invoked on startup and when a change occurs that impacts the query composed by the <a href="#">C1.Data.DataSource.ClientViewSource</a> . The default is True. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">BaseView</a>	Gets or sets an instance of <a href="#">C1.Data.ClientView&lt;T&gt;</a> that the <a href="#">C1.Data.DataSource.ClientViewSource</a> uses as the base for composing queries. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">CacheTimeout</a>	Gets or sets the period of time entities loaded in virtual mode are kept in the cache without checking whether they are needed or not. If an entity was neither used nor considered needed for a period of time longer than <a href="#">CacheTimeout</a> , <a href="#">C1.Data.DataSource.ClientViewSource</a> may evict it from the cache. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">CurrentClientView</a>	Gets the current <a href="#">client view</a> used to load entities, or null in <a href="#">virtual mode</a> . (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">DataView</a>	Gets the current view of entities resulting from the last load operation. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)

	<a href="#">Dispatcher</a>	(Inherited from <code>System.Windows.Threading.DispatcherObject</code> )
	<a href="#">EntitySetName</a>	Gets or sets the name of the <b>entity set</b> to load entities from.
	<a href="#">FilterDescriptors</a>	Gets the collection of <a href="#">C1.Data.DataSource.FilterDescriptor</a> objects used when performing loads. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">FilterOperator</a>	Gets or sets the logical operator used for combining <a href="#">FilterDescriptors</a> in the filter collection. The default value is <a href="#">C1.Data.DataSource.FilterDescriptorLogicalOperator.And</a> . (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">GroupDescriptors</a>	Gets the collection of <a href="#">C1.Data.DataSource.GroupDescriptor</a> objects used to organize the loaded entities into groups. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">Include</a>	Gets or sets a comma-separated list of property paths that specify related objects to include during the <a href="#">Load</a> operation. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">IsLoadingData</a>	Gets a value indicating whether the <a href="#">C1.Data.DataSource.ClientViewSource</a> is currently loading data. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">IsSealed</a>	(Inherited from <code>System.Windows.DependencyObject</code> )
	<a href="#">LoadCommand</a>	Gets an <b>System.Windows.Input.ICommand</b> that invokes <a href="#">Load</a> on this <a href="#">C1.Data.DataSource.ClientViewSource</a> . (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">LoadDelay</a>	Gets or sets the delay before an automatic data loading operation is started. It is the delay from the time a change prompting automatic load occurs until the time the resulting <a href="#">Load</a> is started. The default delay is 25 milliseconds. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )

	<a href="#">LoadSize</a>	Gets or sets the maximum number of items to load each time a <a href="#">Load</a> is executed. When equal to 0, all requested entities will be loaded. The default is 0. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">MoveToFirstOnLoad</a>	Gets or sets a value indicating that the first item must be made current after <a href="#">Load</a> operation is completed if current item was not set by other means. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">Name</a>	Overridden. Gets a name of this <a href="#">EntityViewSource</a> to reference it in a <a href="#">C1DataSource.ViewSources</a> collection. It is determined by the <a href="#">EntitySetName</a> but can be overridden by the <a href="#">NameOverride</a> .
	<a href="#">NameOverride</a>	Gets or sets a value that overrides the value of the <a href="#">Name</a> property. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">PageSize</a>	Gets or sets the number of items displayed on each page of the <a href="#">DataView</a> , or the number of items to fetch in each query in <a href="#">virtual mode</a> , or 0 to disable paging. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">SortDescriptors</a>	Gets the collection of <a href="#">C1.Data.DataSource.SortDescriptor</a> objects used to sort the data. (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<a href="#">VirtualMode</a>	Gets or sets a value indicating whether the <a href="#">C1.Data.DataSource.ClientViewSource</a> is in virtual mode. Virtual mode is an innovative technology allowing to bind GUI controls directly to very large data sets without delays and performance degradation and without inconvenience of paging. By default, virtual mode is disabled (the default value is <a href="#">C1.Data.DataSource.VirtualModeKind.None</a> ). (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )

[Top](#)

## See Also

## Reference

[EntityViewSource Class](#)

[C1.Data.Entities Namespace](#)

## EntitySetName Property

[C1.Data.Entities Namespace](#) > [EntityViewSource Class](#) : EntitySetName Property

Gets or sets the name of the **entity set** to load entities from.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.CategoryAttribute("Common")&gt; &lt;System.ComponentModel.DefaultValueAttribute()&gt; Public Property EntitySetName As System.String</pre>	
C#	
<pre>[System.ComponentModel.Category("Common")] [System.ComponentModel.DefaultValue()] public System.string EntitySetName {get; set;}</pre>	

## Remarks

Changing the value of this property causes the [EntityViewSource](#) to reload data if [AutoLoad](#) is set to true.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityViewSource Class](#)

[EntityViewSource Members](#)

## Name Property

[C1.Data.Entities Namespace](#) > [EntityViewSource Class](#) : Name Property

Gets a name of this [EntityViewSource](#) to reference it in a [C1DataSource.ViewSources](#) collection. It is determined by the [EntitySetName](#) but can be overridden by the [NameOverride](#).



## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.CategoryAttribute("Common")&gt; Public Overrides ReadOnly Property Name As System.String</pre>	
C#	
<pre>[System.ComponentModel.Category("Common")] public override System.string Name {get;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference


[EntityViewSource Class](#)

[EntityViewSource Members](#)

# C1.Data.Transactions Namespace

## Overview

## Classes

	Class	Description
	<a href="#">ClientTransaction</a>	Represents a transaction that tracks client-side changes and can roll them back.

## See Also

### Reference

[C1.Data.Entity.4 Assembly](#)

# Classes

## ClientTransaction

[C1.Data.Transactions Namespace](#) : ClientTransaction Class

Represents a transaction that tracks client-side changes and can roll them back.

## Object Model

ClientTransaction

## Syntax

Visual Basic (Declaration)

```
Public Class ClientTransaction
    Implements C1.LiveLinq.ITransaction
```

C#

```
public class ClientTransaction : C1.LiveLinq.ITransaction
```

## Remarks

To create a new independent transaction, use the [C1.Data.ClientCacheBase.CreateTransaction](#) method. To create a child transaction, use the [constructor](#).

## Inheritance Hierarchy

System.Object

**C1.Data.Transactions.ClientTransaction**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Members](#)

[C1.Data.Transactions Namespace](#)

## Overview

[C1.Data.Transactions Namespace](#) : ClientTransaction Class

Represents a transaction that tracks client-side changes and can roll them back.

## Object Model

ClientTransaction

## Syntax

Visual Basic (Declaration)

```
Public Class ClientTransaction
    Implements C1.LiveLinq.ITransaction
```

C#

```
public class ClientTransaction : C1.LiveLinq.ITransaction
```

## Remarks

To create a new independent transaction, use the [C1.Data.ClientCacheBase.CreateTransaction](#) method. To create a child transaction, use the [constructor](#).

## Inheritance Hierarchy

System.Object

**C1.Data.Transactions.ClientTransaction**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Members](#)

[C1.Data.Transactions Namespace](#)


## Members

[Properties](#) [Methods](#) [Events](#)

[C1.Data.Transactions Namespace](#) : ClientTransaction Class



The following tables list the members exposed by [ClientTransaction](#).

## Public Constructors

	Name	Description
	<a href="#">ClientTransaction Constructor</a>	Initializes a child (nested) transaction, a new instance of the <a href="#">ClientTransaction</a> class with a specified parent transaction.






[Top](#)

## Public Properties

	Name	Description
	<a href="#">HasChanges</a>	Gets a value indicating whether any changes were made in the <a href="#">scope</a> of this <a href="#">transaction</a> .
	<a href="#">State</a>	Gets the <a href="#">state</a> the <a href="#">transaction</a> is in.


[Top](#)

## Public Methods

	Name	Description
	<a href="#">Commit</a>	Commits the transaction if it was not committed before. Commits changes that were made while this transaction's scope was open.
	<a href="#">Dispose</a>	Disposes of the <a href="#">ClientTransaction</a> .
	<a href="#">Rollback</a>	Rolls back the transaction.
	<a href="#">Scope</a>	Opens the transaction scope.
	<a href="#">ScopeDataContext</a>	Wraps an object so the transaction <a href="#">scope</a> is automatically opened when a value is being assigned to a property of the wrapped object.

[Top](#)

## Public Events

	Name	Description
	<a href="#">PropertyChanged</a>	Occurs when a property value changes, after it has been changed.

[Top](#)

## See Also

### Reference

[ClientTransaction Class](#)

[C1.Data.Transactions Namespace](#)

## ClientTransaction Constructor

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : ClientTransaction Constructor

The parent transaction. Cannot be null.

Initializes a child (nested) transaction, a new instance of the [ClientTransaction](#) class with a specified parent transaction.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function New( _     ByVal parent As ClientTransaction _ )</pre>	
C#	
<pre>public ClientTransaction(     ClientTransaction parent )</pre>	

### Parameters

*parent*

The parent transaction. Cannot be null.

## Exceptions

Exception	Description
-----------	-------------

<b>System.ArgumentNullException</b>	The parent is null.
-------------------------------------	---------------------

## Remarks

A child transaction is automatically committed/rolled back if its parent transaction is committed/rolled back.

Create a child transaction in cases where you need to open a new window for editing a portion of data that is being editing in an already open transaction.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Class](#)

[ClientTransaction Members](#)

## Methods

[C1.Data.Transactions Namespace](#) : ClientTransaction Class

For a list of all members of this type, see [ClientTransaction members](#).

## Public Methods

	Name	Description
⇒	<a href="#">Commit</a>	Commits the transaction if it was not committed before. Commits changes that were made while this transaction's scope was open.
⇒	<a href="#">Dispose</a>	Disposes of the <a href="#">ClientTransaction</a> .
⇒	<a href="#">Rollback</a>	Rolls back the transaction.
⇒	<a href="#">Scope</a>	Opens the transaction scope.
⇒	<a href="#">ScopeDataContext</a>	Wraps an object so the transaction <a href="#">scope</a> is automatically opened when a value is being assigned to a property of the wrapped object.

[Top](#)

## See Also

### Reference

[ClientTransaction Class](#)

[C1.Data.Transactions Namespace](#)

### Commit Method

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : Commit Method

Commits the transaction if it was not committed before. Commits changes that were made while this transaction's scope was open.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Commit()	
C#	
<b>public void</b> Commit()	

## Exceptions

Exception	Description
<b>System.InvalidOperationException</b>	The <a href="#">State</a> is not <a href="#">C1.LiveLinq.TransactionState.Open</a> .

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Class](#)

[ClientTransaction Members](#)

### Dispose Method

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : Dispose Method

Disposes of the [ClientTransaction](#).

## Syntax

Visual Basic (Declaration)	
<code>Public Sub Dispose()</code>	
C#	
<code>public void Dispose()</code>	

## Remarks

If the [State](#) is [C1.LiveLinq.TransactionState.Open](#), the [ClientTransaction](#) is automatically [rolled back](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Class](#)  
[ClientTransaction Members](#)

### Rollback Method

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : Rollback Method

Rolls back the transaction.

## Syntax

Visual Basic (Declaration)	
<code>Public Sub Rollback()</code>	
C#	
<code>public void Rollback()</code>	

## Exceptions



Exception	Description
<b>System.InvalidOperationException</b>	The <a href="#">State</a> is <a href="#">C1.LiveLinq.TransactionState.Committed</a> or <a href="#">C1.LiveLinq.TransactionState.Committing</a> .

## Remarks

Calling this method cancels the changes that were made in the [scope](#) of this [transaction](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Class](#)

[ClientTransaction Members](#)

### Scope Method

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : Scope Method

Opens the transaction scope.

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> Scope() <b>As</b> System.IDisposable	
C#	
<b>public</b> System.IDisposable Scope()	

### Return Value

An instance of an **System.IDisposable** that will close the scope when its **System.IDisposable.Dispose** method is called.

## Exceptions

Exception	Description
-----------	-------------

<b>System.InvalidOperationException</b>	The <b>ClientTransaction.State</b> is not <a href="#">C1.LiveLinq.TransactionState.Open</a> .
---	---

## Remarks

The transaction tracks changes only when they are made inside an open scope.

Calling **System.IDisposable.Dispose** on the return value closes the scope.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Class](#)

[ClientTransaction Members](#)

### ScopeDataContext Method

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : ScopeDataContext Method

The object to wrap.

Wraps an object so the transaction [scope](#) is automatically opened when a value is being assigned to a property of the wrapped object.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function ScopeDataContext( _     ByVal entity As System.Object _ ) As System.Object</pre>	
C#	
<pre>public System.object ScopeDataContext(     System.object entity )</pre>	

### Parameters

*entity*

The object to wrap.

## Return Value

The wrapped object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientTransaction Class](#)



[ClientTransaction Members](#)

## Properties

[C1.Data.Transactions Namespace](#) : [ClientTransaction Class](#)

For a list of all members of this type, see [ClientTransaction members](#).

## Public Properties

	Name	Description
	<a href="#">HasChanges</a>	Gets a value indicating whether any changes were made in the <a href="#">scope</a> of this <a href="#">transaction</a> .
	<a href="#">State</a>	Gets the <a href="#">state</a> the <a href="#">transaction</a> is in.

[Top](#)

## See Also

## Reference

[ClientTransaction Class](#)

[C1.Data.Transactions Namespace](#)

## HasChanges Property

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : [HasChanges Property](#)

Gets a value indicating whether any changes were made in the [scope](#) of this [transaction](#).

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property HasChanges As System.Boolean</code>	
C#	
<code>public System.bool HasChanges {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Class](#)

[ClientTransaction Members](#)

### State Property

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : State Property

Gets the [state](#) the [transaction](#) is in.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property State As TransactionState</code>	
C#	
<code>public TransactionState State {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference


[ClientTransaction Class](#)  
[ClientTransaction Members](#)  
[TransactionState Enumeration](#)

## Events

[C1.Data.Transactions Namespace](#) : [ClientTransaction Class](#)

For a list of all members of this type, see [ClientTransaction members](#).

## Public Events

	Name	Description
	<a href="#">PropertyChanged</a>	Occurs when a property value changes, after it has been changed.

[Top](#)

## See Also

### Reference

[ClientTransaction Class](#)  
[C1.Data.Transactions Namespace](#)

### PropertyChanged Event

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : [PropertyChanged Event](#)

Occurs when a property value changes, after it has been changed.

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> <a href="#">PropertyChanged</a> <b>As</b> System.ComponentModel.PropertyChangedEventHandler	
C#	
<b>public event</b> System.ComponentModel.PropertyChangedEventHandler <a href="#">PropertyChanged</a>	

## Event Data

The event handler receives an argument of type `System.ComponentModel.PropertyChangedEventArgs` containing data related to this event. The following **PropertyChangedEventArgs** properties provide information specific to this event.

Property	Description
<b>PropertyName</b>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Class](#)

[ClientTransaction Members](#)

# C1.Data.Util Namespace

## Overview

[Inheritance Hierarchy](#)

## Classes

	Class	Description
	<a href="#">DesignTime</a>	

## See Also

### Reference

[C1.Data.Entity.4 Assembly](#)

## Classes

## DesignTime

[C1.Data.Util Namespace](#) : DesignTime Class

## Object Model

DesignTime

# Syntax

Visual Basic (Declaration)	
<code>Public Class DesignTime</code>	
C#	
<code>public class DesignTime</code>	

# Inheritance Hierarchy

System.Object  
    **C1.Data.Util.DesignTime**

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[DesignTime Members](#)  
[C1.Data.Util Namespace](#)

## Overview

[C1.Data.Util Namespace](#) : DesignTime Class

# Object Model

DesignTime

# Syntax

Visual Basic (Declaration)	
<code>Public Class DesignTime</code>	
C#	
<code>public class DesignTime</code>	

# Inheritance Hierarchy

System.Object  
**C1.Data.Util.DesignTime**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DesignTime Members](#)  
[C1.Data.Util Namespace](#)

## Members

[Methods](#)

[C1.Data.Util Namespace](#) : DesignTime Class


The following tables list the members exposed by [DesignTime](#).

## Public Constructors

	Name	Description
	<a href="#">DesignTime Constructor</a>	

[Top](#)

## Public Methods

	Name	Description
 	<a href="#">AboutBox</a>	

[Top](#)

## See Also

### Reference

[DesignTime Class](#)  
[C1.Data.Util Namespace](#)



# DesignTime Constructor

[C1.Data.Util Namespace](#) > [DesignTime Class](#) : DesignTime Constructor

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public DesignTime()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DesignTime Class](#)  
[DesignTime Members](#)

## Methods

>

Name	Description
 <a href="#">AboutBox</a>	

[Top](#)

## See Also

### Reference

[DesignTime Class](#)  
[C1.Data.Util Namespace](#)

### AboutBox Method

[C1.Data.Util Namespace](#) > [DesignTime Class](#) : AboutBox Method

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared Sub AboutBox( _     ByVal type As System.Type _ )</pre>	
C#	
<pre>public static void AboutBox(     System.Type type )</pre>	

## Parameters

*type*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[DesignTime Class](#)

[DesignTime Members](#)


# C1.LiveLinq.4 Assembly








## Namespaces

## C1.LiveLinq Namespace



## Overview

## Classes

	Class	Description
	<a href="#">CompiledQuery</a>	Provides for compilation and cache of queries for reuse.





	<a href="#">DeletedStateIsAvailableAttribute</a>	Indicates that the properties of an object of a class still return correct property values after the object has been deleted from the collection it belonged to.
	<a href="#">Hints</a>	Provides a static (extension) method used as a hint that can be applied to an expression (usually, a property) in a query.
	<a href="#">IndexedQueryExtensions</a>	Provides a set of static (extension) methods for querying objects that implement <a href="#">C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</a> .
	<a href="#">LiveViewExtensions</a>	Provides a set of static (extension) methods used in queries to define live views.
	<a href="#">Ordering&lt;T&gt;</a>	Represents a sorted <a href="#">C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</a> .
	<a href="#">QueryOptimizationException</a>	Represents an exception that is thrown when <a href="#">Hints</a> in a query require using a certain mandatory optimization which is impossible in the current query context.
	<a href="#">SourceChangeEventArgs&lt;T&gt;</a>	Provides data for the <a href="#">IObservableSource&lt;T&gt;.Changed</a> event.

## Interfaces

	Interface	Description
	<a href="#">IObservableSource&lt;T&gt;</a>	Provides methods and events that are required for LiveLinq functionality, indexing and live views.
	<a href="#">ITransaction</a>	Represents a transaction with an explicit scope.

## Enumerations

	Enumeration	Description
--	-------------	-------------

 <a href="#">IndexingHintAction</a>	Specifies the actions taken by LiveLinq query optimizer when it encounters an <b>Indexed()</b> hint applied to an expression (usually, a property) in a query.
 <a href="#">Order</a>	Indicates if a certain order is required in the result collection of an operation.
 <a href="#">SourceChangeType</a>	Describes a change occurring in a collection.
 <a href="#">TransactionState</a>	Enumeration of the possible states an <a href="#">ITransaction</a> can be in.

## See Also

### Reference

[C1.LiveLinq.4 Assembly](#)

## Classes

### CompiledQuery

[C1.LiveLinq Namespace](#) : CompiledQuery Class

Provides for compilation and cache of queries for reuse.

## Object Model

CompiledQuery

## Syntax

Visual Basic (Declaration)	
<b>Public MustInherit NotInheritable Class</b> CompiledQuery	
C#	
<b>public static class</b> CompiledQuery	

## Remarks

If you need to execute the same query many times, each time with different parameter values, use the **CompiledQuery** class to improve performance. Without it, every query execution includes a compilation stage, that does not take much time but that time can accumulate to

significant numbers if it is repeated many times. The **CompiledQuery** class contains a single **Compile** method with several overloads. Call the **Compile** method to create a delegate to represent the compiled query.

## Inheritance Hierarchy

System.Object  
    **C1.LiveLinq.CompiledQuery**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[CompiledQuery Members](#)  
[C1.LiveLinq Namespace](#)

## Overview

[C1.LiveLinq Namespace](#) : CompiledQuery Class

Provides for compilation and cache of queries for reuse.

## Object Model

CompiledQuery

## Syntax

Visual Basic (Declaration)	
<b>Public MustInherit NotInheritable Class</b> CompiledQuery	
C#	
<b>public static class</b> CompiledQuery	

## Remarks

If you need to execute the same query many times, each time with different parameter values, use the **CompiledQuery** class to improve performance. Without it, every query execution includes a compilation stage, that does not take much time but that time can accumulate to significant numbers if it is repeated many times. The **CompiledQuery** class contains a single

**Compile** method with several overloads. Call the **Compile** method to create a delegate to represent the compiled query.

## Inheritance Hierarchy

System.Object  
    **C1.LiveLinq.CompiledQuery**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[CompiledQuery Members](#)  
[C1.LiveLinq Namespace](#)



## Members

[Methods](#)

[C1.LiveLinq Namespace](#) : CompiledQuery Class

The following tables list the members exposed by [CompiledQuery](#).

## Public Methods

	Name	Description
 	<a href="#">Compile</a>	Overloaded. Compiles the query.

[Top](#)


## See Also

### Reference

[CompiledQuery Class](#)  
[C1.LiveLinq Namespace](#)

## Methods

>

Name	Description
 <a href="#">Compile</a>	Overloaded. Compiles the query.

[Top](#)

## See Also

### Reference

[CompiledQuery Class](#)  
[C1.LiveLinq Namespace](#)

### Compile Method

[C1.LiveLinq Namespace](#) > [CompiledQuery Class](#) : Compile Method

Compiles the query.

## Overload List

Overload	Description
<a href="#">Compile&lt;T1,T2,T3,T4,TResult&gt;(Expression&lt;Func&lt;T1,T2,T3,T4,IIndexedSource&lt;TResult&gt;&gt;&gt;)</a>	Compiles the query.
<a href="#">Compile&lt;T1,T2,T3,TResult&gt;(Expression&lt;Func&lt;T1,T2,T3,IIndexedSource&lt;TResult&gt;&gt;&gt;)</a>	Compiles the query.
<a href="#">Compile&lt;T1,T2,TResult&gt;(Expression&lt;Func&lt;T1,T2,IIndexedSource&lt;TResult&gt;&gt;&gt;)</a>	Compiles the query.
<a href="#">Compile&lt;T,TResult&gt;(Expression&lt;Func&lt;T,IIndexedSource&lt;TResult&gt;&gt;&gt;)</a>	Compiles the query.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[CompiledQuery Class](#)

[CompiledQuery Members](#)

`Compile<T1,T2,T3,T4,TResult>(Expression<Func<T1,T2,T3,T4,IIndexedSource<TResult>>>)`  
Method

[C1.LiveLinq Namespace](#) > [CompiledQuery Class](#) > [Compile Method](#) :

`Compile<T1,T2,T3,T4,TResult>(Expression<Func<T1,T2,T3,T4,IIndexedSource<TResult>>>)` Method

The type of the first parameter that has to be passed in when executing the delegate returned by the Compile method.

The type of the second parameter that has to be passed in when executing the delegate returned by the Compile method.

The type of the third parameter that has to be passed in when executing the delegate returned by the Compile method.

The type of the fourth parameter that has to be passed in when executing the delegate returned by the Compile method.

The type of **TResult** in the [IIndexedSource<TResult>](#) returned when executing the delegate returned by the Compile method.

The query expression to be compiled.

Compiles the query.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Shared Function Compile     (Of T1,T2,T3,T4,TResult)( _         ByVal query As System.Linq.Expressions.Expression(Of Func(Of T1,T2,T3,T4,IIndexedSource(Of TResult)))) _ ) As System.Func(Of T1,T2,T3,T4,IIndexedSource(Of TResult))</pre>	
C#	
<pre>public static System.Func&lt;T1,T2,T3,T4,IIndexedSource&lt;TResult&gt;&gt;</pre>	



```
Compile<T1,T2,T3,T4,TResult>(
    System.Linq.Expressions.Expression<Func<T1,T2,T3,T4,IIndexedSource<TResult>>>
    query
)
```

## Parameters

*query*

The query expression to be compiled.

## Type Parameters

*T1*

The type of the first parameter that has to be passed in when executing the delegate returned by the Compile method.

*T2*

The type of the second parameter that has to be passed in when executing the delegate returned by the Compile method.

*T3*

The type of the third parameter that has to be passed in when executing the delegate returned by the Compile method.

*T4*

The type of the fourth parameter that has to be passed in when executing the delegate returned by the Compile method.

*TResult*

The type of **TResult** in the [IIndexedSource<TResult>](#) returned when executing the delegate returned by the Compile method.

## Return Value

The delegate to be called to execute the compiled query with particular parameter values.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[CompiledQuery Class](#)  
[CompiledQuery Members](#)  
[Overload List](#)

`Compile<T1,T2,T3,TResult>(Expression<Func<T1,T2,T3,IIndexedSource<TResult>>>) Method`

[C1.LiveLinq Namespace](#) > [CompiledQuery Class](#) > [Compile Method](#) :

`Compile<T1,T2,T3,TResult>(Expression<Func<T1,T2,T3,IIndexedSource<TResult>>>) Method`

The type of the first parameter that has to be passed in when executing the delegate returned by the Compile method.

The type of the second parameter that has to be passed in when executing the delegate returned by the Compile method.

The type of the third parameter that has to be passed in when executing the delegate returned by the Compile method.

The type of **TResult** in the [IIndexedSource<TResult>](#) returned when executing the delegate returned by the Compile method.

The query expression to be compiled.

Compiles the query.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Shared Function Compile
    (Of T1,T2,T3,TResult)( _
        ByVal query As System.Linq.Expressions.Expression(Of Func(Of
T1,T2,T3,IIndexedSource(Of TResult)))) _
) As System.Func(Of T1,T2,T3,IIndexedSource(Of TResult))
```

C#

```
public static System.Func<T1,T2,T3,IIndexedSource<TResult>>
Compile<T1,T2,T3,TResult>(
    System.Linq.Expressions.Expression<Func<T1,T2,T3,IIndexedSource<TResult>>>
query
)
```

### Parameters

*query*

The query expression to be compiled.

## Type Parameters

*T1*

The type of the first parameter that has to be passed in when executing the delegate returned by the Compile method.

*T2*

The type of the second parameter that has to be passed in when executing the delegate returned by the Compile method.

*T3*

The type of the third parameter that has to be passed in when executing the delegate returned by the Compile method.

*TResult*

The type of **TResult** in the [IndexedSource<TResult>](#) returned when executing the delegate returned by the Compile method.

## Return Value

The delegate to be called to execute the compiled query with particular parameter values.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[CompiledQuery Class](#)

[CompiledQuery Members](#)

[Overload List](#)

[Compile<T1,T2,TResult>\(Expression<Func<T1,T2,IIndexedSource<TResult>>>\) Method](#)

[C1.LiveLinq Namespace](#) > [CompiledQuery Class](#) > [Compile Method](#) :

[Compile<T1,T2,TResult>\(Expression<Func<T1,T2,IIndexedSource<TResult>>>\) Method](#)

The type of the first parameter that has to be passed in when executing the delegate returned by the Compile method.

The type of the second parameter that has to be passed in when executing the delegate returned by the Compile method.

The type of **TResult** in the [IIndexedSource<TResult>](#) returned when executing the delegate returned by the Compile method.

The query expression to be compiled.

Compiles the query.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Shared Function Compile     (Of T1,T2,TResult)( _     ByVal query As System.Linq.Expressions.Expression(Of Func(Of T1,T2,IIndexedSource(Of TResult))) _ ) As System.Func(Of T1,T2,IIndexedSource(Of TResult))</pre>	
C#	
<pre>public static System.Func&lt;T1,T2,IIndexedSource&lt;TResult&gt;&gt; Compile&lt;T1,T2,TResult&gt;(     System.Linq.Expressions.Expression&lt;Func&lt;T1,T2,IIndexedSource&lt;TResult&gt;&gt;&gt; query )</pre>	

### Parameters

*query*

The query expression to be compiled.

### Type Parameters

*T1*

The type of the first parameter that has to be passed in when executing the delegate returned by the Compile method.

*T2*

The type of the second parameter that has to be passed in when executing the delegate returned by the Compile method.

*TResult*

The type of **TResult** in the [IIndexedSource<TResult>](#) returned when executing the delegate returned by the Compile method.

## Return Value

The delegate to be called to execute the compiled query with particular parameter values.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[CompiledQuery Class](#)  
[CompiledQuery Members](#)  
[Overload List](#)

Compile<T,TResult>(Expression<Func<T,IIndexedSource<TResult>>>) Method

[C1.LiveLinq Namespace](#) > [CompiledQuery Class](#) > [Compile Method](#) :

Compile<T,TResult>(Expression<Func<T,IIndexedSource<TResult>>>) Method

The type of the parameter that has to be passed in when executing the delegate returned by the Compile method.

The type of **TResult** in the [IIndexedSource<TResult>](#) returned when executing the delegate returned by the Compile method.

The query expression to be compiled.

Compiles the query.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Shared Function Compile  
    (Of T,TResult)( _  
        ByVal query As System.Linq.Expressions.Expression(Of Func(Of  
T,IIndexedSource(Of TResult))) _  
    ) As System.Func(Of T,IIndexedSource(Of TResult))
```

C#

```
public static System.Func<T,IIndexedSource<TResult>> Compile<T,TResult>(
    System.Linq.Expressions.Expression<Func<T,IIndexedSource<TResult>>> query
)
```

## Parameters

*query*

The query expression to be compiled.

## Type Parameters

*T*

The type of the parameter that has to be passed in when executing the delegate returned by the Compile method.

*TResult*

The type of **TResult** in the [IIndexedSource<TResult>](#) returned when executing the delegate returned by the Compile method.

## Return Value

The delegate to be called to execute the compiled query with particular parameter values.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[CompiledQuery Class](#)

[CompiledQuery Members](#)

[Overload List](#)

## DeletedStateIsAvailableAttribute

[C1.LiveLinq Namespace](#) : DeletedStateIsAvailableAttribute Class

Indicates that the properties of an object of a class still return correct property values after the object has been deleted from the collection it belonged to.

# Object Model

DeletedStateIsAvailableAttribute

## Syntax

Visual Basic (Declaration)

```
<System.AttributeUsageAttribute(ValidOn=AttributeTargets.Class,  
    AllowMultiple=False,  
    Inherited=True)>  
Public NotInheritable Class DeletedStateIsAvailableAttribute  
    Inherits System.Attribute
```

C#

```
[System.AttributeUsage(ValidOn=AttributeTargets.Class,  
    AllowMultiple=false,  
    Inherited=true)]  
public sealed class DeletedStateIsAvailableAttribute : System.Attribute
```

## Remarks

This attribute is used with classes of elements of [C1.LiveLinq.Collections.IndexedCollection<T>](#) and other collections implementing [IObservableSource<T>](#). It gives you a way of changing the value of [IObservableSource<T>.IsDeletedStateAvailable](#) without having to create a full-blown custom implementation of the [IObservableSource<T>](#) interface.

## Inheritance Hierarchy

System.Object  
    System.Attribute  
        **C1.LiveLinq.DeletedStateIsAvailableAttribute**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DeletedStateIsAvailableAttribute Members](#)  
[C1.LiveLinq Namespace](#)

## Overview

[C1.LiveLinq Namespace](#) : DeletedStateIsAvailableAttribute Class

Indicates that the properties of an object of a class still return correct property values after the object has been deleted from the collection it belonged to.

## Object Model

DeletedStateIsAvailableAttribute

## Syntax

Visual Basic (Declaration)

```
<System.AttributeUsageAttribute(ValidOn=AttributeTargets.Class,  
    AllowMultiple=False,  
    Inherited=True)>  
Public NotInheritable Class DeletedStateIsAvailableAttribute  
    Inherits System.Attribute
```

C#

```
[System.AttributeUsage(ValidOn=AttributeTargets.Class,  
    AllowMultiple=false,  
    Inherited=true)]  
public sealed class DeletedStateIsAvailableAttribute : System.Attribute
```

## Remarks

This attribute is used with classes of elements of [C1.LiveLinq.Collections.IndexedCollection<T>](#) and other collections implementing [IObservableSource<T>](#). It gives you a way of changing the value of [IObservableSource<T>.IsDeletedStateAvailable](#) without having to create a full-blown custom implementation of the [IObservableSource<T>](#) interface.

## Inheritance Hierarchy

System.Object

System.Attribute

**C1.LiveLinq.DeletedStateIsAvailableAttribute**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2



## See Also

### Reference

[DeletedStateIsAvailableAttribute Members](#)

[C1.LiveLinq Namespace](#)

## Members

[Properties](#) [Methods](#)

[C1.LiveLinq Namespace](#) : DeletedStateIsAvailableAttribute Class



The following tables list the members exposed by [DeletedStateIsAvailableAttribute](#).

## Public Constructors

	Name	Description
	<a href="#">DeletedStateIsAvailableAttribute Constructor</a>	Initializes a new instance of the <a href="#">DeletedStateIsAvailableAttribute</a> class.



[Top](#)



## Public Properties

	Name	Description
	<a href="#">IsAvailable</a>	<b>true</b> if the properties of an object of the class still return correct property values after the object has been deleted from the collection it belonged to.
	<a href="#">TypeId</a>	(Inherited from System.Attribute)

[Top](#)

## Public Methods

	Name	Description
	<a href="#">Equals</a>	(Inherited from System.Attribute)
	<a href="#">GetHashCode</a>	(Inherited from System.Attribute)

	<a href="#">IsDefaultAttribute</a>	(Inherited from System.Attribute)
	<a href="#">Match</a>	(Inherited from System.Attribute)

[Top](#)

## See Also

### Reference

[DeletedStateIsAvailableAttribute Class](#)  
[C1.LiveLinq Namespace](#)

## DeletedStateIsAvailableAttribute Constructor

[C1.LiveLinq Namespace](#) > [DeletedStateIsAvailableAttribute Class](#) : DeletedStateIsAvailableAttribute  
Constructor

**true** if the properties of an object of the class still return correct property values after the object has been deleted from the collection it belonged to.

Initializes a new instance of the [DeletedStateIsAvailableAttribute](#) class.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function New( _     ByVal isAvailable As System.Boolean _ )</pre>	
C#	
<pre>public DeletedStateIsAvailableAttribute(     System.bool isAvailable )</pre>	

### Parameters

*isAvailable*

**true** if the properties of an object of the class still return correct property values after the object has been deleted from the collection it belonged to.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



- [DeletedStateIsAvailableAttribute Class](#)
- [DeletedStateIsAvailableAttribute Members](#)

## Properties

[C1.LiveLinq Namespace](#) : DeletedStateIsAvailableAttribute Class

For a list of all members of this type, see [DeletedStateIsAvailableAttribute members](#).

## Public Properties

	Name	Description
	<a href="#">IsAvailable</a>	<b>true</b> if the properties of an object of the class still return correct property values after the object has been deleted from the collection it belonged to.
	<a href="#">TypeId</a>	(Inherited from System.Attribute)

[Top](#)

## See Also

### Reference

- [DeletedStateIsAvailableAttribute Class](#)
- [C1.LiveLinq Namespace](#)

### IsAvailable Property

[C1.LiveLinq Namespace](#) > [DeletedStateIsAvailableAttribute Class](#) : IsAvailable Property

**true** if the properties of an object of the class still return correct property values after the object has been deleted from the collection it belonged to.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<code>Public ReadOnly Property IsAvailable As System.Boolean</code>	
C#	
<code>public System.bool IsAvailable {get;}</code>	

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

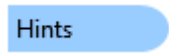
- [DeletedStateIsAvailableAttribute Class](#)
- [DeletedStateIsAvailableAttribute Members](#)

Hints

[C1.LiveLinq Namespace](#) : Hints Class

Provides a static (extension) method used as a hint that can be applied to an expression (usually, a property) in a query.

Object Model



Syntax

Visual Basic (Declaration)	
<code>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;</code> <code>&lt;HintAttribute(C1.LiveLinq.Hints+Converter)&gt;</code> <code>Public MustInherit NotInheritable Class Hints</code>	
C#	
<code>[System.Runtime.CompilerServices.Extension()]</code> <code>[Hint(C1.LiveLinq.Hints+Converter)]</code> <code>public static class Hints</code>	

Remarks

A hint does not change the value of the expression it is applied to, it only tells LiveLinq to create and use an index on that property, if possible.

**Note:** Hints can only be used in indexed queries (with **AsIndexed()** extension method). Using them in live views (with **AsLive()** extension method) will result in an exception.

See [How to create indexes](#).

## Inheritance Hierarchy

System.Object  
    **C1.LiveLinq.Hints**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Hints Members](#)  
[C1.LiveLinq Namespace](#)

## Overview

[C1.LiveLinq Namespace](#) : Hints Class

Provides a static (extension) method used as a hint that can be applied to an expression (usually, a property) in a query.

## Object Model

Hints

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; &lt;HintAttribute(C1.LiveLinq.Hints+Converter)&gt; Public MustInherit NotInheritable Class Hints</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension() [Hint(C1.LiveLinq.Hints+Converter)] public static class Hints</pre>	

## Remarks

A hint does not change the value of the expression it is applied to, it only tells LiveLinq to create and use an index on that property, if possible.

**Note:** Hints can only be used in indexed queries (with **AsIndexed()** extension method). Using them in live views (with **AsLive()** extension method) will result in an exception.

See How to create indexes.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Hints**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Hints Members](#)

[C1.LiveLinq Namespace](#)


## Members

[Properties](#) [Methods](#)

[C1.LiveLinq Namespace](#) : Hints Class

The following tables list the members exposed by [Hints](#).


## Public Properties

	Name	Description
 <b>S</b>	<a href="#">DefaultAction</a>	Gets or sets the default action for indexing hints.

[Top](#)

## Public Methods

	Name	Description
--	------	-------------

 <b>S</b>	<b>Indexed</b>	Overloaded. A hint with specified action.
--	----------------	---

[Top](#)

## See Also

### Reference

[Hints Class](#)


[C1.LiveLinq Namespace](#)

## Methods

[C1.LiveLinq Namespace](#) : [Hints Class](#)

For a list of all members of this type, see [Hints members](#).

## Public Methods

	Name	Description
 <b>S</b>	<b>Indexed</b>	Overloaded. A hint with specified action.

[Top](#)

## See Also

### Reference

[Hints Class](#)

[C1.LiveLinq Namespace](#)

### Indexed Method

[C1.LiveLinq Namespace](#) > [Hints Class](#) : Indexed Method

A hint with specified action.

## Overload List

Overload	Description
<a href="#">Indexed&lt;T&gt;(T,IndexingHintAction)</a>	A hint with specified action.
<a href="#">Indexed&lt;T&gt;(T)</a>	A hint with default action.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[Hints Class](#)  
[Hints Members](#)

Indexed<T>(T,IndexingHintAction) Method

[C1.LiveLinq Namespace](#) > [Hints Class](#) > [Indexed Method](#) : Indexed<T>(T,IndexingHintAction) Method

This can be any type, because hints are applicable to expressions of any type.

The value of the expression the hint is applied to. This value is not actually used by the hint, because the hint is never executed, its role is purely declarative.

The action specified by the hint.

A hint with specified action.

# Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function Indexed(Of T)( _     ByVal value As T, _     ByVal action As IndexingHintAction _ ) As T</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static T Indexed&lt;T&gt;(     T value,     IndexingHintAction action )</pre>	

## Parameters

*value*

The value of the expression the hint is applied to. This value is not actually used by the hint, because the hint is never executed, its role is purely declarative.

*action*



The action specified by the hint.

## Type Parameters

*T*

This can be any type, because hints are applicable to expressions of any type.

## Return Value

Formally, the hint returns the same value that it receives in the parameter. In fact, it is never executed, its role is purely declarative.

## Remarks

A hint does not change the value of the expression it is applied to. It only tells LiveLinq query optimizer to create and use an index on that expression, if possible. In fact, hints are removed from the expression before it is executed.

**See Also:** How to create indexes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Hints Class](#)

[Hints Members](#)

[Overload List](#)

Indexed<T>(T) Method

[C1.LiveLinq Namespace](#) > [Hints Class](#) > [Indexed Method](#) : Indexed<T>(T) Method

This can be any type, because hints are applicable to expressions of any type.

The value of the expression the hint is applied to. This value is not actually used by the hint, because the hint is never executed, its role is purely declarative.

A hint with default action.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function Indexed(Of T)( _
    ByVal value As T _
) As T
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static T Indexed<T>(
    T value
)
```

## Parameters

*value*

The value of the expression the hint is applied to. This value is not actually used by the hint, because the hint is never executed, its role is purely declarative.

## Type Parameters

*T*

This can be any type, because hints are applicable to expressions of any type.

## Return Value

Formally, the hint returns the same value that it receives in the parameter. In fact, it is never executed, its role is purely declarative.

## Remarks

Current default action is defined by the value of the static [DefaultAction](#) property. By default, it is [IndexingHintAction](#). **Optional.**

Use the other **Indexed** overload if you need a hint with non-default action.

A hint does not change the value of the expression it is applied to. It only tells LiveLinq query optimizer to create and use an index on that expression, if possible. In fact, hints are removed from the expression before it is executed.

**See Also:** How to create indexes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [Hints Class](#)
- [Hints Members](#)
- [Overload List](#)

## Properties

[C1.LiveLinq Namespace](#) : [Hints Class](#)

For a list of all members of this type, see [Hints members](#).

## Public Properties

	Name	Description
 <b>S</b>	<a href="#">DefaultAction</a>	Gets or sets the default action for indexing hints.

[Top](#)

# See Also

## Reference

- [Hints Class](#)
- [C1.LiveLinq Namespace](#)

## DefaultAction Property

[C1.LiveLinq Namespace](#) > [Hints Class](#) : [DefaultAction Property](#)

Gets or sets the default action for indexing hints.

## Syntax

Visual Basic (Declaration)	
<code>Public Shared Property DefaultAction As IndexingHintAction</code>	
C#	
<code>public static IndexingHintAction DefaultAction {get; set;}</code>	

## Remarks

The default is [IndexingHintAction](#).**Optional**. This setting is global for the application. If you want to change it, do it at application startup.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Hints Class](#)

[Hints Members](#)

## IndexedQueryExtensions

[C1.LiveLinq Namespace](#) : IndexedQueryExtensions Class

Provides a set of static (extension) methods for querying objects that implement [C1.LiveLinq.Indexing.IIndexedSource<T>](#).

## Object Model

IndexedQueryExtensions

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public MustInherit NotInheritable Class IndexedQueryExtensions</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static class IndexedQueryExtensions</pre>	

## Remarks

The methods in this class provide an implementation of the LINQ query operators for querying data sources that implement [C1.LiveLinq.Indexing.IIndexedSource<T>](#). Those data sources include LiveLinq to Objects, LiveLinq to DataSet and LiveLinq to XML, see [How to query collections with LiveLinq](#).

These implementations of query operators use indexing and other optimization techniques to speed up query execution.

Not all standard LINQ query operators are present here, but it does not mean that they cannot be used in the same query. For the operators that are not present here, standard LINQ implementations are used, because they don't require or don't allow optimization.

**Note:** Live views are also LiveLinq data sources, but they have their own implementations of query operators defined in the [C1.LiveLinq.LiveViews.View<T>](#) class. Live view implementations are heavier, require more resources, so it is not recommended to use live view implementations in cases where you don't need live view functionality (for example, for querying read-only collections), see Live View Performance. If you have a live view, but want to query it using operators from **IndexedQueryExtensions** instead of [C1.LiveLinq.LiveViews.View<T>](#), use [AsIndexed<T>](#) to "downgrade" the live view to an [C1.LiveLinq.Indexing.IIndexedSource<T>](#).

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.IndexedQueryExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Members](#)

[C1.LiveLinq Namespace](#)

## Overview

[C1.LiveLinq Namespace](#) : IndexedQueryExtensions Class

Provides a set of static (extension) methods for querying objects that implement [C1.LiveLinq.Indexing.IIndexedSource<T>](#).

## Object Model

IndexedQueryExtensions

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public MustInherit NotInheritable Class IndexedQueryExtensions
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static class IndexedQueryExtensions
```

## Remarks

The methods in this class provide an implementation of the LINQ query operators for querying data sources that implement [C1.LiveLinq.Indexing.IIndexedSource<T>](#). Those data sources include LiveLinq to Objects, LiveLinq to DataSet and LiveLinq to XML, see [How to query collections with LiveLinq](#).

These implementations of query operators use indexing and other optimization techniques to speed up query execution.

Not all standard LINQ query operators are present here, but it does not mean that they cannot be used in the same query. For the operators that are not present here, standard LINQ implementations are used, because they don't require or don't allow optimization.

**Note:** Live views are also LiveLinq data sources, but they have their own implementations of query operators defined in the [C1.LiveLinq.LiveViews.View<T>](#) class. Live view implementations are heavier, require more resources, so it is not recommended to use live view implementations in cases where you don't need live view functionality (for example, for querying read-only collections), see [Live View Performance](#). If you have a live view, but want to query it using operators from **IndexedQueryExtensions** instead of [C1.LiveLinq.LiveViews.View<T>](#), use [AsIndexed<T>](#) to "downgrade" the live view to an [C1.LiveLinq.Indexing.IIndexedSource<T>](#).

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.IndexedQueryExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Members](#)  
[C1.LiveLinq Namespace](#)

## Members



[Methods](#)

C1.LiveLinq Namespace : IndexedQueryExtensions Class

The following tables list the members exposed by [IndexedQueryExtensions](#).

## Public Methods

	Name	Description
≡💎 S	<a href="#">AsIndexed&lt;T&gt;</a>	Returns the input typed as <a href="#">C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</a> .
≡💎 S	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a collection according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
≡💎 S	<a href="#">GroupJoin&lt;TOuter,TInner,TKey,TResult&gt;</a>	Correlates the elements of two collections based on equality of keys and groups the results.
≡💎 S	<a href="#">Join</a>	Overloaded. Correlates the elements of two collections based on matching keys.
≡💎 S	<a href="#">OrderBy&lt;T,TKey&gt;</a>	Sorts the elements of a collection in ascending order.
≡💎 S	<a href="#">OrderByDescending&lt;T,TKey&gt;</a>	Sorts the elements of a collection in descending order.
≡💎 S	<a href="#">Select&lt;TSource,TResult&gt;</a>	Projects each element of a collection into a new form.
≡💎 S	<a href="#">SelectMany</a>	Overloaded. Projects each element of a collection to a sequence of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.

≡  S	<a href="#">ToIndexed</a>	Overloaded. Creates an <a href="#">C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</a> based on the specified <a href="#">IObservableSource&lt;T&gt;</a> collection.
≡  S	<a href="#">Where</a>	Overloaded. Filters the source collection based on a predicate.

[Top](#)

## See Also

### Reference





[IndexedQueryExtensions Class](#)  
[C1.LiveLinq Namespace](#)

## Methods

[C1.LiveLinq Namespace](#) : [IndexedQueryExtensions Class](#)

For a list of all members of this type, see [IndexedQueryExtensions members](#).

## Public Methods

	Name	Description
≡  S	<a href="#">AsIndexed&lt;T&gt;</a>	Returns the input typed as <a href="#">C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</a> .
≡  S	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a collection according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
≡  S	<a href="#">GroupJoin&lt;TOuter,TInner,TKey,TResult&gt;</a>	Correlates the elements of two collections based on equality of keys and groups the results.
≡  S	<a href="#">Join</a>	Overloaded. Correlates the elements of two



		collections based on matching keys.
≡💎 S	<a href="#">OrderBy&lt;T,TKey&gt;</a>	Sorts the elements of a collection in ascending order.
≡💎 S	<a href="#">OrderByDescending&lt;T,TKey&gt;</a>	Sorts the elements of a collection in descending order.
≡💎 S	<a href="#">Select&lt;TSource,TResult&gt;</a>	Projects each element of a collection into a new form.
≡💎 S	<a href="#">SelectMany</a>	Overloaded. Projects each element of a collection to a sequence of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.
≡💎 S	<a href="#">ToIndexed</a>	Overloaded. Creates an <a href="#">C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</a> based on the specified <a href="#">IObservableSource&lt;T&gt;</a> collection.
≡💎 S	<a href="#">Where</a>	Overloaded. Filters the source collection based on a predicate.

[Top](#)

## See Also

### Reference

[IndexedQueryExtensions Class](#)  
[C1.LiveLinq Namespace](#)

### AsIndexed<T> Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) : [AsIndexed<T> Method](#)

The type of the elements of *source*.

The collection to type as [C1.LiveLinq.Indexing.IIndexedSource<T>](#).

Returns the input typed as [C1.LiveLinq.Indexing.IIndexedSource<T>](#).

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>>
Public Shared Function AsIndexed(Of T)( _
    ByVal source As IIndexedSource(Of T) _
) As IIndexedSource(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static IIndexedSource<T> AsIndexed<T>(
    IIndexedSource<T> source
)
```

### Parameters

*source*

The collection to type as [C1.LiveLinq.Indexing.IIndexedSource<T>](#).

### Type Parameters

*T*

The type of the elements of *source*.

### Return Value

The input collection typed as [C1.LiveLinq.Indexing.IIndexedSource<T>](#).

## Remarks

This method has no effect other than to change the compile-time type of *source* from a type that implements [C1.LiveLinq.Indexing.IIndexedSource<T>](#) to [C1.LiveLinq.Indexing.IIndexedSource<T>](#) itself. It is used to choose between query implementations when a collection implements [C1.LiveLinq.Indexing.IIndexedSource<T>](#) but also has a different set of public query methods available.

The main scenario is when you want to perform queries on live views without creating new live views.

Live views have their own implementations of query operators defined in the [C1.LiveLinq.LiveViews.View<T>](#) class. Live view implementations are heavier, require more resources, so it is not recommended to use live view implementations in cases where you don't need live view functionality (for example, for querying read-only

collections), see Live View Performance. If you have a live view, but want to query it using operators from [IndexedQueryExtensions](#) instead of [C1.LiveLinq.LiveViews.View<T>](#), use this method to "downgrade" the live view to an [C1.LiveLinq.Indexing.IIndexedSource<T>](#). This method does not cause the live view to lose its "live" functionality, it simply returns the live view's implementation of the [C1.LiveLinq.Indexing.IIndexedSource<T>](#) interface.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)

### GroupBy Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) : GroupBy Method

Groups the elements of a collection according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.

## Overload List

Overload	Description
<a href="#">GroupBy&lt;TSource,TKey,TElement,TResult&gt;(IIndexedSource&lt;TSource&gt;,Expression&lt;Func&lt;TSource,TKey&gt;&gt;,Expression&lt;Func&lt;TSource,TElement&gt;&gt;,Expression&lt;Func&lt;TKey,IEnumerable&lt;TElement&gt;,TResult&gt;&gt;)</a>	Groups the elements of a collection according

	rdin g to a spec ified key selec tor func tion and creat es a resul t valu e from each grou p and its key. The elem ents of each grou p are proj ecte d by usin g a
--	---

	specified function.
<a href="#">GroupBy&lt;TSource,TKey,TElement&gt;(IIndexedSource&lt;TSource&gt;,Expression&lt;Func&lt;TSource,TKey&gt;&gt;,Expression&lt;Func&lt;TSource,TElement&gt;&gt;)</a>	Groups the elements of a collection according to a specified key selector function and projects the elements for each group by using

	g a spec ified func tion.
GroupBy<TSource,TKey,TResult>(IndexedSource<TSource>,Expression<Func<TSource,T Key>>,Expression<Func<TKey,IEnumerable<TSource>,TResult>>)	Groups the elem ents of a colle ctio n acco rdin g to a spec ified key selec tor func tion and creat es a resul t valu e from each grou

	p and its key.
<a href="#">GroupBy&lt;TSource,TKey&gt;(IIndexedSource&lt;TSource&gt;,Expression&lt;Func&lt;TSource,TKey&gt;&gt;)</a>	Grou ps the elem ents of a colle ctio n acco rdin g to a spec ified key selec tor func tion.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)

[IndexedQueryExtensions Members](#)

GroupBy<TSource,TKey,TElement,TResult>(IIndexedSource<TSource>,Expression<Func<TSource,TKey>>,Expression<Func<TSource,TElement>>,Expression<Func<TKey,IEnumerable<TElement>>,TResult>>) Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [GroupBy Method](#) :

GroupBy<TSource,TKey,TElement,TResult>(IIndexedSource<TSource>,Expression<Func<TSource,TKey>>,Expression<Func<TSource,TElement>>,Expression<Func<TKey,IEnumerable<TElement>,TResult>>) Method

The type of the elements of *source*.

The type of the key returned by *keySelector*.

The type of the elements in the **System.Linq.IGrouping`2**.

The type of the result value returned by *resultSelector*

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) whose elements to group

A function to extract the key for each element.

A function to map each source element to an element in the **System.Linq.IGrouping`2**.

A function to create a result value from each group.

Groups the elements of a collection according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute()&gt; Public Overloads Shared Function GroupBy     (Of TSource,TKey,TElement,TResult)( _     ByVal source As IIndexedSource(Of TSource), _     ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of TSource,TKey)), _     ByVal elementSelector As System.Linq.Expressions.Expression(Of Func(Of TSource,TElement)), _     ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of TKey,IEnumerable(Of TElement),TResult)) _ ) As IIndexedSource(Of TResult)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static IIndexedSource&lt;TResult&gt; GroupBy&lt;TSource,TKey,TElement,TResult&gt;(</pre>	



```

IIndexedSource<TSource> source,
System.Linq.Expressions.Expression<Func<TSource,TKey>> keySelector,
System.Linq.Expressions.Expression<Func<TSource,TElement>>
elementSelector,

System.Linq.Expressions.Expression<Func<TKey,IEnumerable<TElement>,TResult>>
resultSelector
)

```

## Parameters

*source*

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) whose elements to group

*keySelector*

A function to extract the key for each element.

*elementSelector*

A function to map each source element to an element in the **System.Linq.IGrouping`2**.

*resultSelector*

A function to create a result value from each group.

## Type Parameters

*TSource*

The type of the elements of *source*.

*TKey*

The type of the key returned by *keySelector*.

*TElement*

The type of the elements in the **System.Linq.IGrouping`2**.

*TResult*

The type of the result value returned by *resultSelector*

## Return Value

A collection of elements of type *TResult* where each element represents a projection over a group and its key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)

[IndexedQueryExtensions Members](#)

[Overload List](#)

`GroupBy<TSource,TKey,TElement>(IIndexedSource<TSource>,Expression<Func<TSource,TKey>>,Expression<Func<TSource,TElement>>)` Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [GroupBy Method](#) :

`GroupBy<TSource,TKey,TElement>(IIndexedSource<TSource>,Expression<Func<TSource,TKey>>,Expression<Func<TSource,TElement>>)` Method

The type of the elements of *source*.

The type of the key returned by *keySelector*.

The type of the elements in the **System.Linq.IGrouping`2**.

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) whose elements to group

A function to extract the key for each element.

A function to map each source element to an element in the **System.Linq.IGrouping`2**.

Groups the elements of a collection according to a specified key selector function and projects the elements for each group by using a specified function.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function GroupBy
    (Of TSource,TKey,TElement)( _
    ByVal source As IIndexedSource(Of TSource), _
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of
TSource,TKey)), _
```

```

    ByVal elementSelector As System.Linq.Expressions.Expression(Of Func(Of
TSource, TElement)) _
) As IIndexedSource(Of IGrouping(Of TKey, TElement))

```

C#

```

[System.Runtime.CompilerServices.Extension()]
public static IIndexedSource<IGrouping<TKey, TElement>>
GroupBy<TSource, TKey, TElement>(
    IIndexedSource<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource, TKey>> keySelector,
    System.Linq.Expressions.Expression<Func<TSource, TElement>> elementSelector
)

```

## Parameters

*source*

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) whose elements to group

*keySelector*

A function to extract the key for each element.

*elementSelector*

A function to map each source element to an element in the **System.Linq.IGrouping`2**.

## Type Parameters

*TSource*

The type of the elements of *source*.

*TKey*

The type of the key returned by *keySelector*.

*TElement*

The type of the elements in the **System.Linq.IGrouping`2**.

## Return Value

A collection of **System.Linq.IGrouping`2** objects each containing a collection of objects of type *TElement* and a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not

supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

GroupBy<TSource,TKey,TResult>(IIndexedSource<TSource>,Expression<Func<TSource,TKey>>,Expression<Func<TKey,IEnumerable<TSource>,TResult>>) Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [GroupBy Method](#) :

GroupBy<TSource,TKey,TResult>(IIndexedSource<TSource>,Expression<Func<TSource,TKey>>,Expression<Func<TKey,IEnumerable<TSource>,TResult>>) Method

The type of the elements of *source*.

The type of the key returned by *keySelector*.

The type of the result value returned by *resultSelector*

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) whose elements to group

A function to extract the key for each element.

A function to create a result value from each group.

Groups the elements of a collection according to a specified key selector function and creates a result value from each group and its key.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function GroupBy
    (Of TSource,TKey,TResult)( _
    ByVal source As IIndexedSource(Of TSource), _
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of
TSource,TKey)), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
TKey,IEnumerable(Of TSource),TResult)) _
) As IIndexedSource(Of TResult)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static IIndexedSource<TResult> GroupBy<TSource, TKey, TResult>(
    IIndexedSource<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource, TKey>> keySelector,
    System.Linq.Expressions.Expression<Func<TKey, IEnumerable<TSource>, TResult>>
    resultSelector
)
```

## Parameters

*source*

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) whose elements to group

*keySelector*

A function to extract the key for each element.

*resultSelector*

A function to create a result value from each group.

## Type Parameters

*TSource*

The type of the elements of *source*.

*TKey*

The type of the key returned by *keySelector*.

*TResult*

The type of the result value returned by *resultSelector*

## Return Value

A collection of elements of type *TResult* where each element represents a projection over a group and its key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

GroupBy<TSource,TKey>(IIndexedSource<TSource>,Expression<Func<TSource,TKey>>)  
Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [GroupBy Method](#) :

GroupBy<TSource,TKey>(IIndexedSource<TSource>,Expression<Func<TSource,TKey>>) Method

The type of the elements of *source*.

The type of the key returned by *keySelector*.

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) whose elements to group

A function to extract the key for each element.

Groups the elements of a collection according to a specified key selector function.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function GroupBy
    (Of TSource,TKey)( _
    ByVal source As IIndexedSource(Of TSource), _
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of
TSource,TKey)) _
) As IIndexedSource(Of IGrouping(Of TKey,TSource))
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static IIndexedSource<IGrouping<TKey,TSource>> GroupBy<TSource,TKey>(
    IIndexedSource<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,TKey>> keySelector
)
```

## Parameters

*source*

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) whose elements to group

*keySelector*

A function to extract the key for each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

*TKey*

The type of the key returned by *keySelector*.

## Return Value

A collection of **System.Linq.IGrouping`2** objects each containing a sequence of objects and a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

## GroupJoin<TOuter,TInner,TKey,TResult> Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) : GroupJoin<TOuter,TInner,TKey,TResult> Method

The type of the elements of the first collection.

The type of the elements of the second collection.

The type of the keys returned by the key selector functions.

The type of the result elements.

The first collection to join.

The collection to join to the first collection.

A function to extract the join key from each element of the first collection.

A function to extract the join key from each element of the second collection.

A function to create a result element from an element from the first collection and a collection of matching elements from the second collection.

Correlates the elements of two collections based on equality of keys and groups the results.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Shared Function GroupJoin
    (Of TOuter,TInner,TKey,TResult)( _
    ByVal outer As IIndexedSource(Of TOuter), _
    ByVal inner As System.Collections.Generic.IEnumerable(Of TInner), _
    ByVal outerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
TOuter,TKey)), _
    ByVal innerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
TInner,TKey)), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
TOuter,IEnumerable(Of TInner),TResult)) _
) As IIndexedSource(Of TResult)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]
public static IIndexedSource<TResult> GroupJoin<TOuter,TInner,TKey,TResult>(
    IIndexedSource<TOuter> outer,
    System.Collections.Generic.IEnumerable<TInner> inner,
    System.Linq.Expressions.Expression<Func<TOuter,TKey>> outerKeySelector,
    System.Linq.Expressions.Expression<Func<TInner,TKey>> innerKeySelector,
    System.Linq.Expressions.Expression<Func<TOuter,IEnumerable<TInner>,TResult>>
    resultSelector
)
```

## Parameters

*outer*

The first collection to join.

*inner*

The collection to join to the first collection.

*outerKeySelector*



A function to extract the join key from each element of the first collection.

*innerKeySelector*

A function to extract the join key from each element of the second collection.

*resultSelector*

A function to create a result element from an element from the first collection and a collection of matching elements from the second collection.

## Type Parameters

*TOuter*

The type of the elements of the first collection.

*TInner*

The type of the elements of the second collection.

*TKey*

The type of the keys returned by the key selector functions.

*TResult*

The type of the result elements.

## Return Value

An [IIndexedSource<TResult>](#) that contains elements of type *TResult* that are obtained by performing a grouped join on two collections.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexedQueryExtensions Class](#)

[IndexedQueryExtensions Members](#)

## Join Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) : Join Method

Correlates the elements of two collections based on matching keys.

## Overload List

Overload	Description
<a href="#">Join&lt;TOuter,TInner,TKey,TResult&gt;(IIndexedSource&lt;TOuter&gt;,IEnumerable&lt;TInner&gt;,Expression&lt;Func&lt;TOuter,TKey&gt;&gt;,Expression&lt;Func&lt;TInner,TKey&gt;&gt;,Expression&lt;Func&lt;TOuter,TInner,TResult&gt;&gt;)</a>	Correlates the elements of two collections based on matching keys.
<a href="#">Join&lt;TOuter,TInner,TKey,TResult&gt;(IEnumerable&lt;TOuter&gt;,IIndexedSource&lt;TInner&gt;,Expression&lt;Func&lt;TOuter,TKey&gt;&gt;,Expression&lt;Func&lt;TInner,TKey&gt;&gt;,Expression&lt;Func&lt;TOuter,TInner,TResult&gt;&gt;)</a>	Correlates the elements of two collections based on matching keys.

	d on matc hing keys.
<a href="#">Join&lt;TOuter,TInner,TKey,TResult&gt;(IIndexedSource&lt;TOuter&gt;,IIndexedSource&lt;TInner&gt;,Expression&lt;Func&lt;TOuter,TKey&gt;&gt;,Expression&lt;Func&lt;TInner,TKey&gt;&gt;,Expression&lt;Func&lt;TOuter,TInner,TResult&gt;&gt;)</a>	Corr elate s the elem ents of two colle ction s base d on matc hing keys.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)

[IndexedQueryExtensions Members](#)

[Join<TOuter,TInner,TKey,TResult>\(IIndexedSource<TOuter>,IEnumerable<TInner>,Expression<Func<TOuter,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<TOuter,TInner,TResult>>\)](#) Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [Join Method](#) :

[Join<TOuter,TInner,TKey,TResult>\(IIndexedSource<TOuter>,IEnumerable<TInner>,Expression<Func<TOut](#)

er,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<TOuter,TInner,TResult>>) Method

The type of the elements of the first collection.

The type of the elements of the second collection.

The type of the keys returned by the key selector functions.

The type of the result elements.

The first collection to join.

The second collection to join.

A function to extract the join key from each element of the first collection.

A function to extract the join key from each element of the second collection.

A function to create a result element from two matching elements.

Correlates the elements of two collections based on matching keys.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function Join     (Of TOuter,TInner,TKey,TResult)( _     ByVal outer As IIndexedSource(Of TOuter), _     ByVal inner As System.Collections.Generic.IEnumerable(Of TInner), _     ByVal outerKeySelector As System.Linq.Expressions.Expression(Of Func(Of TOuter,TKey)), _     ByVal innerKeySelector As System.Linq.Expressions.Expression(Of Func(Of TInner,TKey)), _     ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of TOuter,TInner,TResult)) _ ) As IIndexedSource(Of TResult)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static IIndexedSource&lt;TResult&gt; Join&lt;TOuter,TInner,TKey,TResult&gt;(     IIndexedSource&lt;TOuter&gt; outer,     System.Collections.Generic.IEnumerable&lt;TInner&gt; inner,     System.Linq.Expressions.Expression&lt;Func&lt;TOuter,TKey&gt;&gt; outerKeySelector,     System.Linq.Expressions.Expression&lt;Func&lt;TInner,TKey&gt;&gt; innerKeySelector,</pre>	

```
System.Linq.Expressions.Expression<Func<TOuter,TInner,TResult>>  
resultSelector  
)
```

## Parameters

*outer*

The first collection to join.

*inner*

The second collection to join.

*outerKeySelector*

A function to extract the join key from each element of the first collection.

*innerKeySelector*

A function to extract the join key from each element of the second collection.

*resultSelector*

A function to create a result element from two matching elements.

## Type Parameters

*TOuter*

The type of the elements of the first collection.

*TInner*

The type of the elements of the second collection.

*TKey*

The type of the keys returned by the key selector functions.

*TResult*

The type of the result elements.

## Return Value

An [IIndexedSource<TResult>](#) that has elements of type *TResult* that are obtained by performing an inner join on two collections.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

Join<TOuter,TInner,TKey,TResult>(IEnumerable<TOuter>,IIndexedSource<TInner>,Expression<Func<TOuter,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<TOuter,TInner,TResult>>) Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [Join Method](#) :

Join<TOuter,TInner,TKey,TResult>(IEnumerable<TOuter>,IIndexedSource<TInner>,Expression<Func<TOuter,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<TOuter,TInner,TResult>>) Method

The type of the elements of the first collection.

The type of the elements of the second collection.

The type of the keys returned by the key selector functions.

The type of the result elements.

The first collection to join.

The second collection to join.

A function to extract the join key from each element of the first collection.

A function to extract the join key from each element of the second collection.

A function to create a result element from two matching elements.

Correlates the elements of two collections based on matching keys.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
```

```
Public Overloads Shared Function Join
```

```
(Of TOuter,TInner,TKey,TResult)( _
```

```
ByVal outer As System.Collections.Generic.IEnumerable(Of TOuter), _
```

```

    ByVal inner As IIndexedSource(Of TInner), _
    ByVal outerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
TOuter,TKey)), _
    ByVal innerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
TInner,TKey)), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
TOuter,TInner,TResult)) _
) As IIndexedSource(Of TResult)

```

C#

```

[System.Runtime.CompilerServices.Extension()]
public static IIndexedSource<TResult> Join<TOuter,TInner,TKey,TResult>(
    System.Collections.Generic.IEnumerable<TOuter> outer,
    IIndexedSource<TInner> inner,
    System.Linq.Expressions.Expression<Func<TOuter,TKey>> outerKeySelector,
    System.Linq.Expressions.Expression<Func<TInner,TKey>> innerKeySelector,
    System.Linq.Expressions.Expression<Func<TOuter,TInner,TResult>>
resultSelector
)

```

## Parameters

*outer*

The first collection to join.

*inner*

The second collection to join.

*outerKeySelector*

A function to extract the join key from each element of the first collection.

*innerKeySelector*

A function to extract the join key from each element of the second collection.

*resultSelector*

A function to create a result element from two matching elements.

## Type Parameters

*TOuter*

The type of the elements of the first collection.

*TInner*

The type of the elements of the second collection.

*TKey*

The type of the keys returned by the key selector functions.

*TResult*

The type of the result elements.

## Return Value

An [IIndexedSource<TResult>](#) that has elements of type *TResult* that are obtained by performing an inner join on two collections.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

`Join<TOuter,TInner,TKey,TResult>(IIndexedSource<TOuter>,IIndexedSource<TInner>,Expression<Func<TOuter,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<TOuter,TInner,TResult>>)` Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [Join Method](#) :

`Join<TOuter,TInner,TKey,TResult>(IIndexedSource<TOuter>,IIndexedSource<TInner>,Expression<Func<TOuter,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<TOuter,TInner,TResult>>)` Method

The type of the elements of the first collection.

The type of the elements of the second collection.

The type of the keys returned by the key selector functions.

The type of the result elements.

The first collection to join.

The second collection to join.



A function to extract the join key from each element of the first collection.

A function to extract the join key from each element of the second collection.

A function to create a result element from two matching elements.

Correlates the elements of two collections based on matching keys.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function Join
    (Of TOuter,TInner,TKey,TResult)( _
    ByVal outer As IIndexedSource(Of TOuter), _
    ByVal inner As IIndexedSource(Of TInner), _
    ByVal outerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
TOuter,TKey)), _
    ByVal innerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
TInner,TKey)), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
TOuter,TInner,TResult))) _
) As IIndexedSource(Of TResult)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]
public static IIndexedSource<TResult> Join<TOuter,TInner,TKey,TResult>(
    IIndexedSource<TOuter> outer,
    IIndexedSource<TInner> inner,
    System.Linq.Expressions.Expression<Func<TOuter,TKey>> outerKeySelector,
    System.Linq.Expressions.Expression<Func<TInner,TKey>> innerKeySelector,
    System.Linq.Expressions.Expression<Func<TOuter,TInner,TResult>>
resultSelector
)
```

## Parameters

*outer*

The first collection to join.

*inner*

The second collection to join.

*outerKeySelector*

A function to extract the join key from each element of the first collection.

*innerKeySelector*

A function to extract the join key from each element of the second collection.

*resultSelector*

A function to create a result element from two matching elements.

## Type Parameters

*TOuter*

The type of the elements of the first collection.

*TInner*

The type of the elements of the second collection.

*TKey*

The type of the keys returned by the key selector functions.

*TResult*

The type of the result elements.

## Return Value

An [IIndexedSource<TResult>](#) that has elements of type *TResult* that are obtained by performing an inner join on two collections.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

## OrderBy<T,TKey> Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) : OrderBy<T,TKey> Method

The type of the elements of *source*.

The type of the key returned by *keySelector*.

A collection of values to order.

A function to extract a key from an element.

Sorts the elements of a collection in ascending order.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Shared Function OrderBy
    (Of T,TKey)( _
    ByVal source As IIndexedSource(Of T), _
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of
T,TKey)) _
) As Ordering(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static Ordering<T> OrderBy<T,TKey>(
    IIndexedSource<T> source,
    System.Linq.Expressions.Expression<Func<T,TKey>> keySelector
)
```

## Parameters

*source*

A collection of values to order.

*keySelector*

A function to extract a key from an element.

## Type Parameters

*T*

The type of the elements of *source*.

*TKey*

The type of the key returned by *keySelector*.

## Return Value

An [C1.LiveLinq.Indexing.IndexedSource<T>](#) whose elements are sorted according to a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)

## OrderByDescending<T,TKey> Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) : OrderByDescending<T,TKey> Method

The type of the elements of *source*.

The type of the key returned by *keySelector*.

A collection of values to order.

A function to extract a key from an element.

Sorts the elements of a collection in descending order.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Shared Function OrderByDescending  
    (Of T,TKey)( _  
    ByVal source As IIndexedSource(Of T), _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T,TKey)) _  
    ) As Ordering(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static Ordering<T> OrderByDescending<T, TKey>(
    IIndexedSource<T> source,
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector
)
```

## Parameters

*source*

A collection of values to order.

*keySelector*

A function to extract a key from an element.

## Type Parameters

*T*

The type of the elements of *source*.

*TKey*

The type of the key returned by *keySelector*.

## Return Value

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) whose elements are sorted in descending order according to a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexedQueryExtensions Class](#)

[IndexedQueryExtensions Members](#)

## Select<TSource,TResult> Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) : [Select<TSource,TResult> Method](#)

The type of the elements of *source*.

The type of the value returned by *selector*.

An [IIndexedSource<TSource>](#) collection of values to invoke a transform function on.

A transform function to apply to each element.

Projects each element of a collection into a new form.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Function Select     (Of TSource,TResult)( _     ByVal source As IIndexedSource(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,TResult)) _ ) As IIndexedSource(Of TResult)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static IIndexedSource&lt;TResult&gt; Select&lt;TSource,TResult&gt;(     IIndexedSource&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,TResult&gt;&gt; selector )</pre>	

### Parameters

*source*

An [IIndexedSource<TSource>](#) collection of values to invoke a transform function on.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

*TResult*

The type of the value returned by *selector*.

### Return Value

An [IndexedSource<TResult>](#) whose elements are the result of invoking the transform function on each element of *source*.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)

### SelectMany Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) : SelectMany Method

Projects each element of a collection to a sequence of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.

## Overload List

Overload	Description
<a href="#">SelectMany&lt;TSource,TCollection,TResult&gt;(IIndexedSource&lt;TSource&gt;,Expression&lt;Func&lt;TSource,IEnumerable&lt;TCollection&gt;&gt;&gt;,Expression&lt;Func&lt;TSource,TCollection,TResult&gt;&gt;)</a>	Projects each element of a collection to a sequence of collections,

	flattens the resulting collections into one collection, and invokes a result selector function on each element therein.
<code>SelectMany&lt;TSource,TResult&gt;(IEnumerable&lt;TSource&gt;,Expression&lt;Func&lt;TSource,IEnumerable&lt;TResult&gt;&gt;&gt;)</code>	Projects each element of a collection to a sequence



	of collec tions and flatte ns the result ing collec tions into one collec tion.
--	---

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)

[IndexedQueryExtensions Members](#)

SelectMany<TSource,TCollection,TResult>(IIndexedSource<TSource>,Expression<Func<TSource, IEnumerable<TCollection>>>,Expression<Func<TSource,TCollection,TResult>>) Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [SelectMany Method](#) :

SelectMany<TSource,TCollection,TResult>(IIndexedSource<TSource>,Expression<Func<TSource,IEnumerable<TCollection>>>,Expression<Func<TSource,TCollection,TResult>>) Method

The type of the elements of *source*.

The type of the intermediate elements collected by *collectionSelector*.

The type of the elements of the resulting collection.

A collection of values to project.

A transform function to apply to each element of the input collection.

A transform function to apply to each element of the intermediate sequence.

Projects each element of a collection to a sequence of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function SelectMany     (Of TSource,TCollection,TResult)( _     ByVal source As IIndexedSource(Of TSource), _     ByVal collectionSelector As System.Linq.Expressions.Expression(Of Func(Of TSource,IEnumerable(Of TCollection))), _     ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of TSource,TCollection,TResult)) _ ) As IIndexedSource(Of TResult)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static IIndexedSource&lt;TResult&gt; SelectMany&lt;TSource,TCollection,TResult&gt;(     IIndexedSource&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,IEnumerable&lt;TCollection&gt;&gt;&gt; collectionSelector,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,TCollection,TResult&gt;&gt; resultSelector )</pre>	

### Parameters

*source*

A collection of values to project.

*collectionSelector*

A transform function to apply to each element of the input collection.

*resultSelector*

A transform function to apply to each element of the intermediate sequence.

### Type Parameters

*TSource*

The type of the elements of *source*.

*TCollection*

The type of the intermediate elements collected by *collectionSelector*.

*TResult*

The type of the elements of the resulting collection.

## Return Value

An [IndexedSource<TResult>](#) whose elements are the result of invoking the one-to-many transform function *collectionSelector* on each element of *source* and then mapping each of those sequence elements and their corresponding source element to a result element.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

[SelectMany<TSource,TResult>\(IEnumerable<TSource>,Expression<Func<TSource,IEnumerable<TResult>>>\) Method](#)

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [SelectMany Method](#) :

[SelectMany<TSource,TResult>\(IEnumerable<TSource>,Expression<Func<TSource,IEnumerable<TResult>>>\) Method](#)

The type of the elements of *source*.

The type of the elements of the sequence returned by *selector*.

A collection of values to project.

A transform function to apply to each element.

Projects each element of a collection to a sequence of collections and flattens the resulting collections into one collection.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function SelectMany
    (Of TSource,TResult)( _
    ByVal source As IIndexedSource(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,IEnumerable(Of TResult)))) _
) As IIndexedSource(Of TResult)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]
public static IIndexedSource<TResult> SelectMany<TSource,TResult>(
    IIndexedSource<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,IEnumerable<TResult>>>
selector
)
```

### Parameters

*source*

A collection of values to project.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

*TResult*

The type of the elements of the sequence returned by *selector*.

### Return Value

An [IIndexedSource<TResult>](#) whose elements are the result of invoking the one-to-many transform function on each element of the source collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

### ToIndexed Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) : ToIndexed Method

Creates an [C1.LiveLinq.Indexing.IIndexedSource<T>](#) based on the specified [IObservableSource<T>](#) collection.

## Overload List

Overload	Description
<a href="#">ToIndexed&lt;T&gt;(IObservableSource&lt;T&gt;)</a>	Creates an <a href="#">C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</a> based on the specified <a href="#">IObservableSource&lt;T&gt;</a> collection.
<a href="#">ToIndexed&lt;T&gt;(IBindingList)</a>	Creates an <a href="#">C1.LiveLinq.Collections.IndexedCollection&lt;T&gt;</a> based on the specified <b>System.ComponentModel.IBindingList</b> data source.
<a href="#">ToIndexed&lt;T&gt;(BindingList&lt;T&gt;)</a>	A typed specialization of the <a href="#">ToIndexed&lt;T&gt;(IBindingList)</a> method.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ToIndexed<T>\(IObservableSource<T>\) Method](#)  
[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [ToIndexed Method](#) :  
[ToIndexed<T>\(IObservableSource<T>\) Method](#)

The type of the elements in the collection.

An [IObservableSource<T>](#) collection to base an [C1.LiveLinq.Indexing.IIndexedSource<T>](#) on.

Creates an [C1.LiveLinq.Indexing.IIndexedSource<T>](#) based on the specified [IObservableSource<T>](#) collection.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function ToIndexed(Of T)( _     ByVal source As IObservableSource(Of T) _ ) As IIndexedSource(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static IIndexedSource&lt;T&gt; ToIndexed&lt;T&gt;(     IObservableSource&lt;T&gt; source )</pre>	

### Parameters

*source*

An [IObservableSource<T>](#) collection to base an [C1.LiveLinq.Indexing.IIndexedSource<T>](#) on.

### Type Parameters

*T*

The type of the elements in the collection.

### Return Value

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) that contains the same elements as the [IObservableSource<T>](#) collection and enables indexing of that collection.

## Remarks

Use this method to index and query your collection if that collection is your own custom implementation of the [IObservableSource<T>](#) interface.

Elements of the source collection aren't duplicated or copied to a new collection. This method just wraps the original collection in an [C1.LiveLinq.Indexing.IndexedSource<T>](#), enabling its indexing by using the change notification mechanism of [IObservableSource<T>](#).

**Note:** Indexes created on the resulting [C1.LiveLinq.Indexing.IndexedSource<T>](#) are owned by it and not by the original collection. Every **ToIndexed()** call creates a separate object that has its own separate indexes. Avoid calling **ToIndexed()** repeatedly for the same collection because it can increase the cost of maintaining indexes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

ToIndexed<T>(IBindingList) Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [ToIndexed Method](#) :

ToIndexed<T>(IBindingList) Method

The type of the elements in the collection.

An **System.ComponentModel.IBindingList** data source to represent as an [C1.LiveLinq.Collections.IndexedCollection<T>](#).

Creates an [C1.LiveLinq.Collections.IndexedCollection<T>](#) based on the specified **System.ComponentModel.IBindingList** data source.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function ToIndexed(Of T)( _  
    ByVal source As System.ComponentModel.IBindingList _
```

```
) As IndexedCollection(Of T)
```

```
C#
```

```
[System.Runtime.CompilerServices.Extension()]  
public static IndexedCollection<T> ToIndexed<T>(  
    System.ComponentModel.IBindingList source  
)
```

## Parameters

*source*

An **System.ComponentModel.IBindingList** data source to represent as an [C1.LiveLinq.Collections.IndexedCollection<T>](#).

## Type Parameters

*T*

The type of the elements in the collection.

## Return Value

An [C1.LiveLinq.Collections.IndexedCollection<T>](#) that contains the same elements as the **System.ComponentModel.IBindingList** and enables indexing of that data source.

## Remarks

Use this method to index and query your existing data sources. The only requirements for the data source is that it implements the standard data binding interface **System.ComponentModel.IBindingList**.

**Note:** Indexes created on the resulting [C1.LiveLinq.Collections.IndexedCollection<T>](#) are owned by it and not by the original data source. Every **ToIndexed()** call creates a separate object that has its own separate indexes. Avoid calling **ToIndexed()** repeatedly for the same collection because it can increase the cost of maintaining indexes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference



[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

## ToIndexed<T>(BindingList<T>) Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [ToIndexed Method](#) :

ToIndexed<T>(BindingList<T>) Method

The type of the elements in the collection.

A **System.ComponentModel.BindingList`1** data source to represent as an [C1.LiveLinq.Collections.IndexedCollection<T>](#).

A typed specialization of the [ToIndexed<T>\(IBindingList\)](#) method.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function ToIndexed(Of T)( _     ByVal source As System.ComponentModel.BindingList(Of T) _ ) As IndexedCollection(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static IndexedCollection&lt;T&gt; ToIndexed&lt;T&gt;(     System.ComponentModel.BindingList&lt;T&gt; source )</pre>	

## Parameters

*source*

A **System.ComponentModel.BindingList`1** data source to represent as an [C1.LiveLinq.Collections.IndexedCollection<T>](#).

## Type Parameters

*T*

The type of the elements in the collection.

## Return Value

An [C1.LiveLinq.Indexing.IndexedSource<T>](#) that contains the same elements as the **System.ComponentModel.BindingList`1** and enables indexing of that data source.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

### Where Method

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) : Where Method

Filters the source collection based on a predicate.

## Overload List

Overload	Description
<a href="#">Where&lt;T&gt;(IIndexedSource&lt;T&gt;, Expression&lt;Func&lt;T, Boolean&gt;&gt;)</a>	Filters the source collection based on a predicate.
<a href="#">Where&lt;T&gt;(IIndexedSource&lt;T&gt;, Expression&lt;Func&lt;T, Boolean&gt;&gt;, Boolean)</a>	Filters the source collection based on a predicate, preserving the order of the source collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)

[IndexedQueryExtensions Members](#)

[Where<T>\(IIndexedSource<T>,Expression<Func<T,Boolean>>\) Method](#)

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [Where Method](#) :

[Where<T>\(IIndexedSource<T>,Expression<Func<T,Boolean>>\) Method](#)

The type of the elements of *source*.

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) to filter.

A function to test each element for a condition.

Filters the source collection based on a predicate.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function Where(Of T)( _     ByVal source As IIndexedSource(Of T), _     ByVal predicate As System.Linq.Expressions.Expression(Of Func(Of T,Boolean)) _ ) As IIndexedSource(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static IIndexedSource&lt;T&gt; Where&lt;T&gt;(     IIndexedSource&lt;T&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;T,bool&gt;&gt; predicate )</pre>	

### Parameters

*source*

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) to filter.

*predicate*

A function to test each element for a condition.

### Type Parameters

*T*

The type of the elements of *source*.

## Return Value

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) that contains elements from the input collection that satisfy the condition.

## Remarks

If an index is used to optimized performance of this operation, the resulting collection may not be in the same order as the source collection. If you need to preserve the order, use the other overload of the **Where** operator. It will still be optimized, albeit to a lesser degree.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

[Where<T>\(IIndexedSource<T>,Expression<Func<T,Boolean>>,Boolean\) Method](#)

[C1.LiveLinq Namespace](#) > [IndexedQueryExtensions Class](#) > [Where Method](#) :

[Where<T>\(IIndexedSource<T>,Expression<Func<T,Boolean>>,Boolean\) Method](#)

The type of the elements of *source*.

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) to filter.

A function to test each element for a condition.

Specifies whether the source order must be preserved in the result.

Filters the source collection based on a predicate, preserving the order of the source collection.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function Where(Of T)( _  
    ByVal source As IIndexedSource(Of T), _  
    ByVal predicate As System.Linq.Expressions.Expression(Of Func(Of
```

```
T, Boolean)), _
    ByVal preserveOriginalOrder As System.Boolean _
) As IIndexedSource(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static IIndexedSource<T> Where<T>(
    IIndexedSource<T> source,
    System.Linq.Expressions.Expression<Func<T, bool>> predicate,
    System.bool preserveOriginalOrder
)
```

## Parameters

*source*

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) to filter.

*predicate*

A function to test each element for a condition.

*preserveOriginalOrder*

Specifies whether the source order must be preserved in the result.

## Type Parameters

*T*

The type of the elements of *source*.

## Return Value

An [C1.LiveLinq.Indexing.IIndexedSource<T>](#) that contains elements from the input collection that satisfy the condition.

## Remarks

Preserving the order can lessen the effect of performance optimization using an index. Use this overload only if preserving the order in this operation is essential.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedQueryExtensions Class](#)  
[IndexedQueryExtensions Members](#)  
[Overload List](#)

## LiveViewExtensions

[C1.LiveLinq Namespace](#) : LiveViewExtensions Class

Provides a set of static (extension) methods used in queries to define live views.

## Object Model

LiveViewExtensions

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public MustInherit NotInheritable Class LiveViewExtensions</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static class LiveViewExtensions</pre>	

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViewExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Members](#)  
[C1.LiveLinq Namespace](#)

# Overview

[C1.LiveLinq Namespace](#) : LiveViewExtensions Class

Provides a set of static (extension) methods used in queries to define live views.

# Object Model

LiveViewExtensions

# Syntax

Visual Basic (Declaration)	
<code>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public MustInherit NotInheritable Class LiveViewExtensions</code>	
C#	
<code>[System.Runtime.CompilerServices.Extension()] public static class LiveViewExtensions</code>	

# Inheritance Hierarchy

System.Object  
    **C1.LiveLinq.LiveViewExtensions**

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[LiveViewExtensions Members](#)  
[C1.LiveLinq Namespace](#)










# Members

[Methods](#)

[C1.LiveLinq Namespace](#) : LiveViewExtensions Class

The following tables list the members exposed by [LiveViewExtensions](#).

# Public Methods

	Name	Description
≡  S	<a href="#">AsLive</a>	Overloaded. Creates a view based on the specified <a href="#">IObservableSource&lt;T&gt;</a> collection.
≡  S	<a href="#">AsNonUpdatable&lt;T&gt;</a>	Specifies a view as read-only.
≡  S	<a href="#">AsUpdatable&lt;T&gt;</a>	Specifies a view as updatable.
≡  S	<a href="#">LiveAggregate</a>	Overloaded. Applies an accumulator function over a view.
≡  S	<a href="#">LiveAverage</a>	Overloaded. Computes the average of a view of <b>System.Int32</b> values.
≡  S	<a href="#">LiveCount</a>	Overloaded. A view representing the number of elements in a view.
≡  S	<a href="#">LiveMax</a>	Overloaded. Computes the maximum value of a view of <b>System.Double</b> values.
≡  S	<a href="#">LiveMin</a>	Overloaded. Computes the minimum value of a view of nullable <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.
≡  S	<a href="#">LiveSum</a>	Overloaded. Computes the sum of a view of <b>System.Int32</b> values.

[Top](#)

## See Also

### Reference

[LiveViewExtensions Class](#)  
[C1.LiveLinq Namespace](#)










## Methods

[C1.LiveLinq Namespace](#) : [LiveViewExtensions Class](#)

For a list of all members of this type, see [LiveViewExtensions members](#).



## Public Methods

	Name	Description
≡  S	<a href="#">AsLive</a>	Overloaded. Creates a view based on the specified <a href="#">IObservableSource&lt;T&gt;</a> collection.
≡  S	<a href="#">AsNonUpdatable&lt;T&gt;</a>	Specifies a view as read-only.
≡  S	<a href="#">AsUpdatable&lt;T&gt;</a>	Specifies a view as updatable.
≡  S	<a href="#">LiveAggregate</a>	Overloaded. Applies an accumulator function over a view.
≡  S	<a href="#">LiveAverage</a>	Overloaded. Computes the average of a view of <b>System.Int32</b> values.
≡  S	<a href="#">LiveCount</a>	Overloaded. A view representing the number of elements in a view.
≡  S	<a href="#">LiveMax</a>	Overloaded. Computes the maximum value of a view of <b>System.Double</b> values.
≡  S	<a href="#">LiveMin</a>	Overloaded. Computes the minimum value of a view of nullable <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.
≡  S	<a href="#">LiveSum</a>	Overloaded. Computes the sum of a view of <b>System.Int32</b> values.

[Top](#)

## See Also

### Reference

[LiveViewExtensions Class](#)

[C1.LiveLinq Namespace](#)

### AsLive Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : AsLive Method

Creates a view based on the specified [IObservableSource<T>](#) collection.

## Overload List

Overload	Description
<a href="#">AsLive&lt;T&gt;(IObservableSource&lt;T&gt;)</a>	Creates a view based on the specified <a href="#">IObservableSource&lt;T&gt;</a> collection.
<a href="#">AsLive&lt;T&gt;(IObservableSource&lt;T&gt;,ViewOrder)</a>	Creates a view based on the specified <a href="#">IObservableSource&lt;T&gt;</a> collection.
<a href="#">AsLive&lt;T&gt;(IBindingList)</a>	Creates a view based on the specified <b>System.ComponentModel.IBindingList</b> data source.
<a href="#">AsLive&lt;T&gt;(IBindingList,ViewOrder)</a>	Creates a view based on the specified <b>System.ComponentModel.IBindingList</b> data source.
<a href="#">AsLive&lt;T&gt;(BindingList&lt;T&gt;)</a>	A typed specialization of the <a href="#">AsLive&lt;T&gt;(IBindingList)</a> method.
<a href="#">AsLive&lt;T&gt;(BindingList&lt;T&gt;,ViewOrder)</a>	A typed specialization of the <a href="#">AsLive&lt;T&gt;(IBindingList,ViewOrder)</a> method.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[AsLive<T>\(IObservableSource<T>\) Method](#)

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) : [AsLive<T>\(IObservableSource<T>\)](#) Method

The type of the elements in the collection.

The [IObservableSource<T>](#) collection to expose as a view.

Creates a view based on the specified [IObservableSource<T>](#) collection.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function AsLive(Of T)( _     ByVal source As IObservableSource(Of T) _ ) As View(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsLive&lt;T&gt;(     IObservableSource&lt;T&gt; source )</pre>	

### Parameters

*source*

The [IObservableSource<T>](#) collection to expose as a view.

### Type Parameters

*T*

The type of the elements in the collection.

### Return Value

A view that contains the same elements as the [IObservableSource<T>](#)

## Remarks

The resulting view may have its elements ordered differently than they are ordered in the *source* collection. Correspondingly, views built on this resulting view (for example, if you filter it with **Where**) will not preserve the source order either. If you need to preserve the source order, consider using the other **AsLive** overload where you can specify to what extent you need the order to be preserved.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

- [LiveViewExtensions Class](#)
- [LiveViewExtensions Members](#)
- [Overload List](#)

AsLive<T>(IObservableSource<T>,ViewOrder) Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) :  
AsLive<T>(IObservableSource<T>,ViewOrder) Method

- The type of the elements in the collection.
- The [IObservableSource<T>](#) collection to expose as a view.
- Specifies whether to preserve source item order.
- Creates a view based on the specified [IObservableSource<T>](#) collection.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function AsLive(Of T)( _     ByVal source As IObservableSource(Of T), _     ByVal order As ViewOrder _ ) As View(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsLive&lt;T&gt;(     IObservableSource&lt;T&gt; source,     ViewOrder order )</pre>	

### Parameters

- source*
- The [IObservableSource<T>](#) collection to expose as a view.
- order*

Specifies whether to preserve source item order.

## Type Parameters

*T*

The type of the elements in the collection.

## Return Value

A view that contains the same elements as the [IObservableSource<T>](#)

## Remarks

If the *order* parameter specifies preserving item order, the order of items in the source is preserved, at a certain performance cost, in the resulting view and in views based on it (for example, if you filter it with **Where**).

Note that **Join** does not preserve source order. If you need to order a join result, use **OrderBy** after **Join**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

AsLive<T>(IBindingList) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) : AsLive<T>(IBindingList) Method

The type of the elements in the view.

The **System.ComponentModel.IBindingList** data source to expose as a view.

Creates a view based on the specified **System.ComponentModel.IBindingList** data source.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
```

```
Public Overloads Shared Function AsLive(Of T)( _
    ByVal source As System.ComponentModel.IBindingList _
) As View(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<T> AsLive<T>(
    System.ComponentModel.IBindingList source
)
```

## Parameters

*source*

The **System.ComponentModel.IBindingList** data source to expose as a view.

## Type Parameters

*T*

The type of the elements in the view.

## Return Value

A view that contains the same elements as the **System.ComponentModel.IBindingList**.

## Remarks

Use this method to build views from existing data sources. The only requirements for the data source is that it implements the standard data binding interface **System.ComponentModel.IBindingList**.

The resulting view may have its elements ordered differently than they are ordered in the *source* collection. Correspondingly, views built on this resulting view (for example, if you filter it with **Where**) will not preserve the source order either. If you need to preserve the source order, consider using the other **AsLive** overload where you can specify to what extent you need the order to be preserved.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## AsLive<T>(IBindingList,ViewOrder) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) : AsLive<T>(IBindingList,ViewOrder)  
 Method

The type of the elements in the view.

The **System.ComponentModel.IBindingList** data source to expose as a view.

Specifies whether to preserve source item order.

Creates a view based on the specified **System.ComponentModel.IBindingList** data source.

## Syntax

Visual Basic (Declaration)	
<pre> &lt;System.Runtime.CompilerServices.ExtensionAttribute()&gt; Public Overloads Shared Function AsLive(Of T)( _     ByVal source As System.ComponentModel.IBindingList, _     ByVal order As ViewOrder _ ) As View(Of T) </pre>	
C#	
<pre> [System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsLive&lt;T&gt;(     System.ComponentModel.IBindingList source,     ViewOrder order ) </pre>	

## Parameters

*source*

The **System.ComponentModel.IBindingList** data source to expose as a view.

*order*

Specifies whether to preserve source item order.

## Type Parameters

*T*

The type of the elements in the view.

## Return Value

A view that contains the same elements as the **System.ComponentModel.IBindingList**.

## Remarks

Use this method to build views from existing data sources. The only requirements for the data source is that it implements the standard data binding interface **System.ComponentModel.IBindingList**.

If the *order* parameter specifies preserving item order, the order of items in the source is preserved, at a certain performance cost, in the resulting view and in views based on it (for example, if you filter it with **Where**).

Note that **Join** does not preserve source order. If you need to order a join result, use **OrderBy** after **Join**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

**AsLive<T>(BindingList<T>) Method**

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) : **AsLive<T>(BindingList<T>) Method**

The type of the elements in the view.

The **System.ComponentModel.BindingList<T>** data source to expose as a view.

A typed specialization of the [AsLive<T>\(IBindingList\)](#) method.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function AsLive(Of T)( _
```



```

    ByVal source As System.ComponentModel.BindingList(Of T) _
) As View(Of T)

```

C#

```

[System.Runtime.CompilerServices.Extension()]
public static View<T> AsLive<T>(
    System.ComponentModel.BindingList<T> source
)

```

## Parameters

*source*

The **System.ComponentModel.BindingList`1** data source to expose as a view.

## Type Parameters

*T*

The type of the elements in the view.

## Return Value

A view that contains the same elements as the **System.ComponentModel.BindingList`1**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

AsLive<T>(BindingList<T>,ViewOrder) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) :

AsLive<T>(BindingList<T>,ViewOrder) Method

The type of the elements in the view.

The **System.ComponentModel.BindingList`1** data source to expose as a view.

Specifies whether to preserve source item order.

A typed specialization of the [AsLive<T>\(IBindingList,ViewOrder\)](#) method.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function AsLive(Of T)( _     ByVal source As System.ComponentModel.BindingList(Of T), _     ByVal order As ViewOrder _ ) As View(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsLive&lt;T&gt;(     System.ComponentModel.BindingList&lt;T&gt; source,     ViewOrder order )</pre>	

### Parameters

*source*

The **System.ComponentModel.BindingList`1** data source to expose as a view.

*order*

Specifies whether to preserve source item order.

### Type Parameters

*T*

The type of the elements in the view.

### Return Value

A view that contains the same elements as the **System.ComponentModel.BindingList`1**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

Reference

- [LiveViewExtensions Class](#)
- [LiveViewExtensions Members](#)
- [Overload List](#)

AsNonUpdatable<T> Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : AsNonUpdatable<T> Method

The type of the elements in the view.

The view to specify as read-only.

Specifies a view as read-only.

Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Function AsNonUpdatable(Of T)( _     ByVal view As View(Of T) _ ) As View(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsNonUpdatable&lt;T&gt;(     View&lt;T&gt; view )</pre>	

Parameters

*view*

The view to specify as read-only.

Type Parameters

*T*

The type of the elements in the view.

Return Value

The same *view*.

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

- [LiveViewExtensions Class](#)
- [LiveViewExtensions Members](#)
- [AsUpdatable<T> Method](#)

### AsUpdatable<T> Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : AsUpdatable<T> Method

The type of the elements in the view.

The view to specify as updatable.

Specifies a view as updatable.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Function AsUpdatable(Of T)( _     ByVal view As View(Of T) _ ) As View(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsUpdatable&lt;T&gt;(     View&lt;T&gt; view )</pre>	

### Parameters

*view*

The view to specify as updatable.

### Type Parameters

*T*

The type of the elements in the view.

## Return Value

The same *view*.

## Remarks

This method is used with **Join** operation to specify which of the two parts of the join you want to be updatable.

Properties exposed by a view can be updatable or read-only. Updatable properties of a view can be modified directly in the view. Read-only properties of a view cannot be modified directly in the view, but they still reflect up-to-date values of the source, so the difference is often not critical, you can always modify corresponding property in the source, that will automatically change the property in the view.

Only one of the two arguments of a **Join** can be updatable, the one you qualified with **AsUpdatable()**. In absence of this qualification, both parts of the join are read-only. For example, in `from c in customers.AsLive() join o in orders.AsLive().AsUpdatable() on o.CustomerID equals c.CustomerID select new { c.CustomerName, o.OrderDate, o.OrderAmount }` **CustomerName** is read-only, and **OrderDate** and **OrderAmount** are updatable.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[IsReadOnly Property](#)  
[ViewRow Class](#)  
[ViewRowState Enumeration](#)

### LiveAggregate Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : LiveAggregate Method

Applies an accumulator function over a view.

## Overload List

Overload	Des
----------	-----

	crip tio n
LiveAggregate<TSource>(View<TSource>,Expression<Func<TSource,TSource,TSource>>, Expression<Func<TSource,TSource,TSource>>,Expression<Func<TSource,TSource,Boolean >>)	Ap plie s an acc um ulat or fun ctio n ove r a vie w.
LiveAggregate<TSource,TAccumulate>(View<TSource>,TAccumulate,Expression<Func<TA ccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,TAccumulate> >,Expression<Func<TAccumulate,TSource,Boolean>>)	Ap plie s an acc um ulat or fun ctio n ove r a vie w. The spe

	<p>cified          ed          see          d          value          is          used          d          as          the          initial          accumulator          or          value.</p>
<p><code>LiveAggregate&lt;TSource,TAccumulate,TResult&gt;(View&lt;TSource&gt;,TAccumulate,Expression&lt;Func&lt;TAccumulate,TSource,TAccumulate&gt;&gt;,Expression&lt;Func&lt;TAccumulate,TSource,TAccumulate&gt;&gt;,Expression&lt;Func&lt;TAccumulate,TSource,Boolean&gt;&gt;,Expression&lt;Func&lt;TAccumulate,TResult&gt;&gt;)</code></p>	<p>Applies an accumulator or function over a view. The</p>

	specified value is used as the initial accumulator value, and the specified function is used to select the result value
--	--



	ue.
--	-----

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)

LiveAggregate<TSource>(View<TSource>,Expression<Func<TSource,TSource,TSource>>,Expression<Func<TSource,TSource,TSource>>,Expression<Func<TSource,TSource,Boolean>>)  
Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAggregate Method](#) :  
LiveAggregate<TSource>(View<TSource>,Expression<Func<TSource,TSource,TSource>>,Expression<Func<TSource,TSource,TSource>>,Expression<Func<TSource,TSource,Boolean>>)  
Method

The type of the elements of *source*.

A view to aggregate over.

An accumulator function to be invoked on each element that is added to the source view.

A function to be applied to the accumulated value and to an element to obtain the changed accumulated value, when an element is removed from the source view.

A function used to determine whether *funcRemove* must be applied when an element is removed from the source view, or the accumulated value is not affected by its removal.

Applies an accumulator function over a view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAggregate(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal funcAdd As System.Linq.Expressions.Expression(Of Func(Of TSource,TSource,TSource))), _</pre>	

```

    ByVal funcRemove As System.Linq.Expressions.Expression(Of Func(Of
TSource,TSource,TSource)), _
    ByVal funcRemoveDefined As System.Linq.Expressions.Expression(Of Func(Of
TSource,TSource,Boolean)) _
) As AggregationView(Of)

```

C#

```

[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,TSource> LiveAggregate<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,TSource,TSource>> funcAdd,
    System.Linq.Expressions.Expression<Func<TSource,TSource,TSource>>
funcRemove,
    System.Linq.Expressions.Expression<Func<TSource,TSource,bool>>
funcRemoveDefined
)

```

## Parameters

*source*

A view to aggregate over.

*funcAdd*

An accumulator function to be invoked on each element that is added to the source view.

*funcRemove*

A function to be applied to the accumulated value and to an element to obtain the changed accumulated value, when an element is removed from the source view.

*funcRemoveDefined*

A function used to determine whether *funcRemove* must be applied when an element is removed from the source view, or the accumulated value is not affected by its removal.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the final accumulator value.

## Remarks

It is possible to use standard LINQ query operator **Aggregate** instead of **LiveAggregate**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Aggregate** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAggregate** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

`LiveAggregate<TSource,TAccumulate>(View<TSource>,TAccumulate,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,Boolean>>)` Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAggregate Method](#) :

`LiveAggregate<TSource,TAccumulate>(View<TSource>,TAccumulate,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,Boolean>>)` Method

The type of the elements of *source*.

The type of the accumulator value.

A view to aggregate over.

The initial accumulator value.

An accumulator function to be invoked on each element that is added to the source view.

A function to be applied to the accumulated value and to an element to obtain the changed accumulated value, when an element is removed from the source view.

A function used to determine whether *funcRemove* must be applied when an element is removed from the source view, or the accumulated value is not affected by its removal.

Applies an accumulator function over a view. The specified seed value is used as the initial accumulator value.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAggregate
    (Of TSource,TAccumulate)( _
    ByVal source As View(Of TSource), _
    ByVal seed As TAccumulate, _
    ByVal funcAdd As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate,TSource,TAccumulate)), _
    ByVal funcRemove As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate,TSource,TAccumulate)), _
    ByVal funcRemoveDefined As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate,TSource,Boolean))) _
) As AggregationView(Of TSource,TAccumulate)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,TAccumulate>
LiveAggregate<TSource,TAccumulate>(
    View<TSource> source,
    TAccumulate seed,
    System.Linq.Expressions.Expression<Func<TAccumulate,TSource,TAccumulate>>
funcAdd,
    System.Linq.Expressions.Expression<Func<TAccumulate,TSource,TAccumulate>>
funcRemove,
    System.Linq.Expressions.Expression<Func<TAccumulate,TSource,bool>>
funcRemoveDefined
)
```

### Parameters

*source*

A view to aggregate over.

*seed*

The initial accumulator value.

*funcAdd*

An accumulator function to be invoked on each element that is added to the source view.

*funcRemove*

A function to be applied to the accumulated value and to an element to obtain the changed accumulated value, when an element is removed from the source view.

*funcRemoveDefined*

A function used to determine whether *funcRemove* must be applied when an element is removed from the source view, or the accumulated value is not affected by its removal.

## Type Parameters

*TSource*

The type of the elements of *source*.

*TAccumulate*

The type of the accumulator value.

## Return Value

A view representing the final accumulator value.

## Remarks

It is possible to use standard LINQ query operator **Aggregate** instead of **LiveAggregate**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Aggregate** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAggregate** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

LiveAggregate<TSource,TAccumulate,TResult>(View<TSource>,TAccumulate,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,Boolean>>,Expression<Func<TAccumulate,TResult>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAggregate Method](#) :

LiveAggregate<TSource,TAccumulate,TResult>(View<TSource>,TAccumulate,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,Boolean>>,Expression<Func<TAccumulate,TResult>>) Method

The type of the elements of *source*.

The type of the accumulator value.

The type of the resulting value.

A view to aggregate over.

The initial accumulator value.

An accumulator function to be invoked on each element that is added to the source view.

A function to be applied to the accumulated value and to an element to obtain the changed accumulated value, when an element is removed from the source view.

A function used to determine whether *funcRemove* must be applied when an element is removed from the source view, or the accumulated value is not affected by its removal.

A function to transform the final accumulator value into the result value.

Applies an accumulator function over a view. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.

## Syntax

Visual Basic (Declaration)

```

<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAggregate
    (Of TSource,TAccumulate,TResult)( _
    ByVal source As View(Of TSource), _
    ByVal seed As TAccumulate, _
    ByVal funcAdd As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate,TSource,TAccumulate)), _
    ByVal funcRemove As System.Linq.Expressions.Expression(Of Func(Of

```

```

TAccumulate, TSource, TAccumulate)), _
    ByVal funcRemoveDefined As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate, TSource, Boolean))), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate, TResult)) _
) As AggregationView(Of TSource, TResult)

```

C#

```

[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource, TResult>
LiveAggregate<TSource, TAccumulate, TResult>(
    View<TSource> source,
    TAccumulate seed,
    System.Linq.Expressions.Expression<Func<TAccumulate, TSource, TAccumulate>>
funcAdd,
    System.Linq.Expressions.Expression<Func<TAccumulate, TSource, TAccumulate>>
funcRemove,
    System.Linq.Expressions.Expression<Func<TAccumulate, TSource, bool>>
funcRemoveDefined,
    System.Linq.Expressions.Expression<Func<TAccumulate, TResult>>
resultSelector
)

```

## Parameters

*source*

A view to aggregate over.

*seed*

The initial accumulator value.

*funcAdd*

An accumulator function to be invoked on each element that is added to the source view.

*funcRemove*

A function to be applied to the accumulated value and to an element to obtain the changed accumulated value, when an element is removed from the source view.

*funcRemoveDefined*

A function used to determine whether *funcRemove* must be applied when an element is removed from the source view, or the accumulated value is not affected by its removal.

*resultSelector*

A function to transform the final accumulator value into the result value.

## Type Parameters

*TSource*

The type of the elements of *source*.

*TAccumulate*

The type of the accumulator value.

*TResult*

The type of the resulting value.

## Return Value

A view representing the final accumulator value.

## Remarks

It is possible to use standard LINQ query operator **Aggregate** instead of **LiveAggregate**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Aggregate** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAggregate** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference



[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveAverage Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : LiveAverage Method

Computes the average of a view of **System.Int32** values.

## Overload List

Overload	Description
<a href="#">LiveAverage(View&lt;Int32&gt;)</a>	Computes the average of a view of <b>System.Int32</b> values.
<a href="#">LiveAverage(View&lt;Nullable&lt;Int32&gt;&gt;)</a>	Computes the average of a view of nullable <b>System.Int32</b> values.
<a href="#">LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int32&gt;&gt;)</a>	Computes the average of a view of <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.
<a href="#">LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int32&gt;&gt;)</a>	Computes the average of a view of nullable <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.

<code>32&gt;&gt;&gt;)</code>	average of a view of nullable <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveAverage(View&lt;Int64&gt;)</code>	Computes the average of a view of <b>System.Int64</b> values.
<code>LiveAverage(View&lt;Nullable&lt;Int64&gt;&gt;)</code>	Computes the average of a view of nullable <b>System.Int64</b> values.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int64&gt;&gt;)</code>	Computes the average of a view of <b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source

	view.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int64&gt;&gt;&gt;)</code>	Computes the average of a view of nullable <b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveAverage(View&lt;Decimal&gt;)</code>	Computes the average of a view of <b>System.Decimal</b> values.
<code>LiveAverage(View&lt;Nullable&lt;Decimal&gt;&gt;)</code>	Computes the average of a view of nullable <b>System.Decimal</b> values.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Decimal&gt;&gt;)</code>	Computes the average of a view of <b>System.Decimal</b> values that are obtained by invoking a

	transform function on each element of the source view.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Decimal&gt;&gt;&gt;)</code>	Computes the average of a view of nullable <b>System.Decimal</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveAverage(View&lt;Double&gt;)</code>	Computes the average of a view of <b>System.Double</b> values.
<code>LiveAverage(View&lt;Nullable&lt;Double&gt;&gt;)</code>	Computes the average of a view of nullable <b>System.Double</b> values.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Double&gt;&gt;)</code>	Computes the average of a view of

	<p><b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.</p>
<p><code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Double&gt;&gt;&gt;)</code></p>	<p>Computes the average of a view of nullable <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.</p>
<p><code>LiveAverage(View&lt;Single&gt;)</code></p>	<p>Computes the average of a view of <b>System.Single</b> values.</p>
<p><code>LiveAverage(View&lt;Nullable&lt;Single&gt;&gt;)</code></p>	<p>Computes the average of a view of nullable <b>System.Single</b> values.</p>

<a href="#">LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Single&gt;&gt;)</a>	Computes the average of a view of <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.
<a href="#">LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Single&gt;&gt;&gt;)</a>	Computes the average of a view of nullable <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

## LiveAverage(View<Int32>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) : LiveAverage(View<Int32>) Method

A view containing the values to calculate the average of.

Computes the average of a view of **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Integer) _ ) As AggregationView(Of Integer,Double)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;int,double&gt; LiveAverage(     View&lt;int&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the average of.

### Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage(View<Nullable<Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage(View<Nullable<Int32>>) Method

A view containing the values to calculate the average of.

Computes the average of a view of nullable **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Nullable(Of Integer)) _ ) As AggregationView(Of Nullable(Of Integer),Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;int&gt;,Nullable&lt;double&gt;&gt; LiveAverage(     View&lt;Nullable&lt;int&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the average of.

### Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, the average value is null.



It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAverage(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Integer))) _
) As AggregationView(Of TSource,Double)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,double> LiveAverage<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,int>> selector  
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

**LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method**

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

**LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method**

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of nullable **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre> &lt;System.Runtime.CompilerServices.ExtensionAttribute()&gt; Public Overloads Shared Function LiveAverage(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Integer)))) _ ) As AggregationView(Of TSource,Nullable(Of Double)) </pre>	
C#	
<pre> [System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;double&gt;&gt; LiveAverage&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;int&gt;&gt;&gt; selector ) </pre>	

### Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage(View<Int64>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) : LiveAverage(View<Int64>) Method

A view containing the values to calculate the average of.

Computes the average of a view of **System.Int64** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Long) _</pre>	

```
) As AggregationView(Of Long,Double)
```

```
C#
```

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<long,double> LiveAverage(  
    View<long> source  
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage(View<Nullable<Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage(View<Nullable<Int64>>) Method

A view containing the values to calculate the average of.

Computes the average of a view of nullable **System.Int64** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveAverage( _  
    ByVal source As View(Of Nullable(Of Long)) _  
) As AggregationView(Of Nullable(Of Long), Nullable(Of Double))
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<Nullable<long>, Nullable<double>> LiveAverage(  
    View<Nullable<long>> source  
)
```

### Parameters

*source*

A view containing the values to calculate the average of.

### Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

`LiveAverage<TSource>(View<TSource>, Expression<Func<TSource, Int64>>)` Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

`LiveAverage<TSource>(View<TSource>, Expression<Func<TSource, Int64>>)` Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAverage(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource, Long)) _
) As AggregationView(Of TSource, Double)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource, double> LiveAverage<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource, long>> selector
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

`LiveAverage<TSource>(View<TSource>, Expression<Func<TSource, Nullable<Int64>>>) Method`

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

`LiveAverage<TSource>(View<TSource>, Expression<Func<TSource, Nullable<Int64>>>) Method`

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of nullable **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax



Visual Basic (Declaration)	
<pre> &lt;System.Runtime.CompilerServices.ExtensionAttribute()&gt; Public Overloads Shared Function LiveAverage(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Long)))) _ ) As AggregationView(Of TSource,Nullable(Of Double)) </pre>	
C#	
<pre> [System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;double&gt;&gt; LiveAverage&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;long&gt;&gt;&gt; selector ) </pre>	

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage(View<Decimal>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) : LiveAverage(View<Decimal>) Method

A view containing the values to calculate the average of.

Computes the average of a view of **System.Decimal** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Decimal) _ ) As AggregationView(Of Decimal,Decimal)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;decimal,decimal&gt; LiveAverage(     View&lt;decimal&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the average of.

### Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage(View<Nullable<Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage(View<Nullable<Decimal>>) Method

A view containing the values to calculate the average of.

Computes the average of a view of nullable **System.Decimal** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveAverage( _  
    ByVal source As View(Of Nullable(Of Decimal)) _  
) As AggregationView(Of Nullable(Of Decimal), Nullable(Of Decimal))
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<Nullable<decimal>, Nullable<decimal>>
```

```
LiveAverage(  
    View<Nullable<decimal>> source  
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAverage(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Decimal))) _
) As AggregationView(Of TSource,Decimal)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,decimal> LiveAverage<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,decimal>> selector
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that

**Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>)  
Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of nullable **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Decimal)))) _ ) As AggregationView(Of TSource,Nullable(Of Decimal))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()]</pre>	

```
public static AggregationView<TSource,Nullable<decimal>>
LiveAverage<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,Nullable<decimal>>>
selector
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveAverage(View<Double>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) : LiveAverage(View<Double>) Method

A view containing the values to calculate the average of.

Computes the average of a view of **System.Double** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Double) _ ) As AggregationView(Of Double,Double)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;double,double&gt; LiveAverage(     View&lt;double&gt; source )</pre>	

## Parameters

*source*

A view containing the values to calculate the average of.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.



## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage(View<Nullable<Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage(View<Nullable<Double>>) Method

A view containing the values to calculate the average of.

Computes the average of a view of nullable **System.Double** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Nullable(Of Double))) _ ) As AggregationView(Of Nullable(Of Double),Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;double&gt;,Nullable&lt;double&gt;&gt; LiveAverage(     View&lt;Nullable&lt;double&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the average of.

### Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAverage(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Double))) _
```

```
) As AggregationView(Of TSource,Double)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,double> LiveAverage<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,double>> selector  
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

`LiveAverage<TSource>(View<TSource>, Expression<Func<TSource, Nullable<Double>>>)`  
Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

`LiveAverage<TSource>(View<TSource>, Expression<Func<TSource, Nullable<Double> > >)` Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of nullable **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource, Nullable(Of Double))) _ ) As AggregationView(Of TSource, Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource, Nullable&lt;double&gt;&gt; LiveAverage&lt;TSource&gt;(      View&lt;TSource&gt; source,      System.Linq.Expressions.Expression&lt;Func&lt;TSource, Nullable&lt;double&gt;&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage(View<Single>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) : LiveAverage(View<Single>) Method

A view containing the values to calculate the average of.

Computes the average of a view of **System.Single** values.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAverage( _
    ByVal source As View(Of Single) _
) As AggregationView(Of Single,Double)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<float,double> LiveAverage(
    View<float> source
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage(View<Nullable<Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage(View<Nullable<Single>>) Method

A view containing the values to calculate the average of.

Computes the average of a view of nullable **System.Single** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Nullable(Of Single)) _ ) As AggregationView(Of Nullable(Of Single),Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;float&gt;,Nullable&lt;double&gt;&gt; LiveAverage(     View&lt;Nullable&lt;float&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the average of.

### Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Single))) _ ) As AggregationView(Of TSource,Double)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,double&gt; LiveAverage&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,float&gt;&gt; selector )</pre>	

### Parameters

*source*



A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of nullable **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Single)))) _ ) As AggregationView(Of TSource,Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;double&gt;&gt; LiveAverage&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;float&gt;&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

### Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

### LiveCount Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : LiveCount Method

A view representing the number of elements in a view.

## Overload List

Overload	Description
<a href="#">LiveCount&lt;T&gt;(View&lt;T&gt;)</a>	A view representing the number of elements in a view.
<a href="#">LiveCount&lt;T&gt;(View&lt;T&gt;,Expression&lt;Func&lt;T,Boolean&gt;&gt;)</a>	A view representing the number of elements of the specified view satisfying a condition.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)

LiveCount<T>(View<T>) Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveCount Method](#) : LiveCount<T>(View<T>)  
Method

- The type of the elements of *source*.
- A view that contains elements to be counted.
- A view representing the number of elements in a view.

Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveCount(Of T)( _     ByVal source As View(Of T) _ ) As AggregationView(Of T,Integer)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;T,int&gt; LiveCount&lt;T&gt;(     View&lt;T&gt; source )</pre>	

Parameters

- source*
- A view that contains elements to be counted.

Type Parameters

- T*
- The type of the elements of *source*.

Remarks

It is possible to use standard LINQ query operator **Count** instead of **LiveCount**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Count** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveCount** will use a

more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveCount<T>(View<T>,Expression<Func<T,Boolean>>) Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveCount Method](#) :  
LiveCount<T>(View<T>,Expression<Func<T,Boolean>>) Method

- The type of the elements of *source*.
- A view that contains elements to be tested and counted.
- A function to test each element for a condition.
- A view representing the number of elements of the specified view satisfying a condition.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveCount(Of T)( _     ByVal source As View(Of T), _     ByVal predicate As System.Linq.Expressions.Expression(Of Func(Of T,Boolean)) _ ) As AggregationView(Of T,Integer)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;T,int&gt; LiveCount&lt;T&gt;(     View&lt;T&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;T,bool&gt;&gt; predicate )</pre>	

## Parameters

*source*

A view that contains elements to be tested and counted.

*predicate*

A function to test each element for a condition.

## Type Parameters

*T*

The type of the elements of *source*.

## Remarks

It is possible to use standard LINQ query operator **Count** instead of **LiveCount**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Count** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveCount** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveMax Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : LiveMax Method

Computes the maximum value of a view of **System.Double** values.

## Overload List

Overload	Description
----------	-------------

LiveMax(View<Double>)	Computes the maximum value of a view of <b>System.Double</b> values.
LiveMax(View<Nullable<Double>>)	Computes the maximum value of a view of nullable <b>System.Double</b> values.
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Double>>)	Computes the maximum value of a view of <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>)	Computes the maximum value of a view of nullable <b>System.Double</b> values that are obtained by invoking a transform

	function on each element of the source view.
<code>LiveMax(View&lt;Single&gt;)</code>	Computes the maximum value of a view of <b>System.Single</b> values.
<code>LiveMax(View&lt;Nullable&lt;Single&gt;&gt;)</code>	Computes the maximum value of a view of nullable <b>System.Single</b> values.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Single&gt;&gt;)</code>	Computes the maximum value of a view of <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Single&gt;&gt;&gt;)</code>	Computes the maximum value of a view of nullable



	<p><b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.</p>
<p><code>LiveMax&lt;TSource,TResult&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,TResult&gt;&gt;&gt;)</code></p>	<p>Invokes a transform function on each element of a view of elements of a generic type and computes the maximum resulting value.</p>
<p><code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;)</code></p>	<p>Computes the maximum value of a view of elements of a generic type.</p>
<p><code>LiveMax(View&lt;Int32&gt;)</code></p>	<p>Computes the maximum value of a view of <b>System.Int32</b> values.</p>
<p><code>LiveMax(View&lt;Nullable&lt;Int32&gt;&gt;)</code></p>	<p>Computes the</p>

	maximum value of a view of nullable <b>System.Int32</b> values.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int32&gt;&gt;)</code>	Computes the maximum value of a view of <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int32&gt;&gt;&gt;)</code>	Computes the maximum value of a view of nullable <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMax(View&lt;Decimal&gt;)</code>	Computes the maximum value of a view

	of <b>System.Decimal</b> values.
<code>LiveMax(View&lt;Nullable&lt;Decimal&gt;&gt;)</code>	Computes the maximum value of a view of nullable <b>System.Decimal</b> values.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Decimal&gt;&gt;)</code>	Computes the maximum value of a view of <b>System.Decimal</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Decimal&gt;&gt;&gt;)</code>	Computes the maximum value of a view of nullable <b>System.Decimal</b> values that are obtained by invoking a transform function on each element of the source

	view.
<code>LiveMax(View&lt;Int64&gt;)</code>	Computes the maximum value of a view of <b>System.Int64</b> values.
<code>LiveMax(View&lt;Nullable&lt;Int64&gt;&gt;)</code>	Computes the maximum value of a view of nullable <b>System.Int64</b> values.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int64&gt;&gt;)</code>	Computes the maximum value of a view of <b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int64&gt;&gt;&gt;)</code>	Computes the maximum value of a view of nullable <b>System.Int64</b> values that are obtained by

	invoking a transform function on each element of the source view.
--	---

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

LiveMax(View<Double>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Double>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of **System.Double** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax( _     ByVal source As View(Of Double) _ ) As AggregationView(Of Double,Double)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;double,double&gt; LiveMax(     View&lt;double&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax(View<Nullable<Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax(View<Nullable<Double>>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of nullable **System.Double** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax( _     ByVal source As View(Of Nullable(Of Double)) _</pre>	

```
) As AggregationView(Of Nullable(Of Double),Nullable(Of Double))
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<Nullable<double>,Nullable<double>> LiveMax(  
    View<Nullable<double>> source  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Double)) _ ) As AggregationView(Of TSource,Double)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,double&gt; LiveMax&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,double&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

### Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.



It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>) Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :  
LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>) Method

- The type of the elements of *source*.
- A view containing the values to determine the maximum value of.
- A transform function to apply to each element.
- Computes the maximum value of a view of nullable **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Double))) _ ) As AggregationView(Of TSource,Nullable(Of Double))</pre>	
C#	

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,Nullable<double>> LiveMax<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,Nullable<double>>>  
selector  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveMax(View<Single>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Single>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of **System.Single** values.

## Syntax

### Visual Basic (Declaration)

```

<System.Runtime.CompilerServices.ExtensionAttribute()>
Public Overloads Shared Function LiveMax( _
    ByVal source As View(Of Single) _
) As AggregationView(Of Single,Single)

```

### C#

```

[System.Runtime.CompilerServices.Extension()]
public static AggregationView<float,float> LiveMax(
    View<float> source
)

```

## Parameters

*source*

A view containing the values to determine the maximum value of.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax(View<Nullable<Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax(View<Nullable<Single>>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of nullable **System.Single** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;&gt; Public Overloads Shared Function LiveMax( _     ByVal source As View(Of Nullable(Of Single)) _ ) As AggregationView(Of Nullable(Of Single),Nullable(Of Single))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;float&gt;,Nullable&lt;float&gt;&gt; LiveMax(     View&lt;Nullable&lt;float&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

### Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMax(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Single))) _
```

```
) As AggregationView(Of TSource,Single)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,float> LiveMax<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,float>> selector  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

`LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>)` Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

`LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>)` Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of nullable **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMax(Of TSource)( _  
    ByVal source As View(Of TSource), _  
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of  
TSource,Nullable(Of Single))) _  
) As AggregationView(Of TSource,Nullable(Of Single))
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,Nullable<float>> LiveMax<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,Nullable<float>>> selector  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

`LiveMax<TSource,TResult>(View<TSource>,Expression<Func<TSource,TResult>>)` Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

`LiveMax<TSource,TResult>(View<TSource>,Expression<Func<TSource,TResult>>)` Method

The type of the elements of *source*.

The type of the value returned by *selector*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Invokes a transform function on each element of a view of elements of a generic type and computes the maximum resulting value.

## Syntax



Visual Basic (Declaration)	
<pre> &lt;System.Runtime.CompilerServices.ExtensionAttribute()&gt; Public Overloads Shared Function LiveMax     (Of TSource,TResult)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,TResult)) _ ) As AggregationView(Of TSource,TResult) </pre>	
C#	
<pre> [System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,TResult&gt; LiveMax&lt;TSource,TResult&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,TResult&gt;&gt; selector ) </pre>	

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

*TResult*

The type of the value returned by *selector*.

## Return Value

A view representing the maximum of the values.

## Remarks

If type *TResult* implements **System.IComparable`1**, this method uses that implementation to compare values. Otherwise, if type *TResult* implements **System.IComparable**, that implementation is used to compare values.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax<TSource>(View<TSource>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of elements of a generic type.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMax(Of TSource)( _
    ByVal source As View(Of TSource) _
) As AggregationView(Of TSource,TSource)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,TSource> LiveMax<TSource>(
    View<TSource> source
```

```
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If type *TSource* implements **System.IComparable<T>**, this method uses that implementation to compare values. Otherwise, if type *TSource* implements **System.IComparable**, that implementation is used to compare values.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax(View<Int32>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Int32>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax( _     ByVal source As View(Of Integer) _ ) As AggregationView(Of Integer,Integer)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;int,int&gt; LiveMax(     View&lt;int&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

### Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

### LiveMax(View<Nullable<Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Nullable<Int32>>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of nullable **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax( _     ByVal source As View(Of Nullable(Of Integer)) _ ) As AggregationView(Of Nullable(Of Integer), Nullable(Of Integer))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;int&gt;, Nullable&lt;int&gt;&gt; LiveMax(     View&lt;Nullable&lt;int&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

### Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm,

will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMax(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Integer))) _
) As AggregationView(Of TSource,Integer)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,int> LiveMax<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,int>> selector
```

```
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of nullable **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Integer))) _ ) As AggregationView(Of TSource,Nullable(Of Integer))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;int&gt;&gt; LiveMax&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;int&gt;&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

### Return Value

A view representing the maximum of the values.



## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax(View<Decimal>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Decimal>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of **System.Decimal** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMax( _  
    ByVal source As View(Of Decimal) _  
) As AggregationView(Of Decimal,Decimal)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<decimal,decimal> LiveMax(  
    View<decimal> source
```

```
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMax(View<Nullable<Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax(View<Nullable<Decimal>>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of nullable **System.Decimal** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax( _     ByVal source As View(Of Nullable(Of Decimal)) _ ) As AggregationView(Of Nullable(Of Decimal),Nullable(Of Decimal))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;decimal&gt;,Nullable&lt;decimal&gt;&gt; LiveMax(     View&lt;Nullable&lt;decimal&gt;&gt; source )</pre>	

## Parameters

*source*

A view containing the values to determine the maximum value of.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMax(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Decimal))) _
) As AggregationView(Of TSource,Decimal)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,decimal> LiveMax<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,decimal>> selector
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of nullable **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>>  
Public Overloads Shared Function LiveMax(Of TSource)( _  
    ByVal source As View(Of TSource), _
```

```
ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Nullable(Of Decimal))) _
) As AggregationView(Of

```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,Nullable<decimal>> LiveMax<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,Nullable<decimal>>>
selector
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

#### LiveMax(View<Int64>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Int64>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of **System.Int64** values.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMax( _  
    ByVal source As View(Of Long) _  
) As AggregationView(Of Long,Long)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<long,long> LiveMax(  
    View<long> source  
)
```

### Parameters

*source*

A view containing the values to determine the maximum value of.

### Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the

source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax(View<Nullable<Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Nullable<Int64>>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of nullable **System.Int64** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;&gt; Public Overloads Shared Function LiveMax( _     ByVal source As View(Of Nullable(Of Long)) _ ) As AggregationView(Of Nullable(Of Long),Nullable(Of Long))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;long&gt;,Nullable&lt;long&gt;&gt; LiveMax(     View&lt;Nullable&lt;long&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.



## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>>  
Public Overloads Shared Function LiveMax(Of TSource)( _
```

```

ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource, Long)) _
) As AggregationView(Of TSource, Long)

```

C#

```

[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource, long> LiveMax<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource, long>> selector
)

```

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [LiveViewExtensions Class](#)
- [LiveViewExtensions Members](#)
- [Overload List](#)

`LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>)` Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :  
`LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>)` Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of nullable **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Long))) _ ) As AggregationView(Of TSource,Nullable(Of Long))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;long&gt;&gt; LiveMax&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;long&gt;&gt;&gt; selector )</pre>	

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveMin Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : LiveMin Method

Computes the minimum value of a view of nullable **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Overload List

Overload	Description
<a href="#">LiveMin&lt;TSource&gt;(View&lt;TSource&gt;, Expression&lt;Func&lt;TSource, Nullable&lt;Double&gt;&gt;&gt;)</a>	Computes the minimum value

<code>e&gt;&gt;&gt;)</code>	<p>of a view of nullable <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.</p>
<code>LiveMin(View&lt;Single&gt;)</code>	<p>Computes the minimum value of a view of <b>System.Single</b> values.</p>
<code>LiveMin(View&lt;Nullable&lt;Single&gt;&gt;)</code>	<p>Computes the minimum value of a view of nullable <b>System.Single</b> values.</p>
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Single&gt;&gt;)</code>	<p>Computes the minimum value of a view of <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source</p>

	view.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Single&gt;&gt;&gt;&gt;)</code>	Computes the minimum value of a view of nullable <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMin&lt;TSource,TResult&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,TResult&gt;&gt;&gt;)</code>	Invokes a transform function on each element of a view of elements of a generic type and computes the minimum resulting value.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;)</code>	Computes the minimum value of a view of elements of a generic type.
<code>LiveMin(View&lt;Int32&gt;)</code>	Computes the minimum value of a view of

	<b>System.Int32</b> values.
<code>LiveMin(View&lt;Nullable&lt;Int32&gt;&gt;)</code>	Computes the minimum value of a view of nullable <b>System.Int32</b> values.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int32&gt;&gt;)</code>	Computes the minimum value of a view of <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int32&gt;&gt;&gt;&gt;)</code>	Computes the minimum value of a view of nullable <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.

LiveMin(View<Decimal>)	Computes the minimum value of a view of <b>System.Decimal</b> values.
LiveMin(View<Nullable<Decimal>>)	Computes the minimum value of a view of nullable <b>System.Decimal</b> values.
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>)	Computes the minimum value of a view of <b>System.Decimal</b> values that are obtained by invoking a transform function on each element of the source view.
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>)	Computes the minimum value of a view of nullable <b>System.Decimal</b> values that are obtained by invoking a transform function on each element



	of the source view.
<code>LiveMin(View&lt;Int64&gt;)</code>	Computes the minimum value of a view of <b>System.Int64</b> values.
<code>LiveMin(View&lt;Nullable&lt;Int64&gt;&gt;)</code>	Computes the minimum value of a view of nullable <b>System.Int64</b> values.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int64&gt;&gt;)</code>	Computes the minimum value of a view of <b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int64&gt;&gt;&gt;)</code>	Computes the minimum value of a view of nullable <b>System.Int64</b> values that are obtained by invoking a

	transform function on each element of the source view.
<code>LiveMin(View&lt;Double&gt;)</code>	Computes the minimum value of a view of <b>System.Double</b> values.
<code>LiveMin(View&lt;Nullable&lt;Double&gt;&gt;)</code>	Computes the minimum value of a view of nullable <b>System.Double</b> values.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Double&gt;&gt;)</code>	Computes the minimum value of a view of <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

`LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>)` Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

`LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>)` Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of nullable **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMin(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Nullable(Of Double))) _
) As AggregationView(Of TSource,Nullable(Of Double))
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,Nullable<double>> LiveMin<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,Nullable<double>>>
selector
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin(View<Single>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Single>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of **System.Single** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMin( _  
    ByVal source As View(Of Single) _
```

```
) As AggregationView(Of Single,Single)
```

```
C#
```

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<float,float> LiveMin(  
    View<float> source  
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMin(View<Nullable<Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Nullable<Single>>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of nullable **System.Single** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Nullable(Of Single)) _ ) As AggregationView(Of Nullable(Of Single),Nullable(Of Single))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;float&gt;,Nullable&lt;float&gt;&gt; LiveMin(     View&lt;Nullable&lt;float&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

### Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

**LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method**

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

**LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method**

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Single))) _ ) As AggregationView(Of TSource,Single)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,float&gt; LiveMin&lt;TSource&gt;(      View&lt;TSource&gt; source,      System.Linq.Expressions.Expression&lt;Func&lt;TSource,float&gt;&gt; selector  )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of nullable **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
----------------------------	--



```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMin(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Nullable(Of Single)))) _
) As AggregationView(Of TSource,Nullable(Of Single))
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,Nullable<float>> LiveMin<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,Nullable<float>>> selector
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource,TResult>(View<TSource>,Expression<Func<TSource,TResult>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource,TResult>(View<TSource>,Expression<Func<TSource,TResult>>) Method

The type of the elements of *source*.

The type of the value returned by *selector*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Invokes a transform function on each element of a view of elements of a generic type and computes the minimum resulting value.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin     (Of TSource,TResult)( _         ByVal source As View(Of TSource), _         ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,TResult)) _     ) As AggregationView(Of TSource,TResult)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,TResult&gt; LiveMin&lt;TSource,TResult&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,TResult&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

*TResult*

The type of the value returned by *selector*.

## Return Value

A view representing the minimum of the values.

## Remarks

If type *TResult* implements **System.IComparable<T>**, this method uses that implementation to compare values. Otherwise, if type *TResult* implements **System.IComparable**, that implementation is used to compare values.

It is possible to use standard LINQ query operator **Min** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :  
LiveMin<TSource>(View<TSource>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of elements of a generic type.

Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin(Of TSource)( _     ByVal source As View(Of TSource) _ ) As AggregationView(Of TSource,TSource)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,TSource&gt; LiveMin&lt;TSource&gt;(     View&lt;TSource&gt; source )</pre>	

Parameters

*source*

A view containing the values to determine the minimum value of.

Type Parameters

*TSource*

The type of the elements of *source*.

Return Value

A view representing the minimum of the values.

Remarks

If type *TSource* implements **System.IComparable`1**, this method uses that implementation to compare values. Otherwise, if type *TSource* implements **System.IComparable**, that implementation is used to compare values.

It is possible to use standard LINQ query operator **Min** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin(View<Int32>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Int32>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Integer) _ ) As AggregationView(Of Integer,Integer)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;int,int&gt; LiveMin(     View&lt;int&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin(View<Nullable<Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Nullable<Int32>>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of nullable **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Nullable(Of Integer)) _ ) As AggregationView(Of Nullable(Of Integer),Nullable(Of Integer))</pre>	

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<Nullable<int>,Nullable<int>> LiveMin(  
    View<Nullable<int>> source  
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Integer))) _ ) As AggregationView(Of TSource,Integer)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,int&gt; LiveMin&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,int&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

### Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.



It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of nullable **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMin(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Nullable(Of Integer)))) _
) As AggregationView(Of TSource,Nullable(Of Integer))
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,Nullable<int>> LiveMin<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,Nullable<int>>> selector
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveMin(View<Decimal>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Decimal>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of **System.Decimal** values.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMin( _  
    ByVal source As View(Of Decimal) _  
) As AggregationView(Of Decimal,Decimal)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<decimal,decimal> LiveMin(  
    View<decimal> source  
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin(View<Nullable<Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin(View<Nullable<Decimal>>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of nullable **System.Decimal** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Nullable(Of Decimal)) _ ) As AggregationView(Of Nullable(Of Decimal),Nullable(Of Decimal))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;decimal&gt;,Nullable&lt;decimal&gt;&gt; LiveMin(     View&lt;Nullable&lt;decimal&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

### Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Decimal))) _ ) As AggregationView(Of TSource,Decimal)</pre>	

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,decimal> LiveMin<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,decimal>> selector
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of nullable **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Decimal)))) _ ) As AggregationView(Of TSource,Nullable(Of Decimal))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;decimal&gt;&gt; LiveMin&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;decimal&gt;&gt;&gt; selector )</pre>	

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin(View<Int64>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Int64>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of **System.Int64** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMin( _  
    ByVal source As View(Of Long) _
```



```
) As AggregationView(Of Long,Long)
```

```
C#
```

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<long,long> LiveMin(  
    View<long> source  
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMin(View<Nullable<Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Nullable<Int64>>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of nullable **System.Int64** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Nullable(Of Long)) _ ) As AggregationView(Of Nullable(Of Long),Nullable(Of Long))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;long&gt;,Nullable&lt;long&gt;&gt; LiveMin(     View&lt;Nullable&lt;long&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

### Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

**LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Int64>>)** Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

**LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Int64>>)** Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Long))) _ ) As AggregationView(Of TSource,Long)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,long&gt; LiveMin&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,long&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>)) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>)) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of nullable **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMin(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Nullable(Of Long)))) _
) As AggregationView(Of TSource,Nullable(Of Long))
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,Nullable<long>> LiveMin<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,Nullable<long>>> selector
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin(View<Double>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Double>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of **System.Double** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Double) _ ) As AggregationView(Of Double,Double)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;double,double&gt; LiveMin(     View&lt;double&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

### Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin(View<Nullable<Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin(View<Nullable<Double>>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of nullable **System.Double** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Nullable(Of Double)) _ ) As AggregationView(Of Nullable(Of Double),Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;double&gt;,Nullable&lt;double&gt;&gt; LiveMin(     View&lt;Nullable&lt;double&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
----------------------------	--



```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMin(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Double)) _
) As AggregationView(Of TSource,Double)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,double> LiveMin<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,double>> selector
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

### LiveSum Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : LiveSum Method

Computes the sum of a view of **System.Int32** values.

## Overload List

Overload	Description
<a href="#">LiveSum(View&lt;Int32&gt;)</a>	Computes the sum of a view of <b>System.Int32</b> values.
<a href="#">LiveSum(View&lt;Nullable&lt;Int32&gt;&gt;)</a>	Computes the sum of a view of nullable <b>System.Int32</b> values.
<a href="#">LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int32&gt;&gt;)</a>	Computes the sum of a view of <b>System.Int32</b> values that are obtained by invoking a transform

	function on each element of the source view.
<code>LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int32&gt;&gt;&gt;)</code>	Computes the sum of a view of nullable <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveSum(View&lt;Decimal&gt;)</code>	Computes the sum of a view of <b>System.Decimal</b> values.
<code>LiveSum(View&lt;Nullable&lt;Decimal&gt;&gt;)</code>	Computes the sum of a view of nullable <b>System.Decimal</b> values.
<code>LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Decimal&gt;&gt;)</code>	Computes the sum of a view of <b>System.Decimal</b> values that are obtained by invoking a

	transform function on each element of the source view.
<code>LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Decimal&gt;&gt;&gt;)</code>	Computes the sum of a view of nullable <b>System.Decimal</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveSum(View&lt;Int64&gt;)</code>	Computes the sum of a view of <b>System.Int64</b> values.
<code>LiveSum(View&lt;Nullable&lt;Int64&gt;&gt;)</code>	Computes the sum of a view of nullable <b>System.Int64</b> values.
<code>LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int64&gt;&gt;)</code>	Computes the sum of a view of <b>System.Int64</b> values that are obtained by

	invoking a transform function on each element of the source view.
<code>LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int64&gt;&gt;&gt;&gt;)</code>	Computes the sum of a view of nullable <b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveSum(View&lt;Double&gt;)</code>	Computes the sum of a view of <b>System.Double</b> values.
<code>LiveSum(View&lt;Nullable&lt;Double&gt;&gt;)</code>	Computes the sum of a view of nullable <b>System.Double</b> values.
<code>LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Double&gt;&gt;)</code>	Computes the sum of a view of <b>System.Double</b> values that

	are obtained by invoking a transform function on each element of the source view.
<code>LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Double&gt;&gt;&gt;)</code>	Computes the sum of a view of nullable <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveSum(View&lt;Single&gt;)</code>	Computes the sum of a view of <b>System.Single</b> values.
<code>LiveSum(View&lt;Nullable&lt;Single&gt;&gt;)</code>	Computes the sum of a view of nullable <b>System.Single</b> values.
<code>LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Single&gt;&gt;)</code>	Computes the sum of a view of <b>System.Single</b>

	values that are obtained by invoking a transform function on each element of the source view.
<a href="#">LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Single&gt;&gt;&gt;)</a>	Computes the sum of a view of nullable <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

**LiveSum(View<Int32>) Method**

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Int32>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of **System.Int32** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveSum( _  
    ByVal source As View(Of Integer) _  
) As AggregationView(Of Integer,Integer)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<int,int> LiveSum(  
    View<int> source  
)
```

### Parameters

*source*

A view containing the values to calculate the sum of.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveSum(View<Nullable<Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Nullable<Int32>>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of nullable **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Nullable(Of Integer)) _ ) As AggregationView(Of Nullable(Of Integer),Nullable(Of Integer))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;int&gt;,Nullable&lt;int&gt;&gt; LiveSum(     View&lt;Nullable&lt;int&gt;&gt; source )</pre>	

## Parameters

*source*

A view containing the values to calculate the sum of.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveSum(Of TSource)( _  
    ByVal source As View(Of TSource), _  
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of  
TSource,Integer))) _  
) As AggregationView(Of TSource,Integer)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,int> LiveSum<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,int>> selector  
)
```

### Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of nullable **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Integer)))) _ ) As AggregationView(Of TSource,Nullable(Of Integer))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;int&gt;&gt; LiveSum&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;int&gt;&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum(View<Decimal>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Decimal>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of **System.Decimal** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Decimal) _ ) As AggregationView(Of Decimal,Decimal)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;decimal,decimal&gt; LiveSum(     View&lt;decimal&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the sum of.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum(View<Nullable<Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :  
LiveSum(View<Nullable<Decimal>>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of nullable **System.Decimal** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveSum( _  
    ByVal source As View(Of Nullable(Of Decimal)) _  
) As AggregationView(Of Nullable(Of Decimal), Nullable(Of Decimal))
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<Nullable<decimal>,Nullable<decimal>> LiveSum(  
    View<Nullable<decimal>> source  
)
```

### Parameters

*source*

A view containing the values to calculate the sum of.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Decimal))) _ ) As AggregationView(Of TSource,Decimal)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,decimal&gt; LiveSum&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,decimal&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.



It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of nullable **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Decimal)))) _ ) As AggregationView(Of TSource,Nullable(Of Decimal))</pre>	
C#	

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,Nullable<decimal>> LiveSum<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,Nullable<decimal>>>  
    selector  
)
```

## Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveSum(View<Int64>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Int64>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of **System.Int64** values.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveSum( _  
    ByVal source As View(Of Long) _  
) As AggregationView(Of Long,Long)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<long,long> LiveSum(  
    View<long> source  
)
```

## Parameters

*source*

A view containing the values to calculate the sum of.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum(View<Nullable<Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Nullable<Int64>>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of nullable **System.Int64** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Nullable(Of Long)) _ ) As AggregationView(Of Nullable(Of Long),Nullable(Of Long))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;long&gt;,Nullable&lt;long&gt;&gt; LiveSum(     View&lt;Nullable&lt;long&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the sum of.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveSum(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Long)) _
```

```
) As AggregationView(Of TSource,Long)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,long> LiveSum<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,long>> selector  
)
```

## Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

`LiveSum<TSource>(View<TSource>, Expression<Func<TSource, Nullable<Int64>>>) Method`

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

`LiveSum<TSource>(View<TSource>, Expression<Func<TSource, Nullable<Int64>>>) Method`

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of nullable **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveSum(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource, Nullable(Of Long)))) _
) As AggregationView(Of TSource, Nullable(Of Long))
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource, Nullable<long>> LiveSum<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource, Nullable<long>>> selector
)
```

## Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum(View<Double>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Double>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of **System.Double** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveSum( _  
    ByVal source As View(Of Double) _
```



```
) As AggregationView(Of Double,Double)
```

```
C#
```

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<double,double> LiveSum(  
    View<double> source  
)
```

## Parameters

*source*

A view containing the values to calculate the sum of.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveSum(View<Nullable<Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum(View<Nullable<Double>>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of nullable **System.Double** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Nullable(Of Double)) _ ) As AggregationView(Of Nullable(Of Double),Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;double&gt;,Nullable&lt;double&gt;&gt; LiveSum(     View&lt;Nullable&lt;double&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the sum of.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Double)) _ ) As AggregationView(Of TSource,Double)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,double&gt; LiveSum&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,double&gt;&gt; selector )</pre>	

## Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of nullable **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveSum(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Nullable(Of Double)))) _
) As AggregationView(Of TSource,Nullable(Of Double))
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,Nullable<double>> LiveSum<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,Nullable<double>>>
selector
)
```

## Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum(View<Single>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Single>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of **System.Single** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Single) _ ) As AggregationView(Of Single,Single)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;float,float&gt; LiveSum(     View&lt;float&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the sum of.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum(View<Nullable<Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum(View<Nullable<Single>>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of nullable **System.Single** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Nullable(Of Single)) _ ) As AggregationView(Of Nullable(Of Single),Nullable(Of Single))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;float&gt;,Nullable&lt;float&gt;&gt; LiveSum(     View&lt;Nullable&lt;float&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the sum of.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
----------------------------	--



```

<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveSum(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Single))) _
) As AggregationView(Of TSource,Single)

```

C#

```

[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,float> LiveSum<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,float>> selector
)

```

## Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of nullable **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveSum(Of TSource)( _  
    ByVal source As View(Of TSource), _  
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of  
TSource,Nullable(Of Single)))) _  
) As AggregationView(Of TSource,Nullable(Of Single))
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,Nullable<float>> LiveSum<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,Nullable<float>>> selector  
)
```

### Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## Ordering<T>

[C1.LiveLinq Namespace](#) : Ordering<T> Class

The type of the elements of the collection.

Represents a sorted [C1.LiveLinq.Indexing.IIndexedSource<T>](#).

## Object Model

## Syntax

Visual Basic (Declaration)

```
Public Class Ordering(Of T)
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T)
```

C#

```
public class Ordering<T> : C1.LiveLinq.Indexing.IIndexedSource<T>
```

## Type Parameters

*T*

The type of the elements of the collection.

## Remarks

An **Ordering<T>** can be obtained as a result of an [OrderBy](#) or [OrderByDescending](#) query operator. Query operators (extension methods) [ThenBy](#) and [ThenByDescending](#) operate on objects of type **Ordering<T>**.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Ordering<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Ordering<T> Members](#)

[C1.LiveLinq Namespace](#)

## Overview

[C1.LiveLinq Namespace](#) : Ordering<T> Class

The type of the elements of the collection.

Represents a sorted [C1.LiveLinq.Indexing.IIndexedSource<T>](#).

## Object Model

**Ordering<T>**

## Syntax

Visual Basic (Declaration)

```
Public Class Ordering(Of T)  
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T)
```

C#

```
public class Ordering<T> : C1.LiveLinq.Indexing.IIndexedSource<T>
```

## Type Parameters

*T*

The type of the elements of the collection.

## Remarks

An **Ordering<T>** can be obtained as a result of an [OrderBy](#) or [OrderByDescending](#) query operator. Query operators (extension methods) [ThenBy](#) and [ThenByDescending](#) operate on objects of type **Ordering<T>**.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Ordering<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Ordering<T> Members](#)

[C1.LiveLinq Namespace](#)

## Members

### Methods

[C1.LiveLinq Namespace](#) : [Ordering<T>](#) Class

The following tables list the members exposed by [Ordering<T>](#).

### Public Methods

	Name	Description
⇒	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">Ordering&lt;T&gt;</a> .
⇒	<a href="#">ThenBy&lt;TKey&gt;</a>	Performs a subsequent ordering of the elements in a collection in ascending order according to a key.
⇒	<a href="#">ThenByDescending&lt;TKey&gt;</a>	Performs a subsequent ordering of the elements in a collection in descending order according to a key.

[Top](#)

## See Also

### Reference

[Ordering<T>](#) Class

[C1.LiveLinq Namespace](#)


## Methods

[C1.LiveLinq Namespace](#) : [Ordering<T>](#) Class

For a list of all members of this type, see [Ordering<T>](#) members.

### Public Methods

	Name	Description
⇒	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">Ordering&lt;T&gt;</a> .
⇒	<a href="#">ThenBy&lt;TKey&gt;</a>	Performs a subsequent ordering of the elements in a collection in ascending order according to a key.

	<a href="#">ThenByDescending&lt;TKey&gt;</a>	Performs a subsequent ordering of the elements in a collection in descending order according to a key.
---	--	--

[Top](#)

## See Also

### Reference

[Ordering<T> Class](#)

[C1.LiveLinq Namespace](#)

### GetEnumerator Method

[C1.LiveLinq Namespace](#) > [Ordering<T> Class](#) : GetEnumerator Method

Returns an enumerator that iterates through the [Ordering<T>](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Function GetEnumerator() As System.Collections.Generic.IEnumerator(Of T)</pre>	
C#	
<pre>public System.Collections.Generic.IEnumerator&lt;T&gt; GetEnumerator()</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Ordering<T> Class](#)

[Ordering<T> Members](#)

### ThenBy<TKey> Method

[C1.LiveLinq Namespace](#) > [Ordering<T> Class](#) : ThenBy<TKey>(Expression<Func<T,TKey>>) Method

The type of the key returned by *keySelector*.

A function to extract a key from each element.

Performs a subsequent ordering of the elements in a collection in ascending order according to a key.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function ThenBy(Of TKey)( _     ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of T, TKey)) _ ) As Ordering(Of T)</pre>	
C#	
<pre>public Ordering&lt;T&gt; ThenBy&lt;TKey&gt;(     System.Linq.Expressions.Expression&lt;Func&lt;T, TKey&gt;&gt; keySelector )</pre>	

### Parameters

*keySelector*

A function to extract a key from each element.

### Type Parameters

*TKey*

The type of the key returned by *keySelector*.

### Return Value

A collection whose elements are sorted according to a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Ordering<T> Class](#)

[Ordering<T> Members](#)



## ThenByDescending<TKey> Method

[C1.LiveLinq Namespace](#) > [Ordering<T> Class](#) : ThenByDescending<TKey> Method

The type of the key returned by *keySelector*.

A function to extract a key from each element.

Performs a subsequent ordering of the elements in a collection in descending order according to a key.

## Syntax

Visual Basic (Declaration)

```
Public Function ThenByDescending(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
) As Ordering(Of T)
```

C#

```
public Ordering<T> ThenByDescending<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector  
)
```

## Parameters

*keySelector*

A function to extract a key from each element.

## Type Parameters

*TKey*

The type of the key returned by *keySelector*.

## Return Value

A collection whose elements are sorted in descending order according to a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Ordering<T> Class](#)

[Ordering<T> Members](#)

# QueryOptimizationException

[C1.LiveLinq Namespace](#) : QueryOptimizationException Class

Represents an exception that is thrown when [Hints](#) in a query require using a certain mandatory optimization which is impossible in the current query context.

## Object Model

QueryOptimizationException

## Syntax

Visual Basic (Declaration)

```
<System.SerializableAttribute(>>  
Public Class QueryOptimizationException  
    Inherits System.Exception
```

C#

```
[System.Serializable()]  
public class QueryOptimizationException : System.Exception
```

## Inheritance Hierarchy

System.Object

System.Exception

**C1.LiveLinq.QueryOptimizationException**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[QueryOptimizationException Members](#)

[C1.LiveLinq Namespace](#)

## Overview

[C1.LiveLinq Namespace](#) : QueryOptimizationException Class

Represents an exception that is thrown when [Hints](#) in a query require using a certain mandatory optimization which is impossible in the current query context.

## Object Model

QueryOptimizationException

## Syntax

Visual Basic (Declaration)

```
<System.SerializableAttribute(>>  
Public Class QueryOptimizationException  
    Inherits System.Exception
```

C#

```
[System.Serializable()]  
public class QueryOptimizationException : System.Exception
```

## Inheritance Hierarchy

System.Object

System.Exception

**C1.LiveLinq.QueryOptimizationException**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[QueryOptimizationException Members](#)

[C1.LiveLinq Namespace](#)


## Members

[Properties](#) [Methods](#) [Events](#)

C1.LiveLinq Namespace : QueryOptimizationException Class









The following tables list the members exposed by [QueryOptimizationException](#).

## Public Constructors

	Name	Description
	<a href="#">QueryOptimizationException Constructor</a>	Overloaded.



[Top](#)



## Public Properties

	Name	Description
	<a href="#">Data</a>	(Inherited from System.Exception)
	<a href="#">HelpLink</a>	(Inherited from System.Exception)
	<a href="#">HResult</a>	(Inherited from System.Exception)
	<a href="#">InnerException</a>	(Inherited from System.Exception)
	<a href="#">Message</a>	(Inherited from System.Exception)
	<a href="#">Source</a>	(Inherited from System.Exception)
	<a href="#">StackTrace</a>	(Inherited from System.Exception)
	<a href="#">TargetSite</a>	(Inherited from System.Exception)

[Top](#)


## Public Methods

	Name	Description
	<a href="#">GetBaseException</a>	(Inherited from System.Exception)
	<a href="#">GetObjectData</a>	(Inherited from System.Exception)

	<a href="#">GetType</a>	(Inherited from System.Exception)
	<a href="#">ToString</a>	(Inherited from System.Exception)

[Top](#)

## Protected Events

	Name	Description
	<a href="#">SerializeObjectState</a>	(Inherited from System.Exception)

[Top](#)

## See Also

### Reference

[QueryOptimizationException Class](#)

[C1.LiveLinq Namespace](#)

## QueryOptimizationException Constructor

[C1.LiveLinq Namespace](#) > [QueryOptimizationException Class](#) : QueryOptimizationException Constructor

## Overload List

Overload	Description
<a href="#">QueryOptimizationException Constructor()</a>	Initializes a new instance of the <a href="#">QueryOptimizationException</a> class. This is the default constructor.
<a href="#">QueryOptimizationException Constructor(String)</a>	Initializes a new instance of the <a href="#">QueryOptimizationException</a> class with the specified string.
<a href="#">QueryOptimizationException Constructor(String,Exception)</a>	Initializes a new instance

	of the <a href="#">QueryOptimizationException</a> class using the specified string and inner exception.
<a href="#">QueryOptimizationException</a> <a href="#">Constructor(SerializationInfo,StreamingContext)</a>	Initializes a new instance of the <a href="#">QueryOptimizationException</a> class from serialized data.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[QueryOptimizationException Class](#)

[QueryOptimizationException Members](#)

### QueryOptimizationException Constructor()

[C1.LiveLinq Namespace](#) > [QueryOptimizationException Class](#) > [QueryOptimizationException Constructor](#) : [QueryOptimizationException Constructor\(\)](#)

Initializes a new instance of the [QueryOptimizationException](#) class. This is the default constructor.

## Syntax

Visual Basic (Declaration)	
<b>Public Function New()</b>	
C#	
<b>public</b> QueryOptimizationException()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[QueryOptimizationException Class](#)  
[QueryOptimizationException Members](#)  
[Overload List](#)

### QueryOptimizationException Constructor(String)

[C1.LiveLinq Namespace](#) > [QueryOptimizationException Class](#) > [QueryOptimizationException Constructor](#) :  
QueryOptimizationException Constructor(String)

The string to display when the exception is thrown.

Initializes a new instance of the [QueryOptimizationException](#) class with the specified string.

## Syntax

Visual Basic (Declaration)

```
Public Function New( _  
    ByVal message As System.String _  
)
```

C#

```
public QueryOptimizationException(  
    System.string message  
)
```

### Parameters

*message*

The string to display when the exception is thrown.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[QueryOptimizationException Class](#)  
[QueryOptimizationException Members](#)  
[Overload List](#)

### QueryOptimizationException Constructor(String,Exception)

[C1.LiveLinq Namespace](#) > [QueryOptimizationException Class](#) > [QueryOptimizationException Constructor](#) :  
QueryOptimizationException Constructor(String,Exception)

The string to display when the exception is thrown.

Gets the **Exception** instance that caused the current exception.

Initializes a new instance of the [QueryOptimizationException](#) class using the specified string and inner exception.

## Syntax

Visual Basic (Declaration)

```
Public Function New( _  
    ByVal message As System.String, _  
    ByVal inner As System.Exception _  
)
```

C#

```
public QueryOptimizationException(  
    System.string message,  
    System.Exception inner  
)
```

### Parameters

*message*

The string to display when the exception is thrown.

*inner*

Gets the **Exception** instance that caused the current exception.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2



## See Also

### Reference

[QueryOptimizationException Class](#)  
[QueryOptimizationException Members](#)  
[Overload List](#)

### QueryOptimizationException Constructor(SerializationInfo,StreamingContext)

[C1.LiveLinq Namespace](#) > [QueryOptimizationException Class](#) > [QueryOptimizationException Constructor](#) :  
QueryOptimizationException Constructor(SerializationInfo,StreamingContext)

The object that holds the serialized object data.

The contextual information about the source or destination.

Initializes a new instance of the [QueryOptimizationException](#) class from serialized data.

## Syntax

Visual Basic (Declaration)

```
Protected Function New( _  
    ByVal info As System.Runtime.Serialization.SerializationInfo, _  
    ByVal context As System.Runtime.Serialization.StreamingContext _  
)
```

C#

```
protected QueryOptimizationException(  
    System.Runtime.Serialization.SerializationInfo info,  
    System.Runtime.Serialization.StreamingContext context  
)
```

### Parameters

*info*

The object that holds the serialized object data.

*context*

The contextual information about the source or destination.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[QueryOptimizationException Class](#)  
[QueryOptimizationException Members](#)  
[Overload List](#)

## SourceChangeEventArgs<T>

[C1.LiveLinq Namespace](#) : [SourceChangeEventArgs<T> Class](#)

Type of the changed object.

Provides data for the [IObservableSource<T>.Changed](#) event.

## Object Model

SourceChangeEventArgs<T>

## Syntax

Visual Basic (Declaration)	
<pre>Public Class SourceChangeEventArgs(Of T)     Inherits System.EventArgs</pre>	
C#	
<pre>public class SourceChangeEventArgs&lt;T&gt; : System.EventArgs</pre>	

## Type Parameters

*T*

Type of the changed object.

## Inheritance Hierarchy

System.Object  
  System.EventArgs  
    **C1.LiveLinq.SourceChangeEventArgs<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Members](#)  
[C1.LiveLinq Namespace](#)

### Overview

[C1.LiveLinq Namespace](#) : [SourceChangeEventArgs<T>](#) Class

Type of the changed object.

Provides data for the [IObservableSource<T>.Changed](#) event.

## Object Model

[SourceChangeEventArgs<T>](#)

## Syntax

Visual Basic (Declaration)

```
Public Class SourceChangeEventArgs(Of T)  
    Inherits System.EventArgs
```

C#

```
public class SourceChangeEventArgs<T> : System.EventArgs
```

## Type Parameters

*T*

Type of the changed object.

## Inheritance Hierarchy

System.Object  
  System.EventArgs  
    **C1.LiveLinq.SourceChangeEventArgs<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Members](#)  
[C1.LiveLinq Namespace](#)


### Members

[Properties](#) [Methods](#)

[C1.LiveLinq Namespace](#) : [SourceChangeEventArgs<T>](#) Class




The following tables list the members exposed by [SourceChangeEventArgs<T>](#).

### Public Constructors

	Name	Description
	<a href="#">SourceChangeEventArgs&lt;T&gt; Constructor</a>	Initializes a new instance of the <a href="#">SourceChangeEventArgs&lt;T&gt;</a> class.



[Top](#)


### Public Properties

	Name	Description
	<a href="#">ChangeType</a>	Gets the type of change.
	<a href="#">Item</a>	Gets the object that is being changed.
	<a href="#">Ordinal</a>	Gets the ordinal position of the collection item that is being changed.

[Top](#)

### Public Methods

	Name	Description
	<a href="#">Equals</a>	Overloaded. Determines whether the specified object is equal to the current object.
	<a href="#">GetHashCode</a>	Return a hash code for this instance.

	<b>To String</b>	Returns a string that represents this instance of <code>SourceChangeEventArgs&lt;T&gt;</code> .
---	------------------	---

[Top](#)

See Also

Reference

[SourceChangeEventArgs<T> Class](#)  
[C1.LiveLinq Namespace](#)

SourceChangeEventArgs<T> Constructor

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : SourceChangeEventArgs<T> Constructor

Changed object.

Type of change.

Ordinal position of the changed item.

Initializes a new instance of the [SourceChangeEventArgs<T>](#) class.

Syntax

Visual Basic (Declaration)	
<pre>Public Function New( _     ByVal item As T, _     ByVal changeType As SourceChangeType, _     ByVal ordinal As System.Integer _ )</pre>	
C#	
<pre>public SourceChangeEventArgs&lt;T&gt;(     T item,     SourceChangeType changeType,     System.int ordinal )</pre>	

Parameters

*item*

Changed object.

*changeType*

Type of change.

*ordinal*

Ordinal position of the changed item.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)




[SourceChangeEventArgs<T> Members](#)

### Methods

[C1.LiveLinq Namespace](#) : [SourceChangeEventArgs<T> Class](#)

For a list of all members of this type, see [SourceChangeEventArgs<T> members](#).

## Public Methods

	Name	Description
	<a href="#">Equals</a>	Overloaded. Determines whether the specified object is equal to the current object.
	<a href="#">GetHashCode</a>	Return a hash code for this instance.
	<a href="#">ToString</a>	Returns a string that represents this instance of <a href="#">SourceChangeEventArgs&lt;T&gt;</a> .

[Top](#)

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[C1.LiveLinq Namespace](#)

## Equals Method

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : Equals Method

Determines whether the specified object is equal to the current object.

## Overload List

Overload	Description
<a href="#">Equals(Object)</a>	Determines whether the specified object is equal to the current object.
<a href="#">Equals(SourceChangeEventArgs&lt;T&gt;)</a>	Determines whether the specified object is equal to the current object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[SourceChangeEventArgs<T> Members](#)

[Equals\(Object\) Method](#)

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) > [Equals Method](#) : Equals(Object) Method

The object to compare with the current object.

Determines whether the specified object is equal to the current object.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Overrides Function Equals( _     ByVal obj As System.Object _ ) As System.Boolean</pre>	
C#	

```
public override System.bool Equals(
    System.object obj
)
```

## Parameters

*obj*

The object to compare with the current object.

## Return Value

**true** if the specified object is equal to the current one; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[SourceChangeEventArgs<T> Class](#)  
[SourceChangeEventArgs<T> Members](#)  
[Overload List](#)

Equals(SourceChangeEventArgs<T>) Method

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) > [Equals Method](#) :

Equals(SourceChangeEventArgs<T>) Method

The object to compare with the current object.

Determines whether the specified object is equal to the current object.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Equals( _     ByVal args As SourceChangeEventArgs(Of T) _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool Equals(     SourceChangeEventArgs&lt;T&gt; args</pre>	



```
)
```

## Parameters

*args*

The object to compare with the current object.

## Return Value

**true** if the specified object is equal to the current one; otherwise, false.

## Remarks

Two [SourceChangeEventArgs<T>](#) are considered equal if the values of their [Item](#) and [ChangeType](#) properties are equal.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[SourceChangeEventArgs<T> Class](#)  
[SourceChangeEventArgs<T> Members](#)  
[Overload List](#)

## GetHashCode Method

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : GetHashCode Method

Return a hash code for this instance.

## Syntax

Visual Basic (Declaration)

```
Public Overrides Function GetHashCode() As System.Integer
```

C#

```
public override System.int GetHashCode()
```

## Return Value

A 32-bit signed integer hash code.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[SourceChangeEventArgs<T> Members](#)

### ToString Method

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : ToString Method

Returns a string that represents this instance of [SourceChangeEventArgs<T>](#).

## Syntax

Visual Basic (Declaration)	
<b>Public Overrides Function</b> ToString() <b>As</b> System.String	
C#	
<b>public override</b> System.string ToString()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)




[SourceChangeEventArgs<T> Members](#)

### Properties

[C1.LiveLinq Namespace](#) : [SourceChangeEventArgs<T> Class](#)

For a list of all members of this type, see [SourceChangeEventArgs<T> members](#).

## Public Properties

	Name	Description
	<a href="#">ChangeType</a>	Gets the type of change.
	<a href="#">Item</a>	Gets the object that is being changed.
	<a href="#">Ordinal</a>	Gets the ordinal position of the collection item that is being changed.

[Top](#)

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[C1.LiveLinq Namespace](#)

### ChangeType Property

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : ChangeType Property

Gets the type of change.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property ChangeType As SourceChangeType</code>	
C#	
<code>public SourceChangeType ChangeType {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[SourceChangeEventArgs<T> Members](#)

## Item Property

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : Item Property

Gets the object that is being changed.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Item As T</code>	
C#	
<code>public T Item {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[SourceChangeEventArgs<T> Members](#)

## Ordinal Property

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : Ordinal Property

Gets the ordinal position of the collection item that is being changed.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Ordinal As System.Integer</code>	
C#	
<code>public System.int Ordinal {get;}</code>	

## Remarks

This property can return -1 (ordinal unknown) if the collection cannot provide this information (if [IObservableSource<T>.SupportsItemOrdinals](#) returns **false**).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[SourceChangeEventArgs<T> Members](#)

## Enumerations

### IndexingHintAction

[C1.LiveLinq Namespace](#) : IndexingHintAction Enumeration

Specifies the actions taken by LiveLinq query optimizer when it encounters an **Indexed()** hint applied to an expression (usually, a property) in a query.

## Syntax

Visual Basic (Declaration)	
<pre>Public Enum IndexingHintAction     Inherits System.Enum</pre>	
C#	
<pre>public enum IndexingHintAction : System.Enum</pre>	

## Members

Member	Description
<b>Mandatory</b>	Create an index for this expression, if such index does not already exist. Use this index in executing the query (perform an index scan using that index). Throw an exception if it is impossible to use it in query execution.
<b>Optional</b>	Create an index for this expression, if such index does not already exist. Use this index in executing the query, if it is possible. Do not throw exception if it is impossible to use that index in query execution.

<b>UseExistingIndex</b>	Check that there exists an index for this expression. If it does not exist, throw an exception. Use this index in executing the query. Throw an exception if it is impossible to use this index in query execution.
-------------------------	---

## Inheritance Hierarchy

System.Object  
  System.ValueType  
    System.Enum  
      **C1.LiveLinq.IndexingHintAction**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.LiveLinq Namespace](#)

## Order

[C1.LiveLinq Namespace](#) : Order Enumeration

Indicates if a certain order is required in the result collection of an operation.

## Syntax

Visual Basic (Declaration)	
<pre>Public Enum Order     Inherits System.Enum</pre>	
C#	
<pre>public enum Order : System.Enum</pre>	

## Members

Member	Description
<b>Ascending</b>	The resulting collection must be ordered in ascending key order.

<b>Descending</b>	The resulting collection must be ordered in descending key order.
<b>Unordered</b>	No particular order is required in the resulting collection.

## Inheritance Hierarchy

System.Object  
  System.ValueType  
    System.Enum  
      **C1.LiveLinq.Order**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.LiveLinq Namespace](#)

## SourceChangeType

[C1.LiveLinq Namespace](#) : SourceChangeType Enumeration

Describes a change occurring in a collection.

## Syntax

Visual Basic (Declaration)	
<pre><b>Public Enum</b> SourceChangeType     <b>Inherits</b> System.Enum</pre>	
C#	
<pre><b>public enum</b> SourceChangeType : System.Enum</pre>	

## Members

Member	Description
<b>Add</b>	An item is added to the collection.

<b>Modify</b>	At least one of the properties of an item has changed its value.
<b>Remove</b>	An item is removed from the collection.
<b>Reset</b>	Multiple items have changed in the collection. Indexes must be rebuilt.

## Inheritance Hierarchy

System.Object

System.ValueType

System.Enum

**C1.LiveLinq.SourceChangeType**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.LiveLinq Namespace](#)

## TransactionState

[C1.LiveLinq Namespace](#) : TransactionState Enumeration

Enumeration of the possible states an [ITransaction](#) can be in.

## Syntax

Visual Basic (Declaration)	
<pre><b>Public Enum</b> TransactionState     <b>Inherits</b> System.Enum</pre>	
C#	
<pre><b>public enum</b> TransactionState : System.Enum</pre>	

## Members

Member	Description
--------	-------------



<b>Committed</b>	A transaction is committed.
<b>Committing</b>	A transaction is in the process of being committed.
<b>Open</b>	A transaction is open.
<b>RolledBack</b>	A transaction is rolled back.
<b>RollingBack</b>	A transaction is in the process of being rolled back.

## Inheritance Hierarchy

System.Object  
     System.ValueType  
         System.Enum  
             **C1.LiveLinq.TransactionState**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.LiveLinq Namespace](#)

## Interfaces

### IObservableSource<T>

[C1.LiveLinq Namespace](#) : IObservableSource<T> Interface

The type of the elements in the collection.

Provides methods and events that are required for LiveLinq functionality, indexing and live views.

## Object Model

IObservableSource<T>

## Syntax

Visual Basic (Declaration)	
<b>Public Interface</b> <code>IObservableSource(Of T)</code>	
C#	
<b>public interface</b> <code>IObservableSource&lt;T&gt;</code>	

## Type Parameters

*T*

The type of the elements in the collection.

## Remarks

Indexing and live view functionality is available for any collection that supports change notifications necessary for maintaining indexes and live views, that is, fires events when an item is added to or removed from the collection and when a property of the item changes. So, the members of this interface are mostly concerned with providing such notifications.

Classes implementing this interface usually also implement [C1.LiveLinq.Indexing.IIndexedSource<T>](#).

Both these interfaces are implemented by all main LiveLinq collection classes: [C1.LiveLinq.Collections.IndexedCollection<T>](#), [C1.LiveLinq.AdoNet.IndexedDataTable<TRow>](#), [C1.LiveLinq.LiveViews.View<T>](#).

You need to implement this interface only if you want to define your own indexable collection classes and then only if they don't inherit from [C1.LiveLinq.Collections.IndexedCollection<T>](#), see LiveLinq to Objects: [IndexedCollection\(T\)](#) and other collection classes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IObservableSource<T> Members](#)  
[C1.LiveLinq Namespace](#)

## Overview

[C1.LiveLinq Namespace](#) : [IObservableSource<T>](#) Interface

The type of the elements in the collection.

Provides methods and events that are required for LiveLinq functionality, indexing and live views.

## Object Model

[IObservableSource<T>](#)

## Syntax

Visual Basic (Declaration)

```
Public Interface IObservableSource(Of T)
```

C#

```
public interface IObservableSource<T>
```

## Type Parameters

*T*

The type of the elements in the collection.

## Remarks

Indexing and live view functionality is available for any collection that supports change notifications necessary for maintaining indexes and live views, that is, fires events when an item is added to or removed from the collection and when a property of the item changes. So, the members of this interface are mostly concerned with providing such notifications.

Classes implementing this interface usually also implement [C1.LiveLinq.Indexing.IIndexedSource<T>](#).

Both these interfaces are implemented by all main LiveLinq collection classes: [C1.LiveLinq.Collections.IndexedCollection<T>](#), [C1.LiveLinq.AdoNet.IndexedDataTable<TRow>](#), [C1.LiveLinq.LiveViews.View<T>](#).

You need to implement this interface only if you want to define your own indexable collection classes and then only if they don't inherit from [C1.LiveLinq.Collections.IndexedCollection<T>](#), see LiveLinq to Objects: [IndexedCollection\(T\)](#) and other collection classes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IObservableSource<T> Members](#)  
[C1.LiveLinq Namespace](#)




## Members

[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq Namespace](#) : [IObservableSource<T> Interface](#)


The following tables list the members exposed by [IObservableSource<T>](#).

## Public Properties

	Name	Description
	<a href="#">CreateNew</a>	This delegate is used to create new items. If it is null, a public parameterless constructor of type <b>T</b> is used.
	<a href="#">IsDeletedStateAvailable</a>	Gets a value indicating whether an item of this collection can still return correct property values after it has been deleted from the collection.
	<a href="#">SupportsItemOrdinals</a>	Gets a value that indicates whether this collection is capable of providing the ordinal position of the changed item when it notifies of an item change.


[Top](#)

## Public Methods

	Name	Description
	<a href="#">EnableItemOrdinals</a>	After this method is called, the collection is required to provide <a href="#">SourceChangeEventArgs&lt;T&gt;.Ordinal</a> in event data.

[Top](#)

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after an item of the collection or the entire collection has changed.

[Top](#)

## See Also

### Reference


[IObservableSource<T> Interface](#)  
[C1.LiveLinq Namespace](#)

## Methods

[C1.LiveLinq Namespace](#) : [IObservableSource<T> Interface](#)

For a list of all members of this type, see [IObservableSource<T> members](#).

## Public Methods

	Name	Description
	<a href="#">EnableItemOrdinals</a>	After this method is called, the collection is required to provide <a href="#">SourceChangeEventArgs&lt;T&gt;.Ordinal</a> in event data.

[Top](#)

## See Also

### Reference

[IObservableSource<T> Interface](#)  
[C1.LiveLinq Namespace](#)

### EnableItemOrdinals Method

[C1.LiveLinq Namespace](#) > [IObservableSource<T> Interface](#) : [EnableItemOrdinals Method](#)

After this method is called, the collection is required to provide [SourceChangeEventArgs<T>.Ordinal](#) in event data.

## Syntax

Visual Basic (Declaration)	
<b>Sub</b> EnableItemOrdinals()	
C#	
<b>void</b> EnableItemOrdinals()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IObservableSource<T> Interface](#)




[IObservableSource<T> Members](#)

## Properties

[C1.LiveLinq Namespace](#) : IObservableSource<T> Interface

For a list of all members of this type, see [IObservableSource<T> members](#).

## Public Properties

	Name	Description
	<a href="#">CreateNew</a>	This delegate is used to create new items. If it is null, a public parameterless constructor of type <b>T</b> is used.
	<a href="#">IsDeletedStateAvailable</a>	Gets a value indicating whether an item of this collection can still return correct property values after it has been deleted from the collection.
	<a href="#">SupportsItemOrdinals</a>	Gets a value that indicates whether this collection is capable of providing the ordinal position of the changed item when it notifies of an item change.

[Top](#)

## See Also

## Reference

[IObservableSource<T> Interface](#)

[C1.LiveLinq Namespace](#)

## CreateNew Property

[C1.LiveLinq Namespace](#) > [IObservableSource<T> Interface](#) : CreateNew Property

This delegate is used to create new items. If it is null, a public parameterless constructor of type **T** is used.

## Syntax

Visual Basic (Declaration)	
<b>ReadOnly Property</b> CreateNew <b>As</b> System.Func(Of T)	
C#	
System.Func<T> CreateNew { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IObservableSource<T> Interface](#)

[IObservableSource<T> Members](#)

## IsDeletedStateAvailable Property

[C1.LiveLinq Namespace](#) > [IObservableSource<T> Interface](#) : IsDeletedStateAvailable Property

Gets a value indicating whether an item of this collection can still return correct property values after it has been deleted from the collection.

## Syntax

Visual Basic (Declaration)	
<b>ReadOnly Property</b> IsDeletedStateAvailable <b>As</b> System.Boolean	
C#	

```
System.bool IsDeletedStateAvailable {get;}
```

### Property Value

**true** if deleted items can still be used to get property values.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IObservableSource<T> Interface](#)

[IObservableSource<T> Members](#)

[DeletedStateIsAvailableAttribute Class](#)

### SupportsItemOrdinals Property

[C1.LiveLinq Namespace](#) > [IObservableSource<T> Interface](#) : SupportsItemOrdinals Property

Gets a value that indicates whether this collection is capable of providing the ordinal position of the changed item when it notifies of an item change.

## Syntax

Visual Basic (Declaration)	
<b>ReadOnly Property</b> SupportsItemOrdinals <b>As</b> System.Boolean	
C#	
System. <b>bool</b> SupportsItemOrdinals { <b>get</b> ;}	

### Property Value

**true** if the collection can provide item ordinal positions.

## Remarks

If this property returns **true**, LiveLinq can call the [EnableItemOrdinals](#) method to require providing ordinals.

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2


See Also

Reference

[IObservableSource<T> Interface](#)  
[IObservableSource<T> Members](#)

Events

>

Name	Description
 <a href="#">Changed</a>	Occurs after an item of the collection or the entire collection has changed.

[Top](#)

See Also

Reference

[IObservableSource<T> Interface](#)  
[C1.LiveLinq Namespace](#)

Changed Event

[C1.LiveLinq Namespace](#) > [IObservableSource<T> Interface](#) : Changed Event

Occurs after an item of the collection or the entire collection has changed.

Syntax

Visual Basic (Declaration)	
<code>Event Changed As System.EventHandler(Of SourceChangeEventArgs(Of T))</code>	
C#	
<code>event System.EventHandler&lt;SourceChangeEventArgs&lt;T&gt;&gt; Changed</code>	

Event Data

The event handler receives an argument of type [SourceChangeEventArgs<T>](#) containing data related to this event. The following **SourceChangeEventArgs<T>** properties provide information specific to this event.

Property	Description
<a href="#">ChangeType</a>	Gets the type of change.
<a href="#">Item</a>	Gets the object that is being changed.
<a href="#">Ordinal</a>	Gets the ordinal position of the collection item that is being changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IObservableSource<T> Interface](#)

[IObservableSource<T> Members](#)

## ITransaction

[C1.LiveLinq Namespace](#) : ITransaction Interface

Represents a transaction with an explicit scope.

## Object Model

ITransaction

## Syntax

Visual Basic (Declaration)	
<b>Public Interface</b> ITransaction	
C#	
<b>public interface</b> ITransaction	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ITransaction Members](#)  
[C1.LiveLinq Namespace](#)

## Overview

[C1.LiveLinq Namespace](#) : ITransaction Interface

Represents a transaction with an explicit scope.

# Object Model

ITransaction

## Syntax

Visual Basic (Declaration)	
<b>Public Interface</b> ITransaction	
C#	
<b>public interface</b> ITransaction	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ITransaction Members](#)  
[C1.LiveLinq Namespace](#)



## Members

[Properties](#) [Methods](#)

[C1.LiveLinq Namespace](#) : ITransaction Interface




The following tables list the members exposed by [ITransaction](#).

## Public Properties

	Name	Description
	<a href="#">HasChanges</a>	Gets a value indicating whether any changes were made in the scope of this transaction.
	<a href="#">State</a>	Gets the current state of the transaction.

[Top](#)

## Public Methods

	Name	Description
	<a href="#">Commit</a>	Commits changes that were made while this transaction's scope was open.
	<a href="#">Rollback</a>	Rolls back changes that were made while this transaction's scope was open.
	<a href="#">Scope</a>	Opens a transaction scope.

[Top](#)

## See Also

### Reference

[ITransaction Interface](#)


[C1.LiveLinq Namespace](#)



## Methods

[C1.LiveLinq Namespace](#) : [ITransaction Interface](#)

For a list of all members of this type, see [ITransaction members](#).

## Public Methods

	Name	Description
	<a href="#">Commit</a>	Commits changes that were made while this transaction's scope was open.

	<a href="#">Rollback</a>	Rolls back changes that were made while this transaction's scope was open.
	<a href="#">Scope</a>	Opens a transaction scope.

[Top](#)

## See Also

### Reference

[ITransaction Interface](#)

[C1.LiveLinq Namespace](#)

### Commit Method

[C1.LiveLinq Namespace](#) > [ITransaction Interface](#) : Commit Method

Commits changes that were made while this transaction's scope was open.

## Syntax

Visual Basic (Declaration)	
<b>Sub</b> Commit()	
C#	
<b>void</b> Commit()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ITransaction Interface](#)

[ITransaction Members](#)

### Rollback Method

[C1.LiveLinq Namespace](#) > [ITransaction Interface](#) : Rollback Method

Rolls back changes that were made while this transaction's scope was open.

# Syntax

Visual Basic (Declaration)	
<code>Sub Rollback()</code>	
C#	
<code>void Rollback()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ITransaction Interface](#)

[ITransaction Members](#)

### Scope Method

[C1.LiveLinq Namespace](#) > [ITransaction Interface](#) : Scope Method

Opens a transaction scope.

# Syntax

Visual Basic (Declaration)	
<code>Function Scope() As System.IDisposable</code>	
C#	
<code>System.IDisposable Scope()</code>	

### Return Value

An instance of **System.IDisposable** that will close the scope when its **System.IDisposable.Dispose** method is called.

## Remarks

The transaction tracks changes only when they are made inside an open scope.

Calling **System.IDisposable.Dispose** on the return value closes the scope.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ITransaction Interface](#)



[ITransaction Members](#)

## Properties

[C1.LiveLinq Namespace](#) : [ITransaction Interface](#)

For a list of all members of this type, see [ITransaction members](#).

## Public Properties

	Name	Description
	<a href="#">HasChanges</a>	Gets a value indicating whether any changes were made in the scope of this transaction.
	<a href="#">State</a>	Gets the current state of the transaction.

[Top](#)

## See Also

### Reference

[ITransaction Interface](#)

[C1.LiveLinq Namespace](#)

## HasChanges Property

[C1.LiveLinq Namespace](#) > [ITransaction Interface](#) : HasChanges Property

Gets a value indicating whether any changes were made in the scope of this transaction.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<b>ReadOnly Property</b> HasChanges As System.Boolean	
C#	
System.bool HasChanges {get;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ITransaction Interface](#)

[ITransaction Members](#)

### State Property

[C1.LiveLinq Namespace](#) > [ITransaction Interface](#) : State Property

Gets the current state of the transaction.

## Syntax

Visual Basic (Declaration)	
<b>ReadOnly Property</b> State As TransactionState	
C#	
TransactionState State {get;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ITransaction Interface](#)



[ITransaction Members](#)



# C1.LiveLinq.Adonet Namespace

## Overview

### Classes

	Class	Description
	<a href="#">AdoNetExtensions</a>	Provides a set of static (extension) methods for LiveLinq to DataSet.
	<a href="#">IndexedDataTable&lt;TRow&gt;</a>	A wrapper for the standard ADO.NET <b>System.Data.DataTable</b> class allowing to use ADO.NET data sources in LiveLinq indexing and live views.

### See Also

#### Reference

[C1.LiveLinq.4 Assembly](#)

## Classes

### AdoNetExtensions

[C1.LiveLinq.Adonet Namespace](#) : AdoNetExtensions Class

Provides a set of static (extension) methods for LiveLinq to DataSet.

### Object Model

AdoNetExtensions

### Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; &lt;HintAttribute(C1.LiveLinq.Adonet.AdonetExtensions+AdonetHintConverter)&gt; Public MustInherit NotInheritable Class AdonetExtensions</pre>	
C#	

```
[System.Runtime.CompilerServices.Extension()]  
[Hint(C1.LiveLinq.AdoNet.AdoNetExtensions+AdoNetHintConverter)]  
public static class AdoNetExtensions
```

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.AdoNet.AdoNetExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Members](#)

[C1.LiveLinq.AdoNet Namespace](#)

## Overview

[C1.LiveLinq.AdoNet Namespace](#) : AdoNetExtensions Class

Provides a set of static (extension) methods for LiveLinq to DataSet.

## Object Model

AdoNetExtensions

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
<HintAttribute(C1.LiveLinq.AdoNet.AdoNetExtensions+AdoNetHintConverter)>  
Public MustInherit NotInheritable Class AdoNetExtensions
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
[Hint(C1.LiveLinq.AdoNet.AdoNetExtensions+AdoNetHintConverter)]  
public static class AdoNetExtensions
```

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.AdoNet.AdoNetExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Members](#)

[C1.LiveLinq.AdoNet Namespace](#)






## Members

[Methods](#)

[C1.LiveLinq.AdoNet Namespace](#) : AdoNetExtensions Class

The following tables list the members exposed by [AdoNetExtensions](#).

## Public Methods

	Name	Description
≡  S	<a href="#">AsIndexed</a>	Overloaded. Wraps a <b>System.Data.DataTable</b> in an <a href="#">IndexedDataTable&lt;DataRow&gt;</a> so it can be indexed and queried using the optimized query operators from <a href="#">C1.LiveLinq.IndexedQueryExtensions</a> .
≡  S	<a href="#">AsLive</a>	Overloaded. Creates a view based on the specified ADO.NET <b>System.Data.DataTable</b> .
≡  S	<a href="#">BeginUpdate</a>	Indicates that massive changes are being made in code, until <a href="#">EndUpdate</a> , to data tables in this <b>System.Data.DataSet</b> .
≡  S	<a href="#">EndUpdate</a>	Indicates the end of massive changes to data tables in this <b>System.Data.DataSet</b> started with <a href="#">BeginUpdate</a> .
≡  S	<a href="#">IndexedField</a>	Overloaded. A hint to create and use an index on the specified data

		column. The hint has default action.
--	--	--------------------------------------

[Top](#)

## See Also

### Reference

[AdoNetExtensions Class](#)






[C1.LiveLinq.AdoNet Namespace](#)

## Methods

[C1.LiveLinq.AdoNet Namespace](#) : [AdoNetExtensions Class](#)

For a list of all members of this type, see [AdoNetExtensions members](#).

## Public Methods

	Name	Description
≡  <a href="#">AsIndexed</a>	<a href="#">AsIndexed</a>	Overloaded. Wraps a <b>System.Data.DataTable</b> in an <a href="#">IndexedDataTable&lt;DataRow&gt;</a> so it can be indexed and queried using the optimized query operators from <a href="#">C1.LiveLinq.IndexedQueryExtensions</a> .
≡  <a href="#">AsLive</a>	<a href="#">AsLive</a>	Overloaded. Creates a view based on the specified ADO.NET <b>System.Data.DataTable</b> .
≡  <a href="#">BeginUpdate</a>	<a href="#">BeginUpdate</a>	Indicates that massive changes are being made in code, until <a href="#">EndUpdate</a> , to data tables in this <b>System.Data.DataSet</b> .
≡  <a href="#">EndUpdate</a>	<a href="#">EndUpdate</a>	Indicates the end of massive changes to data tables in this <b>System.Data.DataSet</b> started with <a href="#">BeginUpdate</a> .
≡  <a href="#">IndexedField</a>	<a href="#">IndexedField</a>	Overloaded. A hint to create and use an index on the specified data column. The hint has default action.

[Top](#)

## See Also

### Reference

[AdoNetExtensions Class](#)  
[C1.LiveLinq.Adonet Namespace](#)

## AsIndexed Method

[C1.LiveLinq.Adonet Namespace](#) > [AdoNetExtensions Class](#) : AsIndexed Method

Wraps a **System.Data.DataTable** in an [IndexedDataTable<DataRow>](#) so it can be indexed and queried using the optimized query operators from [C1.LiveLinq.IndexedQueryExtensions](#).

## Overload List

Overload	Description
<a href="#">AsIndexed(DataTable)</a>	Wraps a <b>System.Data.DataTable</b> in an <a href="#">IndexedDataTable&lt;DataRow&gt;</a> so it can be indexed and queried using the optimized query operators from <a href="#">C1.LiveLinq.IndexedQueryExtensions</a> .
<a href="#">AsIndexed&lt;TRow&gt;(DataTable)</a>	Wraps a <b>System.Data.DataTable</b> in an <a href="#">IndexedDataTable&lt;TRow&gt;</a> so it can be indexed and queried using the optimized query operators from <a href="#">C1.LiveLinq.IndexedQueryExtensions</a> .
<a href="#">AsIndexed&lt;TRow&gt;(TypedTableBase&lt;TRow&gt;)</a>	Wraps a typed ADO.NET data table in an <a href="#">IndexedDataTable&lt;TRow&gt;</a> so it can be indexed and queried using the optimized query operators from <a href="#">C1.LiveLinq.IndexedQueryExtensions</a> .

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)  
[AdoNetExtensions Members](#)

## AsIndexed(DataTable) Method

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) > [AsIndexed Method](#) : AsIndexed(DataTable) Method

A **System.Data.DataTable** to represent as an [IndexedDataTable<DataRow>](#).

Wraps a **System.Data.DataTable** in an [IndexedDataTable<DataRow>](#) so it can be indexed and queried using the optimized query operators from [C1.LiveLinq.IndexedQueryExtensions](#).

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function AsIndexed( _     ByVal table As System.Data.DataTable _ ) As IndexedDataTable(Of DataRow)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static IndexedDataTable&lt;DataRow&gt; AsIndexed(     System.Data.DataTable table )</pre>	

## Parameters

*table*

A **System.Data.DataTable** to represent as an [IndexedDataTable<DataRow>](#).

## Return Value

An [IndexedDataTable<DataRow>](#) that contains the same rows as *table* and enables indexing of its rows.

## Remarks

Use this method to index ADO.NET data tables and query them using the query operators optimized with indexing.

Elements of the source data table aren't duplicated or copied to a new collection. This method just wraps the original data table in an [IndexedDataTable<DataRow>](#).

**Note:** The [IndexedDataTable<DataRow>](#) wrapper is owned by the original **System.Data.DataTable** object (in fact, it is stored in its **ExtendedProperties**). So, if you create a wrapper for the same data table several times, it will be the same object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)  
[AdoNetExtensions Members](#)  
[Overload List](#)

AsIndexed<TRow>(DataTable) Method

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) > [AsIndexed Method](#) :

AsIndexed<TRow>(DataTable) Method

The type of the rows in the *table*.

A **System.Data.DataTable** to represent as an [IndexedDataTable<TRow>](#).

Wraps a **System.Data.DataTable** in an [IndexedDataTable<TRow>](#) so it can be indexed and queried using the optimized query operators from [C1.LiveLinq.IndexedQueryExtensions](#).

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function AsIndexed(Of TRow As System.Data.DataRow)( _  
    ByVal table As System.Data.DataTable _  
) As IndexedDataTable(Of TRow)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static IndexedDataTable<TRow> AsIndexed<TRow>(  
    System.Data.DataTable table  
)  
where TRow: System.Data.DataRow
```

### Parameters

*table*

A **System.Data.DataTable** to represent as an [IndexedDataTable<TRow>](#).

## Type Parameters

*TRow*

The type of the rows in the *table*.

## Return Value

An [IndexedDataTable<TRow>](#) that contains the same rows as *table* and enables indexing of its rows.

## Remarks

Use this method to index ADO.NET data tables and query them using the query operators optimized with indexing.

Elements of the source data table aren't duplicated or copied to a new collection. This method just wraps the original data table in an [IndexedDataTable<TRow>](#).

**Note:** The [IndexedDataTable<TRow>](#) wrapper is owned by the original **System.Data.DataTable** object (in fact, it is stored in its **ExtendedProperties**). So, if you create a wrapper for the same data table several times, it will be the same object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)  
[AdoNetExtensions Members](#)  
[Overload List](#)

[AsIndexed<TRow>\(TypedTableBase<TRow>\) Method](#)

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) > [AsIndexed Method](#) :

[AsIndexed<TRow>\(TypedTableBase<TRow>\) Method](#)

The type of the rows in the *table*.

A typed data table to represent as an [IndexedDataTable<TRow>](#).

Wraps a typed ADO.NET data table in an [IndexedDataTable<TRow>](#) so it can be indexed and queried using the optimized query operators from [C1.LiveLinq.IndexedQueryExtensions](#).

## Syntax



## Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function AsIndexed(Of TRow As System.Data.DataRow)( _  
    ByVal table As System.Data.TypedTableBase(Of TRow) _  
) As IndexedDataTable(Of TRow)
```

## C#

```
[System.Runtime.CompilerServices.Extension()]  
public static IndexedDataTable<TRow> AsIndexed<TRow>(  
    System.Data.TypedTableBase<TRow> table  
)  
where TRow: System.Data.DataRow
```

## Parameters

*table*

A typed data table to represent as an [IndexedDataTable<TRow>](#).

## Type Parameters

*TRow*

The type of the rows in the *table*.

## Return Value

An [IndexedDataTable<TRow>](#) that contains the same rows as *table* and enables indexing of its rows.

## Remarks

Use this method to index typed data tables and query them using the query operators optimized with indexing.

Elements of the source data table aren't duplicated or copied to a new collection. This method just wraps the original data table in an [IndexedDataTable<TRow>](#).

**Note:** The [IndexedDataTable<TRow>](#) wrapper is owned by the original **System.Data.DataTable** object (in fact, it is stored in its **ExtendedProperties**). So, if you create a wrapper for the same data table several times, it will be the same object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)  
[AdoNetExtensions Members](#)  
[Overload List](#)

### AsLive Method

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) : AsLive Method

Creates a view based on the specified ADO.NET **System.Data.DataTable**.

## Overload List

Overload	Description
<a href="#">AsLive(DataTable)</a>	Creates a view based on the specified ADO.NET <b>System.Data.DataTable</b> .
<a href="#">AsLive&lt;TRow&gt;(DataTable)</a>	Creates a view based on the specified ADO.NET <b>System.Data.DataTable</b> .
<a href="#">AsLive&lt;TRow&gt;(TypedTableBase&lt;TRow&gt;)</a>	Creates a view based on the specified typed data table.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)  
[AdoNetExtensions Members](#)

### AsLive(DataTable) Method

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) > [AsLive Method](#) : AsLive(DataTable) Method

The **System.Data.DataTable** to expose as a view.

Creates a view based on the specified ADO.NET **System.Data.DataTable**.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function AsLive( _
    ByVal table As System.Data.DataTable _
) As View(Of DataRow)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<DataRow> AsLive(
    System.Data.DataTable table
)
```

### Parameters

*table*

The **System.Data.DataTable** to expose as a view.

### Return Value

A view that contains the same elements as *table*.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)  
[AdoNetExtensions Members](#)  
[Overload List](#)

AsLive<TRow>(DataTable) Method

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) > [AsLive Method](#) : AsLive<TRow>(DataTable) Method

The type of the rows in the *table*.

The typed data table to expose as a view.

Creates a view based on the specified ADO.NET **System.Data.DataTable**.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function AsLive(Of TRow As System.Data.DataRow)( _
    ByVal table As System.Data.DataTable _
) As View(Of TRow)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<TRow> AsLive<TRow>(
    System.Data.DataTable table
)
where TRow: System.Data.DataRow
```

### Parameters

*table*

The typed data table to expose as a view.

### Type Parameters

*TRow*

The type of the rows in the *table*.

### Return Value

A view that contains the same elements as *table*.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ADO.NET Extensions Class](#)  
[ADO.NET Extensions Members](#)  
[Overload List](#)

## AsLive<TRow>(TypedTableBase<TRow>) Method

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) > [AsLive Method](#) :

AsLive<TRow>(TypedTableBase<TRow>) Method

The type of the rows in the *table*.

The typed data table to expose as a view.

Creates a view based on the specified typed data table.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function AsLive(Of TRow As System.Data.DataRow)( _
    ByVal table As System.Data.TypedTableBase(Of TRow) _
) As View(Of TRow)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<TRow> AsLive<TRow>(
    System.Data.TypedTableBase<TRow> table
)
where TRow: System.Data.DataRow
```

## Parameters

*table*

The typed data table to expose as a view.

## Type Parameters

*TRow*

The type of the rows in the *table*.

## Return Value

A view that contains the same elements as *table*.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)  
[AdoNetExtensions Members](#)  
[Overload List](#)

### BeginUpdate Method

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) : BeginUpdate Method

The **System.Data.DataSet** to start massive changes for.

Indicates that massive changes are being made in code, until [EndUpdate](#), to data tables in this **System.Data.DataSet**.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Sub BeginUpdate( _     ByVal dataSet As System.Data.DataSet _ )</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static void BeginUpdate(     System.Data.DataSet dataSet )</pre>	

### Parameters

*dataSet*

The **System.Data.DataSet** to start massive changes for.

## Remarks

This extension method calls [BeginUpdate](#) for all data tables of this **System.Data.DataSet**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)

[AdoNetExtensions Members](#)

### EndUpdate Method

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) : EndUpdate Method

The **System.Data.DataSet** where massive changes have ended.

Indicates the end of massive changes to data tables in this **System.Data.DataSet** started with [BeginUpdate](#).

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Shared Sub EndUpdate( _  
    ByVal dataSet As System.Data.DataSet _  
)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static void EndUpdate(  
    System.Data.DataSet dataSet  
)
```

### Parameters

*dataSet*

The **System.Data.DataSet** where massive changes have ended.

## Remarks

This extension method calls [IndexedDataTable<TRow>.EndUpdate](#) for all data tables of this **System.Data.DataSet**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)

[AdoNetExtensions Members](#)

### IndexedField Method

[C1.LiveLinq.Adonet Namespace](#) > [AdoNetExtensions Class](#) : IndexedField Method

A hint to create and use an index on the specified data column. The hint has default action.

## Overload List

Overload	Description
<a href="#">IndexedField&lt;T&gt;(DataRow,Int32)</a>	A hint to create and use an index on the specified data column. The hint has default action.
<a href="#">IndexedField&lt;T&gt;(DataRow,DataColumn)</a>	A hint to create and use an index on the specified data column. The hint has default action.
<a href="#">IndexedField&lt;T&gt;(DataRow,String)</a>	A hint to create and use an index on the specified data column. The hint has default action.
<a href="#">IndexedField&lt;T&gt;(DataRow,Int32,IndexingHintAction)</a>	A hint to create and use an index on the specified data column. The hint has specified action.
<a href="#">IndexedField&lt;T&gt;(DataRow,DataColumn,IndexingHintAction)</a>	A hint to create and use an index on the specified data column. The hint has specified action.
<a href="#">IndexedField&lt;T&gt;(DataRow,String,IndexingHintAction)</a>	A hint to create and use an index on the specified data column. The hint has specified action.

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)  
[AdoNetExtensions Members](#)

IndexedField<T>(DataRow,Int32) Method

[Example](#)

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) > [IndexedField Method](#) :  
IndexedField<T>(DataRow,Int32) Method

The type of the data column

The data row.

The zero-based ordinal position of the column.

A hint to create and use an index on the specified data column. The hint has default action.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function IndexedField(Of T)( _     ByVal row As System.Data.DataRow, _     ByVal columnIndex As System.Integer _ ) As T</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static T IndexedField&lt;T&gt;(      System.Data.DataRow row,     System.int columnIndex )</pre>	

### Parameters

*row*

The data row.

*columnIndex*

The zero-based ordinal position of the column.

## Type Parameters

*T*

The type of the data column

## Return Value

The data column value.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on that column, if possible. When the query is executed, the hint method **IndexedField** is replaced with the standard LINQ to DataSet extension method **Field**. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```
var query = from c in customersTable
             where c.IndexedField<string>(0) == "ALFKI"
             select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)

[AdoNetExtensions Members](#)

[Overload List](#)

IndexedField<T>(DataRow, DataColumn) Method

[Example](#)

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) > [IndexedField Method](#) :

IndexedField<T>(DataRow, DataColumn) Method

The type of the data column

The data row.

The data column.

A hint to create and use an index on the specified data column. The hint has default action.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function IndexedField(Of T)( _     ByVal row As System.Data.DataRow, _     ByVal column As System.Data.DataColumn _ ) As T</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static T IndexedField&lt;T&gt;(     System.Data.DataRow row,     System.Data.DataColumn column )</pre>	

### Parameters

*row*

The data row.

*column*

The data column.

### Type Parameters

*T*

The type of the data column

### Return Value

The data column value.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on that column, if possible. When the query is executed, the hint

method **IndexedField** is replaced with the standard LINQ to DataSet extension method **Field**. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```
var query = from c in customersTable
             where c.IndexedField<string>(customerColumn) == "ALFKI"
             select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)  
[AdoNetExtensions Members](#)  
[Overload List](#)

IndexedField<T>(DataRow,String) Method

[Example](#)

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) > [IndexedField Method](#) :  
IndexedField<T>(DataRow,String) Method

The type of the data column

The data row.

The name of the column.

A hint to create and use an index on the specified data column. The hint has default action.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function IndexedField(Of T)( _
    ByVal row As System.Data.DataRow, _
    ByVal coLumnName As System.String _
) As T
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static T IndexedField<T>(   
    System.Data.DataRow row,  
    System.string columnName  
)
```

## Parameters

*row*

The data row.

*columnName*

The name of the column.

## Type Parameters

*T*

The type of the data column

## Return Value

The data column value.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on that column, if possible. When the query is executed, the hint method **IndexedField** is replaced with the standard LINQ to DataSet extension method **Field**. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```
var query = from c in customersTable  
             where c.IndexedField<string>("CustomerID") == "ALFKI"  
             select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[AdoNetExtensions Class](#)  
[AdoNetExtensions Members](#)  
[Overload List](#)

IndexedField<T>(DataRow,Int32,IndexingHintAction) Method

[Example](#)

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) > [IndexedField Method](#) :

IndexedField<T>(DataRow,Int32,IndexingHintAction) Method

The type of the data column

The data row.

The zero-based ordinal position of the column.

The action specified by the hint.

A hint to create and use an index on the specified data column. The hint has specified action.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function IndexedField(Of T)( _  
    ByVal row As System.Data.DataRow, _  
    ByVal columnIndex As System.Integer, _  
    ByVal action As IndexingHintAction _  
) As T
```

C#

```
[System.Runtime.CompilerServices.Extension()  
public static T IndexedField<T>(   
    System.Data.DataRow row,  
    System.int columnIndex,  
    IndexingHintAction action  
)
```

## Parameters

*row*

The data row.

*columnIndex*

The zero-based ordinal position of the column.

*action*

The action specified by the hint.

## Type Parameters

*T*

The type of the data column

## Return Value

The data column value.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on that column, if possible. When the query is executed, the hint method **IndexedField** is replaced with the standard LINQ to DataSet extension method **Field**. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```
var query = from c in customersTable
             where c.IndexedField<string>(0,
IndexingHintAction.Mandatory) == "ALFKI"
             select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[AdoNetExtensions Class](#)

[AdoNetExtensions Members](#)

[Overload List](#)

IndexedField<T>(DataRow, DataColumn, IndexingHintAction) Method

[Example](#)

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) > [IndexedField Method](#) :

IndexedField<T>(DataRow, DataColumn, IndexingHintAction) Method

The type of the data column

The data row.

The data column.

The action specified by the hint.

A hint to create and use an index on the specified data column. The hint has specified action.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function IndexedField(Of T)( _  
    ByVal row As System.Data.DataRow, _  
    ByVal column As System.Data.DataColumn, _  
    ByVal action As IndexingHintAction _  
) As T
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static T IndexedField<T>(   
    System.Data.DataRow row,  
    System.Data.DataColumn column,  
    IndexingHintAction action  
)
```

## Parameters

*row*

The data row.

*column*

The data column.

*action*

The action specified by the hint.

## Type Parameters

*T*



The type of the data column

## Return Value

The data column value.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on that column, if possible. When the query is executed, the hint method **IndexedField** is replaced with the standard LINQ to DataSet extension method **Field**. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```
var query =  
    from c in customersTable  
    where c.IndexedField<string>(customerColumn,  
        IndexingHintAction.Mandatory) == "ALFKI"  
    select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[AdoNetExtensions Class](#)

[AdoNetExtensions Members](#)

[Overload List](#)

IndexedField<T>(DataRow,String,IndexingHintAction) Method

[Example](#)

[C1.LiveLinq.AdoNet Namespace](#) > [AdoNetExtensions Class](#) > [IndexedField Method](#) :

IndexedField<T>(DataRow,String,IndexingHintAction) Method

The type of the data column

The data row.

The name of the column.

The action specified by the hint.

A hint to create and use an index on the specified data column. The hint has specified action.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function IndexedField(Of T)( _     ByVal row As System.Data.DataRow, _     ByVal columnName As System.String, _     ByVal action As IndexingHintAction _ ) As T</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static T IndexedField&lt;T&gt;(     System.Data.DataRow row,     System.string columnName,     IndexingHintAction action )</pre>	

### Parameters

*row*

The data row.

*columnName*

The name of the column.

*action*

The action specified by the hint.

### Type Parameters

*T*

The type of the data column

### Return Value

The data column value.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on that column, if possible. When the query is executed, the hint method **IndexedField** is replaced with the standard LINQ to DataSet extension method **Field**. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```
var query =  
    from c in customersTable  
    where c.IndexedField<string>("CustomerID",  
        IndexingHintAction.Mandatory) == "ALFKI"  
    select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AdoNetExtensions Class](#)  
[AdoNetExtensions Members](#)  
[Overload List](#)

## IndexedDataTable<TRow>

[C1.LiveLinq.AdoNet Namespace](#) : IndexedDataTable<TRow> Class

The type of the rows of the **System.Data.DataTable**. It is a class derived from **System.Data.DataRow**

A wrapper for the standard ADO.NET **System.Data.DataTable** class allowing to use ADO.NET data sources in LiveLinq indexing and live views.

## Object Model

IndexedDataTable<TRow>

## Syntax

Visual Basic (Declaration)

```

<System.Diagnostics.DebuggerDisplayAttribute(Value="\{Count = {Count} \}",
    Name="",
    Type="",
    Target=,
    TargetTypeName="")>
<System.Reflection.DefaultMemberAttribute("Item")>
Public Class IndexedDataTable(Of TRow As System.Data.DataRow)
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of TRow),
    C1.LiveLinq.IObservableSource(Of TRow)

```

C#

```

[System.Diagnostics.DebuggerDisplay(Value="\{Count = {Count} \}",
    Name="",
    Type="",
    Target=,
    TargetTypeName="")]
[System.Reflection.DefaultMember("Item")]
public class IndexedDataTable<TRow> :
    C1.LiveLinq.Indexing.IIndexedSource<TRow>,
    C1.LiveLinq.IObservableSource<TRow>
where TRow: System.Data.DataRow

```

## Type Parameters

*TRow*

The type of the rows of the **System.Data.DataTable**. It is a class derived from **System.Data.DataRow**

## Remarks

**Note:** The **IndexedDataTable** wrapper is owned by the original **System.Data.DataTable** object (in fact, it is stored in its **ExtendedProperties**). So, if you create a wrapper for the same data table several times, it will be the same object.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Adonet.IndexedDataTable<TRow>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedDataTable<TRow> Members](#)

[C1.LiveLinq.AdoNet Namespace](#)

### Overview

[C1.LiveLinq.AdoNet Namespace](#) : [IndexedDataTable<TRow> Class](#)

The type of the rows of the **System.Data.DataTable**. It is a class derived from **System.Data.DataRow**

A wrapper for the standard ADO.NET **System.Data.DataTable** class allowing to use ADO.NET data sources in LiveLinq indexing and live views.

## Object Model

[IndexedDataTable<TRow>](#)

### Syntax

Visual Basic (Declaration)

```
<System.Diagnostics.DebuggerDisplayAttribute(Value="{Count = {Count} \}",  
    Name="",  
    Type="",  
    Target=,  
    TargetTypeName="")>  
<System.Reflection.DefaultMemberAttribute("Item")>  
Public Class IndexedDataTable(Of TRow As System.Data.DataRow)  
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of TRow),  
    C1.LiveLinq.IObservableSource(Of TRow)
```

C#

```
[System.Diagnostics.DebuggerDisplay(Value="{Count = {Count} \}",  
    Name="",  
    Type="",  
    Target=,  
    TargetTypeName="")]  
[System.Reflection.DefaultMember("Item")]  
public class IndexedDataTable<TRow> :  
    C1.LiveLinq.Indexing.IIndexedSource<TRow>,
```

```
C1.LiveLinq.IObservableSource<TRow>
```

```
where TRow: System.Data.DataRow
```

## Type Parameters

*TRow*

The type of the rows of the **System.Data.DataTable**. It is a class derived from **System.Data.DataRow**

## Remarks

**Note:** The **IndexedDataTable** wrapper is owned by the original **System.Data.DataTable** object (in fact, it is stored in its **ExtendedProperties**). So, if you create a wrapper for the same data table several times, it will be the same object.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.AdoNet.IndexedDataTable<TRow>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedDataTable<TRow> Members](#)

[C1.LiveLinq.AdoNet Namespace](#)


## Members




[Properties](#) [Methods](#)

[C1.LiveLinq.AdoNet Namespace](#) : [IndexedDataTable<TRow>](#) Class

The following tables list the members exposed by [IndexedDataTable<TRow>](#).




## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the number of rows in the data table.

	<a href="#">Indexes</a>	The collection of indexes for this <a href="#">IndexedDataTable&lt;TRow&gt;</a>
	<a href="#">Item</a>	Gets the row at the specified ordinal position.
	<a href="#">Table</a>	Gets the ADO.NET <b>System.Data.DataTable</b> object represented by this <a href="#">IndexedDataTable&lt;TRow&gt;</a> .

[Top](#)

## Public Methods

	Name	Description
	<a href="#">BeginUpdate</a>	Suspends notifications while massive changes are being made to a data table.
	<a href="#">EndUpdate</a>	Ends notification suspension started with <a href="#">BeginUpdate</a> .
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">IndexedDataTable&lt;TRow&gt;</a> .

[Top](#)

## See Also

### Reference


[IndexedDataTable<TRow> Class](#)  
[C1.LiveLinq.AdoNet Namespace](#)



## Methods

[C1.LiveLinq.AdoNet Namespace](#) : [IndexedDataTable<TRow> Class](#)

For a list of all members of this type, see [IndexedDataTable<TRow> members](#).

## Public Methods

	Name	Description
	<a href="#">BeginUpdate</a>	Suspends notifications while massive changes are being made to a data table.

	<a href="#">EndUpdate</a>	Ends notification suspension started with <a href="#">BeginUpdate</a> .
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">IndexedDataTable&lt;TRow&gt;</a> .

[Top](#)

## See Also

### Reference

[IndexedDataTable<TRow> Class](#)

[C1.LiveLinq.AdoNet Namespace](#)

### BeginUpdate Method

[C1.LiveLinq.AdoNet Namespace](#) > [IndexedDataTable<TRow> Class](#) : [BeginUpdate Method](#)

Suspends notifications while massive changes are being made to a data table.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> <a href="#">BeginUpdate()</a>	
C#	
<b>public void</b> <a href="#">BeginUpdate()</a>	

## Remarks

This method must be followed by [EndUpdate](#).

Use this method when you already have indexes built over a data table or live views based on that data table, and you need to re-populate it or perform other massive changes to rows of that data table. Without this method, every single change you make causes LiveLinq to perform necessary operations for maintaining your indexes and live views dependent on this data table. In case of massive changes, this can be slower than to wait until the massive changes are done and rebuild the indexes and live views.

Between **BeginUpdate** and [EndUpdate](#) calls, indexes, live views, bound controls and other change notification listeners are not updated, they don't receive change notifications.

When [EndUpdate](#) is called, a [SourceChangeType.Modify](#) or [SourceChangeType.Reset](#) notification is sent, depending on whether the change affected a single row or multiple rows of the data table. Even when you change a single row, it may make sense to enclose your changes in **BeginUpdate/EndUpdate** if you change multiple fields in the row. In that case a



[SourceChangeType.Modify](#) notification is sent. If more than one row was changed, a [SourceChangeType.Reset](#) notification is sent, meaning all indexes, live views and other collections dependent on this data table must be rebuilt from scratch.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedDataTable<TRow> Class](#)  
[IndexedDataTable<TRow> Members](#)

### EndUpdate Method

[C1.LiveLinq.Adonet Namespace](#) > [IndexedDataTable<TRow> Class](#) : EndUpdate Method

Ends notification suspension started with [BeginUpdate](#).

## Syntax

Visual Basic (Declaration)	
<code>Public Sub EndUpdate()</code>	
C#	
<code>public void EndUpdate()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedDataTable<TRow> Class](#)  
[IndexedDataTable<TRow> Members](#)

### GetEnumerator Method

[C1.LiveLinq.Adonet Namespace](#) > [IndexedDataTable<TRow> Class](#) : GetEnumerator Method

Returns an enumerator that iterates through the [IndexedDataTable<TRow>](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Function GetEnumerator() As System.Collections.Generic.IEnumerator(Of TRow)</pre>	
C#	
<pre>public System.Collections.Generic.IEnumerator&lt;TRow&gt; GetEnumerator()</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedDataTable<TRow> Class](#)





[IndexedDataTable<TRow> Members](#)

## Properties

[C1.LiveLinq.AdoNet Namespace](#) : [IndexedDataTable<TRow> Class](#)

For a list of all members of this type, see [IndexedDataTable<TRow> members](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the number of rows in the data table.
	<a href="#">Indexes</a>	The collection of indexes for this <a href="#">IndexedDataTable&lt;TRow&gt;</a>
	<a href="#">Item</a>	Gets the row at the specified ordinal position.
	<a href="#">Table</a>	Gets the ADO.NET <b>System.Data.DataTable</b> object represented by this <a href="#">IndexedDataTable&lt;TRow&gt;</a> .

[Top](#)

## See Also

### Reference

[IndexedDataTable<TRow> Class](#)

[C1.LiveLinq.AdoNet Namespace](#)

### Count Property

[C1.LiveLinq.AdoNet Namespace](#) > [IndexedDataTable<TRow> Class](#) : Count Property

Gets the number of rows in the data table.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> Count <b>As</b> System.Integer	
C#	
<b>public</b> System.int Count { <b>get</b> ;}	

## Remarks

This property returns the same number of rows as the **Count** property of the **System.Data.DataTable** represented by this [IndexedDataTable<TRow>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedDataTable<TRow> Class](#)

[IndexedDataTable<TRow> Members](#)

### Indexes Property

[C1.LiveLinq.AdoNet Namespace](#) > [IndexedDataTable<TRow> Class](#) : Indexes Property

The collection of indexes for this [IndexedDataTable<TRow>](#)

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Indexes As IndexCollection(Of TRow)</code>	
C#	
<code>public IndexCollection&lt;TRow&gt; Indexes {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedDataTable<TRow> Class](#)

[IndexedDataTable<TRow> Members](#)

### Item Property

[C1.LiveLinq.AdoNet Namespace](#) > [IndexedDataTable<TRow> Class](#) : Item Property

The zero-based ordinal position of the row to return.

Gets the row at the specified ordinal position.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Default Property Item( _     ByVal ordinal As System.Integer _ ) As TRow</code>	
C#	
<code>public TRow this[     System.int ordinal ]; {get;}</code>	

### Parameters

*ordinal*

The zero-based ordinal position of the row to return.

## Property Value

The specified row.

## Remarks

This property returns the same row as **Rows[ordinal]** of the **System.Data.DataTable** represented by this [IndexedDataTable<TRow>](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedDataTable<TRow> Class](#)

[IndexedDataTable<TRow> Members](#)

### Table Property

[C1.LiveLinq.Adonet Namespace](#) > [IndexedDataTable<TRow> Class](#) : Table Property

Gets the ADO.NET **System.Data.DataTable** object represented by this [IndexedDataTable<TRow>](#).

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> Table <b>As</b> System.Data.DataTable	
C#	
<b>public</b> System.Data.DataTable Table { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2



## See Also

### Reference

# C1.LiveLinq.Collections Namespace

## Overview

### Classes

	Class	Description
	<a href="#">IndexableObject</a>	Base class for collection element classes.
	<a href="#">IndexedCollection&lt;T&gt;</a>	Data collection class recommended for use in LiveLinq to Objects.

### See Also

#### Reference

[C1.LiveLinq.4 Assembly](#)

## Classes

### IndexableObject

[C1.LiveLinq.Collections Namespace](#) : [IndexableObject Class](#)

Base class for collection element classes.

### Object Model

IndexableObject

### Syntax

Visual Basic (Declaration)	
<code>Public Class IndexableObject</code>	
C#	
<code>public class IndexableObject</code>	

### Remarks

Using [IndexedCollection<T>](#) and other collection classes|tag=LiveLinq to Objects:  
IndexedCollection(T) and other collection classes in LiveLinq to Objects, the element class **T** must implement the property change notification interface **System.ComponentModel.INotifyPropertyChanged**. The easiest way to satisfy this requirement is to derive that class from **IndexableObject**. Then you can use its method [OnPropertyChanged](#) to send the required property change notifications. For example, a **Customer** class can be defined like this:

```
public class Customer : IndexableObject
{
    ..... private string _name; public string Name { get
{ return _name; } set { _name = value; OnPropertyChanged("Name"); } }
    ..... }
```

## Inheritance Hierarchy

System.Object  
    **C1.LiveLinq.Collections.IndexableObject**  
        [C1.LiveLinq.LiveViews.ViewRow](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexableObject Members](#)  
[C1.LiveLinq.Collections Namespace](#)

## Overview

[C1.LiveLinq.Collections Namespace](#) : IndexableObject Class

Base class for collection element classes.

## Object Model

IndexableObject

## Syntax

Visual Basic (Declaration)	
Public Class IndexableObject	
C#	

```
public class IndexableObject
```

## Remarks

Using [IndexedCollection<T>](#) and other collection classes|tag=LiveLinq to Objects: IndexedCollection(T) and other collection classes in LiveLinq to Objects, the element class **T** must implement the property change notification interface

**System.ComponentModel.INotifyPropertyChanged**. The easiest way to satisfy this requirement is to derive that class from **IndexableObject**. Then you can use its method [OnPropertyChanged](#) to send the required property change notifications. For example, a **Customer** class can be defined like this:

```
public class Customer : IndexableObject
{
    ..... private string _name; public string Name { get
{ return _name; } set { _name = value; OnPropertyChanged("Name"); } }
    ..... }
```

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Collections.IndexableObject**

[C1.LiveLinq.LiveViews.ViewRow](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexableObject Members](#)

[C1.LiveLinq.Collections Namespace](#)


## Members

[Methods](#) [Events](#)

[C1.LiveLinq.Collections Namespace](#) : IndexableObject Class

The following tables list the members exposed by [IndexableObject](#).


## Public Constructors

	Name	Description
	<a href="#">IndexableObject Constructor</a>	Initializes a new instance of the <a href="#">IndexableObject</a> class.




[Top](#)

## Protected Methods

	Name	Description
	<a href="#">OnPropertyChanged</a>	Raises the <a href="#">PropertyChanged</a> event.

[Top](#)

## Public Events

	Name	Description
	<a href="#">PropertyChanged</a>	Occurs when a property value changes, after it has been changed.

[Top](#)

## See Also

### Reference

[IndexableObject Class](#)

[C1.LiveLinq.Collections Namespace](#)

## IndexableObject Constructor

[C1.LiveLinq.Collections Namespace](#) > [IndexableObject Class](#) : IndexableObject Constructor

Initializes a new instance of the [IndexableObject](#) class.

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public IndexableObject()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[IndexableObject Class](#)


[IndexableObject Members](#)

## Methods

[C1.LiveLinq.Collections Namespace](#) : [IndexableObject Class](#)

For a list of all members of this type, see [IndexableObject members](#).

# Protected Methods

	Name	Description
	<a href="#">OnPropertyChanged</a>	Raises the <a href="#">PropertyChanged</a> event.

[Top](#)

# See Also

## Reference

[IndexableObject Class](#)

[C1.LiveLinq.Collections Namespace](#)

## OnPropertyChanged Method

[C1.LiveLinq.Collections Namespace](#) > [IndexableObject Class](#) : [OnPropertyChanged Method](#)

The name of the property that is changed.

Raises the [PropertyChanged](#) event.

# Syntax

Visual Basic (Declaration)	
<pre>Protected Overridable Sub OnPropertyChanged( _     ByVal propertyName As System.String _ )</pre>	
C#	
<pre>protected virtual void OnPropertyChanged(     System.string propertyName</pre>	

)

## Parameters

*propertyName*

The name of the property that is changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexableObject Class](#)


[IndexableObject Members](#)

### Events

[C1.LiveLinq.Collections Namespace](#) : [IndexableObject Class](#)

For a list of all members of this type, see [IndexableObject members](#).

## Public Events

	Name	Description
	<a href="#">PropertyChanged</a>	Occurs when a property value changes, after it has been changed.

[Top](#)

## See Also

### Reference

[IndexableObject Class](#)

[C1.LiveLinq.Collections Namespace](#)

### PropertyChanged Event

[C1.LiveLinq.Collections Namespace](#) > [IndexableObject Class](#) : PropertyChanged Event

Occurs when a property value changes, after it has been changed.

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> PropertyChanged <b>As</b> System.ComponentModel.PropertyChangedEventHandler	
C#	
<b>public event</b> System.ComponentModel.PropertyChangedEventHandler PropertyChanged	

## Event Data

The event handler receives an argument of type `System.ComponentModel.PropertyChangedEventArgs` containing data related to this event. The following **PropertyChangedEventArgs** properties provide information specific to this event.

Property	Description
<b>PropertyName</b>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexableObject Class](#)

[IndexableObject Members](#)

## IndexedCollection<T>

[C1.LiveLinq.Collections Namespace](#) : IndexedCollection<T> Class

The type of the elements in the collection.

Data collection class recommended for use in LiveLinq to Objects.

## Object Model

**IndexedCollection<T>**

## Syntax

Visual Basic (Declaration)	
<pre>Public Class IndexedCollection(Of T)     Inherits System.Collections.ObjectModel.Collection(Of T)     Implements C1.LiveLinq.Indexing.IIndexedSource(Of T), C1.LiveLinq.IObservableSource(Of T)</pre>	
C#	
<pre>public class IndexedCollection&lt;T&gt; : System.Collections.ObjectModel.Collection&lt;T&gt;, C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;, C1.LiveLinq.IObservableSource&lt;T&gt;</pre>	

## Type Parameters

*T*

The type of the elements in the collection.

## Remarks

If you don't have preexisting collection classes and need to create your own, the best choice is to use the built-in LiveLinq collection class **IndexedCollection<T>**. It is specifically optimized for LiveLinq use.

Usually, the element class **T** must implement the property change notification interface **System.ComponentModel.INotifyPropertyChanged**. The easiest way to implement **System.ComponentModel.INotifyPropertyChanged** is to derive the element class from [IndexableObject](#).

In rare cases where implementing **System.ComponentModel.INotifyPropertyChanged** is impossible, a custom [C1.LiveLinq.Listeners.PropertyChangeListener<T>](#) can be provided instead.

For details, see Using the built-in collection class IndexedCollection(T) (LiveLinq to Objects).

For alternative options in LiveLinq to Objects, see LiveLinq to Objects: IndexedCollection(T) and other collection classes

The **IndexedCollection<T>** class is not needed in LiveLinq to DataSet and LiveLinq to XML, because in those cases LiveLinq works with collections that already exist in ADO.NET and XML.

## Inheritance Hierarchy

System.Object  
System.Collections.ObjectModel.Collection<T>  
**C1.LiveLinq.Collections.IndexedCollection<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedCollection<T> Members](#)  
[C1.LiveLinq.Collections Namespace](#)

### Overview

[C1.LiveLinq.Collections Namespace](#) : IndexedCollection<T> Class

The type of the elements in the collection.

Data collection class recommended for use in LiveLinq to Objects.

## Object Model

IndexedCollection<T>

## Syntax

Visual Basic (Declaration)

```
Public Class IndexedCollection(Of T)  
    Inherits System.Collections.ObjectModel.Collection(Of T)  
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T),  
    C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class IndexedCollection<T> :  
    System.Collections.ObjectModel.Collection<T>,  
    C1.LiveLinq.Indexing.IIndexedSource<T>, C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

The type of the elements in the collection.

## Remarks

If you don't have preexisting collection classes and need to create your own, the best choice is to use the built-in LiveLinq collection class **IndexedCollection<T>**. It is specifically optimized for LiveLinq use.

Usually, the element class **T** must implement the property change notification interface **System.ComponentModel.INotifyPropertyChanged**. The easiest way to implement **System.ComponentModel.INotifyPropertyChanged** is to derive the element class from [IndexableObject](#).

In rare cases where implementing **System.ComponentModel.INotifyPropertyChanged** is impossible, a custom [C1.LiveLinq.Listeners.PropertyChangeListener<T>](#) can be provided instead.

For details, see Using the built-in collection class IndexedCollection(T) (LiveLinq to Objects).

For alternative options in LiveLinq to Objects, see LiveLinq to Objects: IndexedCollection(T) and other collection classes

The **IndexedCollection<T>** class is not needed in LiveLinq to DataSet and LiveLinq to XML, because in those cases LiveLinq works with collections that already exist in ADO.NET and XML.

## Inheritance Hierarchy

System.Object  
  System.Collections.ObjectModel.Collection<T>  
    **C1.LiveLinq.Collections.IndexedCollection<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedCollection<T> Members](#)  
[C1.LiveLinq.Collections Namespace](#)


### Members

[Properties](#) [Methods](#) [Events](#)

C1.LiveLinq.Collections Namespace : IndexedCollection<T> Class





The following tables list the members exposed by [IndexedCollection<T>](#).

## Public Constructors

	Name	Description
	<a href="#">IndexedCollection&lt;T&gt; Constructor</a>	Overloaded.


[Top](#)

## Public Properties

	Name	Description
	<a href="#">Count</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
	<a href="#">CreateNew</a>	Gets or sets a delegate that is used to create new items. If it is null, a public parameterless constructor of type <b>T</b> is used.
	<a href="#">Indexes</a>	The collection of indexes for this <a href="#">IndexedCollection&lt;T&gt;</a>
	<a href="#">Item</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)



[Top](#)

## Protected Properties

	Name	Description
	<a href="#">Items</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)

[Top](#)

## Public Methods

	Name	Description
	<a href="#">Add</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
	<a href="#">AddRange</a>	Adds the elements of the specified collection to the end of the




		<a href="#">IndexedCollection&lt;T&gt;</a>
⇒	<a href="#">BeginUpdate</a>	Suspends notifications while massive changes are being made to the <a href="#">IndexedCollection&lt;T&gt;</a> .
⇒	<a href="#">Clear</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">Contains</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">CopyTo</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">EndUpdate</a>	Ends notification suspension started with <a href="#">BeginUpdate</a> .
⇒	<a href="#">GetEnumerator</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">IndexOf</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">Insert</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">Remove</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
⇒	<a href="#">RemoveAt</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)

[Top](#)


## Protected Methods

	Name	Description
🔒	<a href="#">ClearItems</a>	Overridden. Removes all elements from the <a href="#">IndexedCollection&lt;T&gt;</a>
🔒	<a href="#">InsertItem</a>	Overridden. Inserts an element into the <a href="#">IndexedCollection&lt;T&gt;</a> at the specified ordinal position.
🔒	<a href="#">OnChanged</a>	Raises the <a href="#">Changed</a> event.
🔒	<a href="#">RemoveItem</a>	Overridden. Removes the element of the <a href="#">IndexedCollection&lt;T&gt;</a> at the specified ordinal position.

	<a href="#">SetItem</a>	Overridden. Replaces the element at the specified ordinal position.
---	-------------------------	---

[Top](#)

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after the collection has changed.

[Top](#)

## See Also

### Reference

[IndexedCollection<T> Class](#)

[C1.LiveLinq.Collections Namespace](#)

## IndexedCollection<T> Constructor

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) : IndexedCollection<T> Constructor

## Overload List

Overload	Description
<a href="#">IndexedCollection&lt;T&gt; Constructor()</a>	Initializes a new instance of the <a href="#">IndexedCollection&lt;T&gt;</a> class.
<a href="#">IndexedCollection&lt;T&gt; Constructor(IList&lt;T&gt;)</a>	Initializes a new instance of the <a href="#">IndexedCollection&lt;T&gt;</a> class that contains elements copied from the specified list.
<a href="#">IndexedCollection&lt;T&gt; Constructor(IList&lt;T&gt;,PropertyChangeListener&lt;T&gt;)</a>	Initializes a new instance of the <a href="#">IndexedCollection&lt;T&gt;</a> class that contains elements copied from the specified list, and provides a custom <a href="#">C1.LiveLinq.Listeners.PropertyChangeListener&lt;T&gt;</a> .

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedCollection<T> Class](#)

[IndexedCollection<T> Members](#)

### IndexedCollection<T> Constructor()

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) > [IndexedCollection<T> Constructor](#) :  
IndexedCollection<T> Constructor()

Initializes a new instance of the [IndexedCollection<T>](#) class.

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public IndexedCollection&lt;T&gt;()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedCollection<T> Class](#)

[IndexedCollection<T> Members](#)

[Overload List](#)

### IndexedCollection<T> Constructor(IList<T>)

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) > [IndexedCollection<T> Constructor](#) :  
IndexedCollection<T> Constructor(IList<T>)

The collection whose elements are copied to the new instance.

Initializes a new instance of the [IndexedCollection<T>](#) class that contains elements copied from the specified list.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function New( _     ByVal list As System.Collections.Generic.IList(Of T) _ )</pre>	
C#	
<pre>public IndexedCollection&lt;T&gt;(     System.Collections.Generic.IList&lt;T&gt; list )</pre>	

### Parameters

*list*

The collection whose elements are copied to the new instance.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedCollection<T> Class](#)  
[IndexedCollection<T> Members](#)  
[Overload List](#)

### IndexedCollection<T> Constructor(IList<T>,PropertyChangeListener<T>)

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) > [IndexedCollection<T> Constructor](#) :  
IndexedCollection<T> Constructor(IList<T>,PropertyChangeListener<T>)

The collection whose elements are copied to the new instance.

The custom [C1.LiveLinq.Listeners.PropertyChangeListener<T>](#) to use with the collection.

Initializes a new instance of the [IndexedCollection<T>](#) class that contains elements copied from the specified list, and provides a custom [C1.LiveLinq.Listeners.PropertyChangeListener<T>](#).

## Syntax

Visual Basic (Declaration)

```
Public Function New( _  
    ByVal list As System.Collections.Generic.IList(Of T), _  
    ByVal itemPropertyChangeListener As PropertyChangeListener(Of T) _  
)
```

C#

```
public IndexedCollection<T>(  
    System.Collections.Generic.IList<T> list,  
    PropertyChangeListener<T> itemPropertyChangeListener  
)
```

### Parameters

*list*

The collection whose elements are copied to the new instance.

*itemPropertyChangeListener*

The custom [C1.LiveLinq.Listeners.PropertyChangeListener<T>](#) to use with the collection.

## Remarks

This constructor is intended for the rare cases where implementing **System.ComponentModel.INotifyPropertyChanged** in the class of the elements of the collection is impossible

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedCollection<T> Class](#)  
[IndexedCollection<T> Members](#)  
[Overload List](#)

## Methods

[C1.LiveLinq.Collections Namespace](#) : [IndexedCollection<T>](#) Class

For a list of all members of this type, see [IndexedCollection<T> members](#).






## Public Methods

	Name	Description
⇒	<a href="#">Add</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;T&gt;</a> )
⇒	<a href="#">AddRange</a>	Adds the elements of the specified collection to the end of the <a href="#">IndexedCollection&lt;T&gt;</a>
⇒	<a href="#">BeginUpdate</a>	Suspends notifications while massive changes are being made to the <a href="#">IndexedCollection&lt;T&gt;</a> .
⇒	<a href="#">Clear</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;T&gt;</a> )
⇒	<a href="#">Contains</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;T&gt;</a> )
⇒	<a href="#">CopyTo</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;T&gt;</a> )
⇒	<a href="#">EndUpdate</a>	Ends notification suspension started with <a href="#">BeginUpdate</a> .
⇒	<a href="#">GetEnumerator</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;T&gt;</a> )
⇒	<a href="#">IndexOf</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;T&gt;</a> )
⇒	<a href="#">Insert</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;T&gt;</a> )
⇒	<a href="#">Remove</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;T&gt;</a> )
⇒	<a href="#">RemoveAt</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;T&gt;</a> )

[Top](#)

## Protected Methods

	Name	Description
--	------	-------------

	<a href="#">ClearItems</a>	Overridden. Removes all elements from the <a href="#">IndexedCollection&lt;T&gt;</a>
	<a href="#">InsertItem</a>	Overridden. Inserts an element into the <a href="#">IndexedCollection&lt;T&gt;</a> at the specified ordinal position.
	<a href="#">OnChanged</a>	Raises the <a href="#">Changed</a> event.
	<a href="#">RemoveItem</a>	Overridden. Removes the element of the <a href="#">IndexedCollection&lt;T&gt;</a> at the specified ordinal position.
	<a href="#">SetItem</a>	Overridden. Replaces the element at the specified ordinal position.

[Top](#)

## See Also

### Reference

[IndexedCollection<T> Class](#)

[C1.LiveLinq.Collections Namespace](#)

### AddRange Method

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) : AddRange Method

The collection whose elements should be added to the end of the [IndexedCollection<T>](#)

Adds the elements of the specified collection to the end of the [IndexedCollection<T>](#)

## Syntax

Visual Basic (Declaration)

```
Public Sub AddRange( _
    ByVal items As System.Collections.Generic.IEnumerable(Of T) _
)
```

C#

```
public void AddRange(
    System.Collections.Generic.IEnumerable<T> items
)
```

### Parameters

*items*

The collection whose elements should be added to the end of the [IndexedCollection<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedCollection<T> Class](#)

[IndexedCollection<T> Members](#)

### BeginUpdate Method

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) : BeginUpdate Method

Suspends notifications while massive changes are being made to the [IndexedCollection<T>](#).

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> BeginUpdate()	
C#	
<b>public void</b> BeginUpdate()	

## Remarks

This method must be followed by [EndUpdate](#).

Use this method when you already have indexes built over the [IndexedCollection<T>](#) or live views based on it, and you need to re-populate this [IndexedCollection<T>](#) or perform other massive changes to items of this collection. Without this method, every single change you make causes LiveLinq to perform necessary operations for maintaining your indexes and live views dependent on this collection. In case of massive changes, this can be slower than to wait until the massive changes are done and rebuild the indexes and live views.

Between **BeginUpdate** and [EndUpdate](#) calls, indexes, live views, bound controls and other change notification listeners are not updated, they don't receive change notifications.

When [EndUpdate](#) is called, a [SourceChangeType.Modify](#) or [SourceChangeType.Reset](#) notification is sent, depending on whether the change affected a single item or multiple items of the collection. Even when you change a single item, it may make sense to enclose your



changes in **BeginUpdate/EndUpdate** if you change multiple properties of the item. In that case a [SourceChangeType.Modify](#) notification is sent. If more than one item was changed, a [SourceChangeType.Reset](#) notification is sent, meaning all indexes, live views and other collections dependent on this data table must be rebuilt from scratch.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedCollection<T> Class](#)

[IndexedCollection<T> Members](#)

### ClearItems Method

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) : ClearItems Method

Removes all elements from the [IndexedCollection<T>](#)

## Syntax

Visual Basic (Declaration)	
<b>Protected Overrides Sub</b> ClearItems()	
C#	
<b>protected override void</b> ClearItems()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedCollection<T> Class](#)

[IndexedCollection<T> Members](#)

## EndUpdate Method

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) : EndUpdate Method

Ends notification suspension started with [BeginUpdate](#).

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> EndUpdate()	
C#	
<b>public void</b> EndUpdate()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedCollection<T> Class](#)

[IndexedCollection<T> Members](#)

## InsertItem Method

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) : InsertItem Method

The zero-based ordinal at which the item must be inserted.

The object to insert.

Inserts an element into the [IndexedCollection<T>](#) at the specified ordinal position.

## Syntax

Visual Basic (Declaration)	
<b>Protected Overrides Sub</b> InsertItem( _ <b>ByVal</b> ordinal <b>As</b> System.Integer, _ <b>ByVal</b> item <b>As</b> T _ )	
C#	

```
protected override void InsertItem(
    System.int ordinal,
    T item
)
```

## Parameters

*ordinal*

The zero-based ordinal at which the item must be inserted.

*item*

The object to insert.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexedCollection<T> Class](#)

[IndexedCollection<T> Members](#)

## OnChanged Method

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) :

OnChanged(SourceChangeEventArgs<T>) Method

Event data.

Raises the [Changed](#) event.

## Syntax

Visual Basic (Declaration)

```
Protected Sub OnChanged( _
    ByVal e As SourceChangeEventArgs(Of T) _
)
```

C#

```
protected void OnChanged(
    SourceChangeEventArgs<T> e
```

)

## Parameters

*e*

Event data.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexedCollection<T> Class](#)

[IndexedCollection<T> Members](#)

## RemoveItem Method

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) : RemoveItem Method

The zero-based ordinal of the element to remove.

Removes the element of the [IndexedCollection<T>](#) at the specified ordinal position.

## Syntax

Visual Basic (Declaration)

```
Protected Overrides Sub RemoveItem( _  
    ByVal ordinal As System.Integer _  
)
```

C#

```
protected override void RemoveItem(  
    System.int ordinal  
)
```

## Parameters

*ordinal*

The zero-based ordinal of the element to remove.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedCollection<T> Class](#)  
[IndexedCollection<T> Members](#)

### SetItem Method

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) : SetItem Method

The zero-based ordinal of the element to replace.

The new value for the element at the specified ordinal. The value can be null for reference types.

Replaces the element at the specified ordinal position.

## Syntax

Visual Basic (Declaration)	
<pre>Protected Overrides Sub SetItem( _     ByVal ordinal As System.Integer, _     ByVal newItem As T _ )</pre>	
C#	
<pre>protected override void SetItem(     System.int ordinal,     T newItem )</pre>	

### Parameters

*ordinal*

The zero-based ordinal of the element to replace.

*newItem*

The new value for the element at the specified ordinal. The value can be null for reference types.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference





[IndexedCollection<T> Class](#)  
[IndexedCollection<T> Members](#)

## Properties

[C1.LiveLinq.Collections Namespace](#) : [IndexedCollection<T> Class](#)


For a list of all members of this type, see [IndexedCollection<T> members](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)
	<a href="#">CreateNew</a>	Gets or sets a delegate that is used to create new items. If it is null, a public parameterless constructor of type <b>T</b> is used.
	<a href="#">Indexes</a>	The collection of indexes for this <a href="#">IndexedCollection&lt;T&gt;</a>
	<a href="#">Item</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)

[Top](#)

## Protected Properties

	Name	Description
	<a href="#">Items</a>	(Inherited from System.Collections.ObjectModel.Collection<T>)

[Top](#)

## See Also

### Reference

[IndexedCollection<T> Class](#)  
[C1.LiveLinq.Collections Namespace](#)

## CreateNew Property

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) : CreateNew Property

Gets or sets a delegate that is used to create new items. If it is null, a public parameterless constructor of type **T** is used.

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> CreateNew <b>As</b> System.Func(Of T)	
C#	
<b>public</b> System.Func<T> CreateNew { <b>get</b> ; <b>set</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedCollection<T> Class](#)

[IndexedCollection<T> Members](#)

## Indexes Property

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) : Indexes Property

The collection of indexes for this [IndexedCollection<T>](#)

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> Indexes <b>As</b> IndexCollection(Of T)	
C#	
<b>public</b> IndexCollection<T> Indexes { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference


[IndexedCollection<T> Class](#)  
[IndexedCollection<T> Members](#)

### Events

[C1.LiveLinq.Collections Namespace](#) : [IndexedCollection<T> Class](#)

For a list of all members of this type, see [IndexedCollection<T> members](#).

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after the collection has changed.

[Top](#)

## See Also

### Reference

[IndexedCollection<T> Class](#)  
[C1.LiveLinq.Collections Namespace](#)

### Changed Event

[C1.LiveLinq.Collections Namespace](#) > [IndexedCollection<T> Class](#) : Changed Event

Occurs after the collection has changed.

## Syntax

Visual Basic (Declaration)	
<code>Public Event Changed As System.EventHandler(Of SourceChangeEventArgs(Of T))</code>	
C#	
<code>public event System.EventHandler&lt;SourceChangeEventArgs&lt;T&gt;&gt; Changed</code>	

## Event Data



The event handler receives an argument of type [SourceChangeEventArgs<T>](#) containing data related to this event. The following **SourceChangeEventArgs<T>** properties provide information specific to this event.

Property	Description
<a href="#">ChangeType</a>	Gets the type of change.
<a href="#">Item</a>	Gets the object that is being changed.
<a href="#">Ordinal</a>	Gets the ordinal position of the collection item that is being changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference




[IndexedCollection<T> Class](#)








[IndexedCollection<T> Members](#)

# C1.LiveLinq.Indexing Namespace


## Overview

### Classes

	Class	Description
	<a href="#">Index&lt;T&gt;</a>	Base class for the <a href="#">Index&lt;T,TKey&gt;</a> class.
	<a href="#">Index&lt;T,TKey&gt;</a>	Indexes a collection by an expression (typically, by a field), providing fast access to items having particular values (or range of values) of that expression.
	<a href="#">IndexCollection&lt;T&gt;</a>	Represents a collection of indexes attached to an indexed collection.

	<a href="#">IndexDefinition&lt;T&gt;</a>	Contains common part of the <a href="#">Index</a> and <a href="#">Subindex</a> classes.
	<a href="#">IndexingAlgorithm</a>	Defines the kind of an index, the algorithm used by that index. Currently, the RedBlackTree algorithm is always used.
	<a href="#">IndexingException</a>	Represents an exception that is thrown when errors are generated using LiveLinq components.
	<a href="#">ScannerCollection&lt;T&gt;</a>	Represents a collection of indexes or subindexes.
	<a href="#">Subindex&lt;T&gt;</a>	Base class for the <a href="#">Subindex&lt;T,TKey&gt;</a> class.
	<a href="#">Subindex&lt;T,TKey&gt;</a>	Defines a subindex, an index definition subordinate to another index definition, its parent.
	<a href="#">SubindexCollection&lt;T&gt;</a>	Represents a collection of subindexes attached to an <a href="#">IndexDefinition&lt;T&gt;</a> .

## Interfaces

	Interface	Description
	<a href="#">IIndexedSource&lt;T&gt;</a>	Represents an indexed collection.

## See Also

### Reference

[C1.LiveLinq.4 Assembly](#)

## Classes

### Index<T>

[C1.LiveLinq.Indexing Namespace](#) : [Index<T>](#) Class

The type of the elements of the collection to index.

Base class for the [Index<T,TKey>](#) class.

## Object Model

## Syntax

Visual Basic (Declaration)

```
Public MustInherit Class Index(Of T)
    Inherits IndexDefinition(Of T)
    Implements C1.LiveLinq.Indexing.Search.IIndexScanner(Of T)
```

C#

```
public abstract class Index<T> : IndexDefinition<T>,
    C1.LiveLinq.Indexing.Search.IIndexScanner<T>
```

## Type Parameters

*T*

The type of the elements of the collection to index.

## Remarks

You don't typically use the **Index<T>** class directly. It provides functionality of the [Index<T,TKey>](#) class that does not depend on the index key type. The base class **Index<T>** is needed only if the index key type is not known, usually in general-purpose code intended for reuse with different key types.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.IndexDefinition<T>](#)

**C1.LiveLinq.Indexing.Index<T>**

[C1.LiveLinq.Indexing.Index<T,TKey>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T> Members](#)  
[C1.LiveLinq.Indexing Namespace](#)  
[Index<T,TKey> Class](#)

## Overview

[C1.LiveLinq.Indexing Namespace](#) : [Index<T>](#) Class

The type of the elements of the collection to index.

Base class for the [Index<T,TKey>](#) class.

## Object Model

[Index<T>](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public MustInherit Class Index(Of T)     Inherits IndexDefinition(Of T)     Implements C1.LiveLinq.Indexing.Search.IIndexScanner(Of T)</pre>	
C#	
<pre>public abstract class Index&lt;T&gt; : IndexDefinition&lt;T&gt;,     C1.LiveLinq.Indexing.Search.IIndexScanner&lt;T&gt;</pre>	

## Type Parameters

*T*

The type of the elements of the collection to index.

## Remarks

You don't typically use the **Index<T>** class directly. It provides functionality of the [Index<T,TKey>](#) class that does not depend on the index key type. The base class **Index<T>** is needed only if the index key type is not known, usually in general-purpose code intended for reuse with different key types.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.IndexDefinition<T>](#)

**C1.LiveLinq.Indexing.Index<T>**

[C1.LiveLinq.Indexing.Index<T,TKey>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T> Members](#)  
[C1.LiveLinq.Indexing Namespace](#)  
[Index<T,TKey> Class](#)







## Members




[Properties](#) [Methods](#)

[C1.LiveLinq.Indexing Namespace](#) : [Index<T>](#) Class

The following tables list the members exposed by [Index<T>](#).






## Public Properties

	Name	Description
	<a href="#">Algorithm</a>	Gets the indexing algorithm used by the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">ItemCount</a>	Gets the number of elements in the indexed collection.
	<a href="#">KeyCount</a>	Gets the number of distinct key values in all items of this collection.
	<a href="#">KeyIsUnique</a>	Gets a value that indicates whether the key used in this index is a unique key for the collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeySelector</a>	Gets the expression used to obtain key value from an element of the indexed collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeyType</a>	Gets the type of the index key. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )

	<b>Locale</b>	Gets the locale information used to compare strings in the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<b>Root</b>	Gets the root index in an index/subindex hierarchy. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<b>Subindexes</b>	Gets the collection of subindexes added to this index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )

[Top](#)

## Public Methods

	Name	Description
	<b>ContainsKey</b>	Returns a value that indicates whether the indexed collection contains an item with the given key value.  Implements <a href="#">IIndexScanner(T).ContainsKey(object)</a>
	<b>Find</b>	Finds items with the specified key value.  Implements <a href="#">IIndexScanner(T).Find(object)</a>
	<b>FindBetween</b>	Finds items with key values in the interval between the specified values.  Implements <a href="#">IIndexScanner(T).FindBetween(object,bool,object,bool,Order)</a>
	<b>FindGreater</b>	Finds items with keys greater than the specified value.  Implements <a href="#">IIndexScanner(T).FindGreater(object,bool,Order)</a>
	<b>FindKeys</b>	Finds items containing any of the specified key values.

		Implements <a href="#">FindKeys(IEnumerable,Order)</a>
≡	<a href="#">FindLess</a>	<p>Finds items with keys less than the specified value.</p> <p>Implements <a href="#">IIndexScanner(T).FindLess(object,bool,Order)</a></p>
≡	<a href="#">FindStartingWith</a>	<p>Finds items with string key values starting with the specified string.</p> <p>Implements <a href="#">IIndexScanner(T).FindStartingWith(string,Func(string,bool),Order)</a></p>
≡	<a href="#">GroupJoin</a>	<p>Overloaded.</p> <p>Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.</p> <p>Implements <a href="#">IIndexScanner(T).GroupJoin</a></p>
≡	<a href="#">Join</a>	<p>Overloaded.</p> <p>Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.</p> <p>Implements <a href="#">IIndexScanner(T).Join</a></p>

[Top](#)

## See Also

### Reference

[Index<T> Class](#)

[C1.LiveLinq.Indexing Namespace](#)

[Index<T,TKey> Class](#)

## Methods

[C1.LiveLinq.Indexing Namespace](#) : [Index<T>](#) Class

For a list of all members of this type, see [Index<T> members](#).

## Public Methods

	Name	Description
⇒💎	<a href="#">ContainsKey</a>	Returns a value that indicates whether the indexed collection contains an item with the given key value.  Implements <a href="#">IIndexScanner(T).ContainsKey(object)</a>
⇒💎	<a href="#">Find</a>	Finds items with the specified key value.  Implements <a href="#">IIndexScanner(T).Find(object)</a>
⇒💎	<a href="#">FindBetween</a>	Finds items with key values in the interval between the specified values.  Implements <a href="#">IIndexScanner(T).FindBetween(object,bool,object,bool,Order)</a>
⇒💎	<a href="#">FindGreater</a>	Finds items with keys greater than the specified value.  Implements <a href="#">IIndexScanner(T).FindGreater(object,bool,Order)</a>
⇒💎	<a href="#">FindKeys</a>	Finds items containing any of the specified key values.  Implements <a href="#">FindKeys(IEnumerable,Order)</a>
⇒💎	<a href="#">FindLess</a>	Finds items with keys less than the specified value.  Implements <a href="#">IIndexScanner(T).FindLess(object,bool,Order)</a>



≡💎	<a href="#">FindStartingWith</a>	<p>Finds items with string key values starting with the specified string.</p> <p>Implements <a href="#">IIndexScanner(T).FindStartingWith(string,Func(string,bool),Order)</a></p>
≡💎	<a href="#">GroupJoin</a>	<p>Overloaded.</p> <p>Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.</p> <p>Implements <a href="#">IIndexScanner(T).GroupJoin</a></p>
≡💎	<a href="#">Join</a>	<p>Overloaded.</p> <p>Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.</p> <p>Implements <a href="#">IIndexScanner(T).Join</a></p>

[Top](#)

## See Also

### Reference

[Index<T> Class](#)

[C1.LiveLinq.Indexing Namespace](#)

[Index<T,TKey> Class](#)

### ContainsKey Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) : ContainsKey Method

The key value to search for.

Returns a value that indicates whether the indexed collection contains an item with the given key value.

Implements [IIndexScanner\(T\).ContainsKey\(object\)](#)

## Syntax

Visual Basic (Declaration)

```
Public Function ContainsKey( _  
    ByVal key As System.Object _  
) As System.Boolean
```

C#

```
public System.bool ContainsKey(  
    System.object key  
)
```

### Parameters

*key*

The key value to search for.

### Return Value

**true** if the indexed collection contains an element with the specified key value; otherwise, **false**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T> Class](#)

[Index<T> Members](#)

### Find Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) : Find Method

The key value to search for.

Finds items with the specified key value.

Implements [IIndexScanner\(T\).Find\(object\)](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Function Find( _     ByVal key As System.Object _ ) As IndexQuery(Of T)</pre>	
C#	
<pre>public IndexQuery&lt;T&gt; Find(     System.object key )</pre>	

## Parameters

*key*

The key value to search for.

## Return Value

An object enumerating items having the specified key value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T> Class](#)

[Index<T> Members](#)

## FindBetween Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) : FindBetween Method

Minimum key value to search for.

If **true**, the result includes items with the minimum key value.

Maximum key value to search for.

If **true**, the result includes items with the maximum key value.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with key values in the interval between the specified values.

Implements [IIndexScanner\(T\).FindBetween\(object,bool,object,bool,Order\)](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Function FindBetween( _     ByVal min As System.Object, _     ByVal minInclusive As System.Boolean, _     ByVal max As System.Object, _     ByVal maxInclusive As System.Boolean, _     ByVal order As Order _ ) As IndexQuery(Of T)</pre>	
C#	
<pre>public IndexQuery&lt;T&gt; FindBetween(     System.object min,     System.bool minInclusive,     System.object max,     System.bool maxInclusive,     Order order )</pre>	

### Parameters

*min*

Minimum key value to search for.

*minInclusive*

If **true**, the result includes items with the minimum key value.

*max*

Maximum key value to search for.

*maxInclusive*

If **true**, the result includes items with the maximum key value.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

### Return Value

An object enumerating all items with key values within the specified limits.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T> Class](#)

[Index<T> Members](#)

### FindGreater Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) : FindGreater Method

Minimum key value to search for.

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly greater than the specified value.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with keys greater than the specified value.

Implements [IIndexScanner\(T\).FindGreater\(object,bool,Order\)](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Function FindGreater( _     ByVal key As System.Object, _     ByVal inclusive As System.Boolean, _     ByVal order As Order _ ) As IndexQuery(Of T)</pre>	
C#	
<pre>public IndexQuery&lt;T&gt; FindGreater(     System.object key,     System.bool inclusive,</pre>	

```
Order order
)
```

## Parameters

*key*

Minimum key value to search for.

*inclusive*

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly greater than the specified value.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values are greater than the specified value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T> Class](#)

[Index<T> Members](#)

## FindKeys Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) : FindKeys Method

The key values to search for.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items containing any of the specified key values.

Implements [FindKeys\(IEnumerable,Order\)](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Function FindKeys( _     ByVal keys As System.Collections.IEnumerable, _     ByVal order As Order _ ) As IndexQuery(Of T)</pre>	
C#	
<pre>public IndexQuery&lt;T&gt; FindKeys(     System.Collections.IEnumerable keys,     Order order )</pre>	

## Parameters

*keys*

The key values to search for.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values belong to the specified key value collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T> Class](#)  
[Index<T> Members](#)

## FindLess Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) : FindLess Method

Maximum key value to search for.

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly less than the specified value.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with keys less than the specified value.

Implements [IIndexScanner\(T\).FindLess\(object,bool,Order\)](#)

## Syntax

### Visual Basic (Declaration)

```
Public Function FindLess( _  
    ByVal key As System.Object, _  
    ByVal inclusive As System.Boolean, _  
    ByVal order As Order _  
) As IndexQuery(Of T)
```

### C#

```
public IndexQuery<T> FindLess(  
    System.object key,  
    System.bool inclusive,  
    Order order  
)
```

## Parameters

*key*

Maximum key value to search for.

*inclusive*

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly less than the specified value.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values are less than the specified value.



## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T> Class](#)

[Index<T> Members](#)

### FindStartingWith Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) : FindStartingWith Method

The string to search for as the beginning of key value strings.

An optional condition that found items must satisfy.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with string key values starting with the specified string.

Implements [IIndexScanner\(T\).FindStartingWith\(string,Func\(string, bool\),Order\)](#)

## Syntax

Visual Basic (Declaration)

```
Public Function FindStartingWith( _  
    ByVal value As System.String, _  
    ByVal keyPredicate As System.Func(Of String,Boolean), _  
    ByVal order As Order _  
) As IndexQuery(Of T,String)
```

C#

```
public IndexQuery<T,string> FindStartingWith(  
    System.string value,  
    System.Func<string,bool> keyPredicate,  
    Order order  
)
```

### Parameters

*value*

The string to search for as the beginning of key value strings.

*keyPredicate*

An optional condition that found items must satisfy.

*order*

Optionally specifies the order of the key values to sort the result  
([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values are strings that have a beginning matching the specified string and satisfy the optional condition.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T> Class](#)

[Index<T> Members](#)

## GroupJoin Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) : GroupJoin Method

Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.

Implements [IIndexScanner\(T\).GroupJoin](#)

## Overload List

Overload	Description
<a href="#">GroupJoin&lt;T2,TResult&gt;(IIndexScanner&lt;T2&gt;,Func&lt;T,IEnumerable&lt;T2&gt;,TR esult&gt;)</a>	Correlates the items of this indexed collection with the items of another indexed collection and groups the

	<p>results by the item of this collection.</p> <p>Implements  <a href="#">IIndexScanner(T).GroupJoin</a></p>
<a href="#">GroupJoin&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,Object&gt;,Func&lt;IEnumerable&lt;T&gt;,T2,TResult&gt;)</a>	<p>Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of the second sequence.</p> <p>Implements  <a href="#">IIndexScanner(T).GroupJoin</a></p>
<a href="#">GroupJoin&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,Object&gt;,Func&lt;T,IEnumerable&lt;T2&gt;,TResult&gt;)</a>	<p>Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of this collection.</p> <p>Implements  <a href="#">IIndexScanner(T).GroupJoin</a></p>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T> Class](#)  
[Index<T> Members](#)

[GroupJoin<T2,TResult>\(IIndexScanner<T2>,Func<T,IEnumerable<T2>,TResult>\) Method](#)  
[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) > [GroupJoin Method](#) :  
[GroupJoin<T2,TResult>\(IIndexScanner<T2>,Func<T,IEnumerable<T2>,TResult>\) Method](#)

The type of the elements of the second collection.

The type of the result elements.

The second indexed collection to join to this collection.

A function to create a result element from an element from this collection and a collection of matching elements from the second collection.

Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.

Implements [IIndexScanner\(T\).GroupJoin](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function GroupJoin     (Of T2,TResult)( _     ByVal source As IIndexScanner(Of T2), _     ByVal resultSelector As System.Func(Of T,IEnumerable(Of T2),TResult) _ ) As System.Collections.Generic.IEnumerable(Of TResult)</pre>	
C#	
<pre>public System.Collections.Generic.IEnumerable&lt;TResult&gt; GroupJoin&lt;T2,TResult&gt;(     IIndexScanner&lt;T2&gt; source,     System.Func&lt;T,IEnumerable&lt;T2&gt;,TResult&gt; resultSelector )</pre>	

### Parameters

*source*

The second indexed collection to join to this collection.

*resultSelector*

A function to create a result element from an element from this collection and a collection of matching elements from the second collection.

## Type Parameters

*T2*

The type of the elements of the second collection.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to group pairs, where each pair consists of an item of this collection and the corresponding enumeration of the items of the second collection joined to it.

## Remarks

Matching of two elements is performed by matching their keys.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T> Class](#)

[Index<T> Members](#)

[Overload List](#)

`GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<IEnumerable<T>,T2,TResult> ) Method`

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) > [GroupJoin Method](#) :

`GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<IEnumerable<T>,T2,TResult>) Method`

The type of the elements of the second sequence.

The type of the result elements.

The second sequence to join to this collection.

A function to extract from an item of the second sequence the value to match against this collection's key value.

A function to create a result element from an element of the second sequence and the collection of matching elements from this collection.

Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of the second sequence.

Implements [IIndexScanner\(T\).GroupJoin](#)

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function GroupJoin
    (Of T2,TResult)( _
    ByVal source As System.Collections.Generic.IEnumerable(Of T2), _
    ByVal keySelector As System.Func(Of T2,Object), _
    ByVal resultSelector As System.Func(Of IEnumerable(Of T),T2,TResult) _
) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
public System.Collections.Generic.IEnumerable<TResult> GroupJoin<T2,TResult>(
    System.Collections.Generic.IEnumerable<T2> source,
    System.Func<T2,object> keySelector,
    System.Func<IEnumerable<T>,T2,TResult> resultSelector
)
```

## Parameters

*source*

The second sequence to join to this collection.

*keySelector*

A function to extract from an item of the second sequence the value to match against this collection's key value.

*resultSelector*

A function to create a result element from an element of the second sequence and the collection of matching elements from this collection.

## Type Parameters

*T2*

The type of the elements of the second sequence.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to group pairs, where each pair consists of an item of the second collection and the corresponding enumeration of the items of this collection joined to it.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T> Class](#)

[Index<T> Members](#)

[Overload List](#)

`GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<T,IEnumerable<T2>,TResult> ) Method`

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) > [GroupJoin Method](#) :

`GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<T,IEnumerable<T2>,TResult>) Method`

The type of the elements of the second sequence.

The type of the result elements.

The second sequence to join to this collection.

A function to extract from an item of the second sequence the value to match against this collection's key value.

A function to create a result element from an element from this collection and a collection of matching elements from the second sequence.

Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of this collection.

Implements [IIndexScanner\(T\).GroupJoin](#)

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function GroupJoin
    (Of T2,TResult)( _
    ByVal source As System.Collections.Generic.IEnumerable(Of T2), _
    ByVal keySelector As System.Func(Of T2,Object), _
    ByVal resultSelector As System.Func(Of T,IEnumerable(Of T2),TResult) _
) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
public System.Collections.Generic.IEnumerable<TResult> GroupJoin<T2,TResult>(
    System.Collections.Generic.IEnumerable<T2> source,
    System.Func<T2,object> keySelector,
    System.Func<T,IEnumerable<T2>,TResult> resultSelector
)
```

### Parameters

*source*

The second sequence to join to this collection.

*keySelector*

A function to extract from an item of the second sequence the value to match against this collection's key value.

*resultSelector*

A function to create a result element from an element from this collection and a collection of matching elements from the second sequence.

### Type Parameters

*T2*

The type of the elements of the second sequence.

*TResult*

The type of the result elements.



## Return Value

Enumeration of objects obtained by applying the result selector to group pairs, where each pair consists of an item of this collection and the corresponding enumeration of the items of the second sequence joined to it.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

- [Index<T> Class](#)
- [Index<T> Members](#)
- [Overload List](#)

## Join Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) : Join Method

Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.

Implements [IIndexScanner\(T\).Join](#)

## Overload List

Overload	Description
<a href="#">Join&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,Object&gt;,Func&lt;T,T2,TResult&gt;,JoinOperator)</a>	Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.  Implements

	<a href="#">IIndexScanner(T).Join</a>
<a href="#">Join&lt;T2,TResult&gt;(IIndexScanner&lt;T2&gt;,Func&lt;T,T2,TResult&gt;,JoinOperator)</a>	<p>Correlates the items of this indexed collection with the items of another indexed collection and returns the combined items with matching keys.</p> <p>Implements <a href="#">IIndexScanner(T).Join</a></p>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T> Class](#)

[Index<T> Members](#)

[Join<T2,TResult>\(IEnumerable<T2>,Func<T2,Object>,Func<T,T2,TResult>,JoinOperator\)](#)  
Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) > [Join Method](#) :

[Join<T2,TResult>\(IEnumerable<T2>,Func<T2,Object>,Func<T,T2,TResult>,JoinOperator\)](#) Method

The type of the elements of the second sequence.

The type of the result elements.

The second sequence to join to this collection.

A function to extract from a second sequence's item the value to match against this collection's key value.

A function to create a result element from two matching elements.

A comparison operator to match elements.

Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.

Implements [IIndexScanner\(T\).Join](#)

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Join
    (Of T2,TResult)( _
    ByVal source As System.Collections.Generic.IEnumerable(Of T2), _
    ByVal keySelector As System.Func(Of T2,Object), _
    ByVal resultSelector As System.Func(Of T,T2,TResult), _
    ByVal op As JoinOperator _
) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
public System.Collections.Generic.IEnumerable<TResult> Join<T2,TResult>(
    System.Collections.Generic.IEnumerable<T2> source,
    System.Func<T2,object> keySelector,
    System.Func<T,T2,TResult> resultSelector,
    JoinOperator op
)
```

## Parameters

*source*

The second sequence to join to this collection.

*keySelector*

A function to extract from a second sequence's item the value to match against this collection's key value.

*resultSelector*

A function to create a result element from two matching elements.

*op*

A comparison operator to match elements.

## Type Parameters

*T2*

The type of the elements of the second sequence.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to pairs of joined elements of the two collections.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T> Class](#)

[Index<T> Members](#)

[Overload List](#)

Join<T2,TResult>(IIndexScanner<T2>,Func<T,T2,TResult>,JoinOperator) Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) > [Join Method](#) :

Join<T2,TResult>(IIndexScanner<T2>,Func<T,T2,TResult>,JoinOperator) Method

The type of the elements of the second collection.

The type of the result elements.

The second indexed collection to join to this collection.

A function to create a result element from two matching elements.

A comparison operator to match elements.

Correlates the items of this indexed collection with the items of another indexed collection and returns the combined items with matching keys.

Implements [IIndexScanner\(T\).Join](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Join     (Of T2,TResult)( _     ByVal source As IIndexScanner(Of T2), _     ByVal resultSelector As System.Func(Of T,T2,TResult), _     ByVal op As JoinOperator _ ) As System.Collections.Generic.IEnumerable(Of TResult)</pre>	
C#	
<pre>public System.Collections.Generic.IEnumerable&lt;TResult&gt; Join&lt;T2,TResult&gt;(     IIndexScanner&lt;T2&gt; source,     System.Func&lt;T,T2,TResult&gt; resultSelector,     JoinOperator op )</pre>	

### Parameters

*source*

The second indexed collection to join to this collection.

*resultSelector*

A function to create a result element from two matching elements.

*op*

A comparison operator to match elements.

### Type Parameters

*T2*

The type of the elements of the second collection.

*TResult*

The type of the result elements.

### Return Value

Enumeration of objects obtained by applying the result selector to pairs of joined elements of the two collections.

## Remarks

Matching of two elements is performed by matching their keys.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference






[Index<T> Class](#)  
[Index<T> Members](#)  
[Overload List](#)





## Properties

[C1.LiveLinq.Indexing Namespace](#) : [Index<T> Class](#)

For a list of all members of this type, see [Index<T> members](#).

## Public Properties

	Name	Description
	<a href="#">Algorithm</a>	Gets the indexing algorithm used by the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">ItemCount</a>	Gets the number of elements in the indexed collection.
	<a href="#">KeyCount</a>	Gets the number of distinct key values in all items of this collection.
	<a href="#">KeyIsUnique</a>	Gets a value that indicates whether the key used in this index is a unique key for the collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeySelector</a>	Gets the expression used to obtain key value from an element of the indexed collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )

	<b>KeyType</b>	Gets the type of the index key. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<b>Locale</b>	Gets the locale information used to compare strings in the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<b>Root</b>	Gets the root index in an index/subindex hierarchy. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<b>Subindexes</b>	Gets the collection of subindexes added to this index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[Index<T> Class](#)

[C1.LiveLinq.Indexing Namespace](#)

[Index<T,TKey> Class](#)

### ItemCount Property

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) : ItemCount Property

Gets the number of elements in the indexed collection.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> ItemCount <b>As</b> System.Integer	
C#	
<b>public</b> System.int ItemCount { <b>get</b> ;}	

### Property Value

The number of elements in the indexed collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T> Class](#)  
[Index<T> Members](#)

### KeyCount Property

[C1.LiveLinq.Indexing Namespace](#) > [Index<T> Class](#) : KeyCount Property

Gets the number of distinct key values in all items of this collection.

## Syntax

Visual Basic (Declaration)	
<code>Public MustOverride ReadOnly Property KeyCount As System.Integer</code>	
C#	
<code>public abstract System.int KeyCount {get;}</code>	

### Property Value

Number of distinct key values in the collection.

## Remarks

This number is not the same as [ItemCount](#), unless the index key is a unique key of that collection, see [IndexDefinition<T>.KeyIsUnique](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T> Class](#)  
[Index<T> Members](#)



# Index<T,TKey>

[C1.LiveLinq.Indexing Namespace](#) : Index<T,TKey> Class

The type of the elements of the collection to index.

The type of the index key.

Indexes a collection by an expression (typically, by a field), providing fast access to items having particular values (or range of values) of that expression.

## Object Model

Index<T,TKey>

## Syntax

Visual Basic (Declaration)	
<pre>Public MustInherit Class Index     (Of T,TKey)     Inherits Index(Of T)     Implements C1.LiveLinq.Indexing.Search.IIndexScanner(Of T), C1.LiveLinq.Indexing.Search.IIndexScanner(Of T,TKey)</pre>	
C#	
<pre>public abstract class Index&lt;T,TKey&gt; : Index&lt;T&gt;, C1.LiveLinq.Indexing.Search.IIndexScanner&lt;T&gt;, C1.LiveLinq.Indexing.Search.IIndexScanner&lt;T,TKey&gt;</pre>	

## Type Parameters

*T*

The type of the elements of the collection to index.

*TKey*

The type of the index key.

## Remarks

Indexes can be created and added to a collection explicitly in code by calling [IndexCollection.Add](#), or their creation can be enforced in LINQ queries by using the [Indexed](#) hint.

In LINQ queries, indexes are used for optimizing query performance if that is specified with an [Indexed](#) hint. Usually, hints are not required, because LiveLinq can automatically determine that an index can be used to speedup a query, but using hints helps to ensure this optimization.

Indexes can also be used programmatically in code, without LINQ syntax, by using the methods of the [C1.LiveLinq.Indexing.Search.IIndexScanner<T,TKey>](#) interface that is implemented by the [Index<T,TKey>](#) class. For example, you can call such methods as [Find](#) directly for an [Index<T,TKey>](#) object.

It must be kept in mind that every index you create introduces a trade-off: it can dramatically speed up searches, but it consumes memory and adds some (usually, small) overhead every time the indexed collection (or any of the items, its elements) is modified. This is why indexes should generally be kept only while you need them for queries. To delete an index, use [IndexCollection.Remove](#).

An index can have subindexes, see [Subindex<T,TKey>](#). Subindexes are optional, not required for any indexing tasks, but can provide additional optimization and help minimize memory requirements when a collection is indexed by multiple keys. In presence of subindexes, an index is the root level of a tree of subindexes.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.IndexDefinition<T>](#)

[C1.LiveLinq.Indexing.Index<T>](#)

**[C1.LiveLinq.Indexing.Index<T,TKey>](#)**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Members](#)

[C1.LiveLinq.Indexing Namespace](#)

### Overview

[C1.LiveLinq.Indexing Namespace](#) : [Index<T,TKey>](#) Class

The type of the elements of the collection to index.

The type of the index key.

Indexes a collection by an expression (typically, by a field), providing fast access to items having particular values (or range of values) of that expression.

## Object Model

`Index<T,TKey>`

## Syntax

Visual Basic (Declaration)	
<pre>Public MustInherit Class Index     (Of T,TKey)     Inherits Index(Of T)     Implements C1.LiveLinq.Indexing.Search.IIndexScanner(Of T), C1.LiveLinq.Indexing.Search.IIndexScanner(Of T,TKey)</pre>	
C#	
<pre>public abstract class Index&lt;T,TKey&gt; : Index&lt;T&gt;, C1.LiveLinq.Indexing.Search.IIndexScanner&lt;T&gt;, C1.LiveLinq.Indexing.Search.IIndexScanner&lt;T,TKey&gt;</pre>	

## Type Parameters

*T*

The type of the elements of the collection to index.

*TKey*

The type of the index key.

## Remarks

Indexes can be created and added to a collection explicitly in code by calling [IndexCollection.Add](#), or their creation can be enforced in LINQ queries by using the [Indexed](#) hint.

In LINQ queries, indexes are used for optimizing query performance if that is specified with an [Indexed](#) hint. Usually, hints are not required, because LiveLinq can automatically determine that an index can be used to speedup a query, but using hints helps to ensure this optimization.

Indexes can also be used programmatically in code, without LINQ syntax, by using the methods of the [C1.LiveLinq.Indexing.Search.IIndexScanner<T,TKey>](#) interface

that is implemented by the [Index<T,TKey>](#) class. For example, you can call such methods as [Find](#) directly for an [Index<T,TKey>](#) object.

It must be kept in mind that every index you create introduces a trade-off: it can dramatically speed up searches, but it consumes memory and adds some (usually, small) overhead every time the indexed collection (or any of the items, its elements) is modified. This is why indexes should generally be kept only while you need them for queries. To delete an index, use [IndexCollection.Remove](#).

An index can have subindexes, see [Subindex<T,TKey>](#). Subindexes are optional, not required for any indexing tasks, but can provide additional optimization and help minimize memory requirements when a collection is indexed by multiple keys. In presence of subindexes, an index is the root level of a tree of subindexes.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.IndexDefinition<T>](#)

[C1.LiveLinq.Indexing.Index<T>](#)

**[C1.LiveLinq.Indexing.Index<T,TKey>](#)**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Members](#)

[C1.LiveLinq.Indexing Namespace](#)


### Members



[Properties](#) [Methods](#)

[C1.LiveLinq.Indexing Namespace](#) : [Index<T,TKey>](#) Class

The following tables list the members exposed by [Index<T,TKey>](#).



### Public Properties











	Name	Description
	<a href="#">Algorithm</a>	Gets the indexing algorithm used by the index. (Inherited from

		<a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">ItemCount</a>	Gets the number of elements in the indexed collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a> )
	<a href="#">KeyCount</a>	Gets the number of distinct key values in all items of this collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a> )
	<a href="#">KeyIsUnique</a>	Gets a value that indicates whether the key used in this index is a unique key for the collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeySelector</a>	Gets the expression used to obtain key value from an element of the indexed collection.
	<a href="#">KeyType</a>	Gets the type of the index key. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Locale</a>	Gets the locale information used to compare strings in the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Root</a>	Gets the root index in an index/subindex hierarchy. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Subindexes</a>	Gets the collection of subindexes added to this index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )

[Top](#)

## Public Methods

	Name	Description
	<a href="#">All</a>	Gets all items in the indexed collection.
	<a href="#">ContainsKey</a>	Overloaded. Returns a value that indicates whether the collection contains an item with the given key value.

≡  <a href="#">Find</a>	Overloaded. Finds items with the specified key value.
≡  <a href="#">FindBetween</a>	Overloaded. Finds items with key values in the interval between the specified values.
≡  <a href="#">FindGreater</a>	Overloaded. Finds items with keys greater than the specified value.
≡  <a href="#">FindKeys</a>	Overloaded. Finds items containing any of the specified key values.
≡  <a href="#">FindLess</a>	Overloaded. Finds items with keys less than the specified value.
≡  <a href="#">FindSingle</a>	Finds the only item with the specified key value and throws an exception if there is not exactly one item with that key value.
≡  <a href="#">FindStartingWith</a>	<p>Finds items with string key values starting with the specified string.</p> <p>Implements <a href="#">IIndexScanner(T).FindStartingWith(string,Func(string,bool),Order)</a></p> <p>(Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a>)</p>
≡  <a href="#">GroupJoin</a>	<p>Overloaded.</p> <p>Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.</p> <p>Implements <a href="#">IIndexScanner(T,TKey).GroupJoin</a></p>
≡  <a href="#">Join</a>	<p>Overloaded.</p> <p>Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.</p> <p>Implements <a href="#">IIndexScanner(T,TKey).Join</a></p>
≡  <a href="#">Keys</a>	Gets distinct key values in all items of the indexed collection.

[Top](#)

## See Also

### Reference

[Index<T,TKey> Class](#)  
[C1.LiveLinq.Indexing Namespace](#)




### Methods

[C1.LiveLinq.Indexing Namespace](#) : [Index<T,TKey> Class](#)

For a list of all members of this type, see [Index<T,TKey> members](#).

## Public Methods

	Name	Description
⇒	<a href="#">All</a>	Gets all items in the indexed collection.
⇒	<a href="#">ContainsKey</a>	Overloaded. Returns a value that indicates whether the collection contains an item with the given key value.
⇒	<a href="#">Find</a>	Overloaded. Finds items with the specified key value.
⇒	<a href="#">FindBetween</a>	Overloaded. Finds items with key values in the interval between the specified values.
⇒	<a href="#">FindGreater</a>	Overloaded. Finds items with keys greater than the specified value.
⇒	<a href="#">FindKeys</a>	Overloaded. Finds items containing any of the specified key values.
⇒	<a href="#">FindLess</a>	Overloaded. Finds items with keys less than the specified value.
⇒	<a href="#">FindSingle</a>	Finds the only item with the specified key value and throws an exception if there is not exactly one item with that key value.
⇒	<a href="#">FindStartingWith</a>	Finds items with string key values starting with the specified string.  Implements <a href="#">IIndexScanner(T).FindStartingWith(string,Func(string,bool),Order)</a>  (Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a> )

 <a href="#">GroupJoin</a>	<p>Overloaded.</p> <p>Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.</p> <p>Implements <a href="#">IIndexScanner(T,TKey).GroupJoin</a></p>
 <a href="#">Join</a>	<p>Overloaded.</p> <p>Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.</p> <p>Implements <a href="#">IIndexScanner(T,TKey).Join</a></p>
 <a href="#">Keys</a>	<p>Gets distinct key values in all items of the indexed collection.</p>

[Top](#)

## See Also

### Reference

[Index<T,TKey> Class](#)  
[C1.LiveLinq.Indexing Namespace](#)

### All Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) : All Method

Specifies the order of the key values to sort the result.

Gets all items in the indexed collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public MustOverride Function All( _     ByVal order As Order _ ) As IndexQuery(Of T,TKey)</pre>	
C#	



```
public abstract IndexQuery<T,TKey> All(
    Order order
)
```

## Parameters

*order*

Specifies the order of the key values to sort the result.

## Return Value

An object enumerating all items of the collection in the specified order of their key values.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T,TKey> Class](#)  
[Index<T,TKey> Members](#)

## ContainsKey Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) : ContainsKey Method

Returns a value that indicates whether the collection contains an item with the given key value.

## Overload List

Overload	Description
<a href="#">ContainsKey(TKey)</a>	Returns a value that indicates whether the collection contains an item with the given key value.
<a href="#">ContainsKey</a>	<p>Returns a value that indicates whether the indexed collection contains an item with the given key value.</p> <p>Implements <a href="#">IIndexScanner(T).ContainsKey(object)</a></p> <p>(Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a>)</p>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)

[Index<T,TKey> Members](#)

ContainsKey(TKey) Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) > [ContainsKey Method](#) : ContainsKey(TKey)  
Method

The key value to search for.

Returns a value that indicates whether the collection contains an item with the given key value.

## Syntax

Visual Basic (Declaration)

```
Public Overloads MustOverride Function ContainsKey( _  
    ByVal key As TKey _  
) As System.Boolean
```

C#

```
public abstract System.bool ContainsKey(  
    TKey key  
)
```

### Parameters

*key*

The key value to search for.

### Return Value

**true** if the collection contains an element with the specified key value; otherwise, **false**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)  
[Index<T,TKey> Members](#)  
[Overload List](#)

### Find Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) : Find Method

Finds items with the specified key value.

## Overload List

Overload	Description
<a href="#">Find(TKey)</a>	Finds items with the specified key value.
<a href="#">Find</a>	<p>Finds items with the specified key value.</p> <p>Implements <a href="#">IIndexScanner(T).Find(object)</a></p> <p>(Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a>)</p>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)  
[Index<T,TKey> Members](#)

### Find(TKey) Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) > [Find Method](#) : Find(TKey) Method

The key value to search for.

Finds items with the specified key value.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads MustOverride Function Find( _     ByVal key As TKey _ ) As IndexQuery(Of T,TKey)</pre>	
C#	
<pre>public abstract IndexQuery&lt;T,TKey&gt; Find(     TKey key )</pre>	

## Parameters

*key*

The key value to search for.

## Return Value

An object enumerating items having the specified key value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T,TKey> Class](#)  
[Index<T,TKey> Members](#)  
[Overload List](#)

## FindBetween Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) : FindBetween Method

Finds items with key values in the interval between the specified values.

## Overload List

Overload	Description
<a href="#">FindBetween(TKey,Boolean,TKey,Boolean,Func&lt;TK</a>	Finds items with key values in the interval

<a href="#">ey,Boolean&gt;,Order)</a>	between the specified values.
<a href="#">FindBetween</a>	<p>Finds items with key values in the interval between the specified values.</p> <p>Implements  <a href="#">IIndexScanner(T).FindBetween(object,bool,object,bool,Order)</a></p> <p>(Inherited from  <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a>)</p>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)

[Index<T,TKey> Members](#)

[FindBetween\(TKey,Boolean,TKey,Boolean,Func<TKey,Boolean>,Order\) Method](#)

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) > [FindBetween Method](#) :

[FindBetween\(TKey,Boolean,TKey,Boolean,Func<TKey,Boolean>,Order\) Method](#)

Minimum key value to search for.

If **true**, the result includes items with the minimum key value.

Maximum key value to search for.

If **true**, the result includes items with the maximum key value.

An optional condition that found items must satisfy.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with key values in the interval between the specified values.

## Syntax

## Visual Basic (Declaration)

```
Public Overloads Overridable Function FindBetween( _  
    ByVal min As TKey, _  
    ByVal minInclusive As System.Boolean, _  
    ByVal max As TKey, _  
    ByVal maxInclusive As System.Boolean, _  
    ByVal keyPredicate As System.Func(Of TKey, Boolean), _  
    ByVal order As Order _  
) As IndexQuery(Of T, TKey)
```

## C#

```
public virtual IndexQuery<T, TKey> FindBetween(  
    TKey min,  
    System.bool minInclusive,  
    TKey max,  
    System.bool maxInclusive,  
    System.Func<TKey, bool> keyPredicate,  
    Order order  
)
```

## Parameters

*min*

Minimum key value to search for.

*minInclusive*

If **true**, the result includes items with the minimum key value.

*max*

Maximum key value to search for.

*maxInclusive*

If **true**, the result includes items with the maximum key value.

*keyPredicate*

An optional condition that found items must satisfy.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLink.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items with key values within the specified limits.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T,TKey> Class](#)  
[Index<T,TKey> Members](#)  
[Overload List](#)

## FindGreater Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) : FindGreater Method

Finds items with keys greater than the specified value.

## Overload List

Overload	Description
<a href="#">FindGreater(TKey,Boolean,Func&lt;TKey,Boolean&gt;,Order)</a>	Finds items with keys greater than the specified value.
<a href="#">FindGreater</a>	<p>Finds items with keys greater than the specified value.</p> <p>Implements <a href="#">IIndexScanner(T).FindGreater(object,bool,Order)</a></p> <p>(Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a>)</p>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)

[Index<T,TKey> Members](#)

[FindGreater\(TKey,Boolean,Func<TKey,Boolean>,Order\) Method](#)

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) > [FindGreater Method](#) :

[FindGreater\(TKey,Boolean,Func<TKey,Boolean>,Order\) Method](#)

Minimum key value to search for.

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly greater than the specified value.

An optional condition that found items must satisfy.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with keys greater than the specified value.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Overridable Function FindGreater( _  
    ByVal key As TKey, _  
    ByVal inclusive As System.Boolean, _  
    ByVal keyPredicate As System.Func(Of TKey, Boolean), _  
    ByVal order As Order _  
) As IndexQuery(Of T, TKey)
```

C#

```
public virtual IndexQuery<T,TKey> FindGreater(  
    TKey key,  
    System.bool inclusive,  
    System.Func<TKey,bool> keyPredicate,  
    Order order  
)
```

### Parameters



*key*

Minimum key value to search for.

*inclusive*

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly greater than the specified value.

*keyPredicate*

An optional condition that found items must satisfy.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values are greater than the specified value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T,TKey> Class](#)  
[Index<T,TKey> Members](#)  
[Overload List](#)

## FindKeys Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) : FindKeys Method

Finds items containing any of the specified key values.

## Overload List

Overload	Description
<a href="#">FindKeys(IEnumerable&lt;TKey&gt;,Order)</a>	Finds items containing any of the specified key values.

<a href="#">FindKeys</a>	<p>Finds items containing any of the specified key values.</p> <p>Implements <a href="#">FindKeys(IEnumerable,Order)</a></p> <p>(Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a>)</p>
--------------------------	--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)  
[Index<T,TKey> Members](#)

[FindKeys\(IEnumerable<TKey>,Order\) Method](#)  
[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) > [FindKeys Method](#) :  
[FindKeys\(IEnumerable<TKey>,Order\) Method](#)

The key values to search for.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items containing any of the specified key values.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads MustOverride Function FindKeys( _     ByVal keys As System.Collections.Generic.IEnumerable(Of TKey), _     ByVal order As Order _ ) As IndexQuery(Of T,TKey)</pre>	
C#	
<pre>public abstract IndexQuery&lt;T,TKey&gt; FindKeys(     System.Collections.Generic.IEnumerable&lt;TKey&gt; keys,     Order order )</pre>	

### Parameters

*keys*

The key values to search for.

*order*

Optionally specifies the order of the key values to sort the result  
([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values belong to the specified key value collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T,TKey> Class](#)

[Index<T,TKey> Members](#)

[Overload List](#)

## FindLess Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) : FindLess Method

Finds items with keys less than the specified value.

## Overload List

Overload	Description
<a href="#">FindLess(TKey,Boolean,Func&lt;TKey,Boolean&gt;,Order)</a>	Finds items with keys less than the specified value.
<a href="#">FindLess</a>	Finds items with keys less than the specified value.  Implements <a href="#">IIndexScanner(T).FindLess(object,bool,Order</a>

	)  (Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a> )
--	---

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)  
[Index<T,TKey> Members](#)

FindLess(TKey,Boolean,Func<TKey,Boolean>,Order) Method  
[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) > [FindLess Method](#) :  
 FindLess(TKey,Boolean,Func<TKey,Boolean>,Order) Method

Maximum key value to search for.

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly less than the specified value.

An optional condition that found items must satisfy.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with keys less than the specified value.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Overridable Function FindLess( _     ByVal key As TKey, _     ByVal inclusive As System.Boolean, _     ByVal keyPredicate As System.Func(Of TKey, Boolean), _     ByVal order As Order _ ) As IndexQuery(Of T, TKey)</pre>	
C#	

```
public virtual IndexQuery<T,TKey> FindLess(  
    TKey key,  
    System.bool inclusive,  
    System.Func<TKey,bool> keyPredicate,  
    Order order  
)
```

## Parameters

*key*

Maximum key value to search for.

*inclusive*

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly less than the specified value.

*keyPredicate*

An optional condition that found items must satisfy.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values are less than the specified value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T,TKey> Class](#)  
[Index<T,TKey> Members](#)  
[Overload List](#)

## FindSingle Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) : FindSingle Method

The key value to search for.

Finds the only item with the specified key value and throws an exception if there is not exactly one item with that key value.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overridable Function FindSingle( _     ByVal key As TKey _ ) As T</pre>	
C#	
<pre>public virtual T FindSingle(     TKey key )</pre>	

### Parameters

*key*

The key value to search for.

### Return Value

The item of the collection that contains the specified key value, if found.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)

[Index<T,TKey> Members](#)

### GroupJoin Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) : GroupJoin Method

Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.

Implements [IIndexScanner\(T,TKey\).GroupJoin](#)

## Overload List

Overload	Description
<a href="#">GroupJoin&lt;T2,TResult&gt;(IIndexScanner&lt;T2,TKey&gt;,Func&lt;T,IEnumerable&lt;T2&gt;,TResult&gt;)</a>	<p>Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.</p> <p>Implements <a href="#">IIndexScanner(T,TKey).GroupJoin</a></p>
<a href="#">GroupJoin&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,TKey&gt;,Func&lt;IEnumerable&lt;T&gt;,T2,TResult&gt;)</a>	<p>Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of the second sequence.</p> <p>Implements <a href="#">IIndexScanner(T,TKey).GroupJoin</a></p>
<a href="#">GroupJoin&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,TKey&gt;,Func&lt;IEnumerable&lt;T2&gt;,TResult&gt;)</a>	<p>Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of this collection.</p>

	<p>Implements</p> <p><a href="#">IIndexScanner(T,TKey).GroupJoin</a></p>
<p><a href="#">GroupJoin&lt;T2,TResult&gt;(IIndexScanner&lt;T2&gt;,Func&lt;T,IEnumerable&lt;T2&gt;,TResult&gt;)</a></p>	<p>Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.</p> <p>Implements</p> <p><a href="#">IIndexScanner(T).GroupJoin</a></p> <p>(Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a>)</p>
<p><a href="#">GroupJoin&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,Object&gt;,Func&lt;IEnumerable&lt;T&gt;,T2,TResult&gt;)</a></p>	<p>Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of the second sequence.</p> <p>Implements</p> <p><a href="#">IIndexScanner(T).GroupJoin</a></p> <p>(Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a>)</p>
<p><a href="#">GroupJoin&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,Object&gt;,Func&lt;T,IEnumerable&lt;T2&gt;,TResult&gt;)</a></p>	<p>Correlates the items of</p>



<code>numerable&lt;T2&gt;,TResult&gt;)</code>	<p>this indexed collection with the items of another sequence and groups the results by the item of this collection.</p> <p>Implements <a href="#">IIndexScanner(T).GroupJoin</a></p> <p>(Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a>)</p>
---	---

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)

[Index<T,TKey> Members](#)

[GroupJoin<T2,TResult>\(IIndexScanner<T2,TKey>,Func<T,IEnumerable<T2>,TResult>\) Method](#)

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) > [GroupJoin Method](#) :

[GroupJoin<T2,TResult>\(IIndexScanner<T2,TKey>,Func<T,IEnumerable<T2>,TResult>\) Method](#)

The type of the elements of the second collection.

The type of the result elements.

The second indexed collection to join to this collection.

A function to create a result element from an element from this collection and a collection of matching elements from the second collection.

Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.

Implements [IIndexScanner\(T,TKey\).GroupJoin](#)

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function GroupJoin  
    (Of T2,TResult)( _  
        ByVal source As IIndexScanner(Of T2,TKey), _  
        ByVal resultSelector As System.Func(Of T,IEnumerable(Of T2),TResult) _  
    ) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
public System.Collections.Generic.IEnumerable<TResult> GroupJoin<T2,TResult>(  
    IIndexScanner<T2,TKey> source,  
    System.Func<T,IEnumerable<T2>,TResult> resultSelector  
)
```

### Parameters

*source*

The second indexed collection to join to this collection.

*resultSelector*

A function to create a result element from an element from this collection and a collection of matching elements from the second collection.

### Type Parameters

*T2*

The type of the elements of the second collection.

*TResult*

The type of the result elements.

### Return Value

Enumeration of objects obtained by applying the result selector to group pairs, where each pair consists of an item of this collection and the corresponding enumeration of the items of the second collection joined to it.

## Remarks

Matching of two elements is performed by matching their keys.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)

[Index<T,TKey> Members](#)

[Overload List](#)

`GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<IEnumerable<T>,T2,TResult>) Method`

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) > [GroupJoin Method](#) :

`GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<IEnumerable<T>,T2,TResult>) Method`

The type of the elements of the second sequence.

The type of the result elements.

The second sequence to join to this collection.

A function to extract from an item of the second sequence the value to match against this collection's key value.

A function to create a result element from an element of the second sequence and the collection of matching elements from this collection.

Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of the second sequence.

Implements [IIndexScanner\(T,TKey\).GroupJoin](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function GroupJoin     (Of T2,TResult)( _     ByVal source As System.Collections.Generic.IEnumerable(Of T2), _     ByVal keySelector As System.Func(Of T2,TKey), _     ByVal resultSelector As System.Func(Of IEnumerable(Of T),T2,TResult) _     ) As System.Collections.Generic.IEnumerable(Of TResult)</pre>	

C#

```
public System.Collections.Generic.IEnumerable<TResult> GroupJoin<T2,TResult>(
    System.Collections.Generic.IEnumerable<T2> source,
    System.Func<T2,TKey> keySelector,
    System.Func<IEnumerable<T>,T2,TResult> resultSelector
)
```

## Parameters

*source*

The second sequence to join to this collection.

*keySelector*

A function to extract from an item of the second sequence the value to match against this collection's key value.

*resultSelector*

A function to create a result element from an element of the second sequence and the collection of matching elements from this collection.

## Type Parameters

*T2*

The type of the elements of the second sequence.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to group pairs, where each pair consists of an item of the second collection and the corresponding enumeration of the items of this collection joined to it.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T,TKey> Class](#)  
[Index<T,TKey> Members](#)  
[Overload List](#)

`GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<T,IEnumerable<T2>,TResult>)`  
Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) > [GroupJoin Method](#) :

`GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<T,IEnumerable<T2>,TResult>)` Method

The type of the elements of the second sequence.

The type of the result elements.

The second sequence to join to this collection.

A function to extract from an item of the second sequence the value to match against this collection's key value.

A function to create a result element from an element from this collection and a collection of matching elements from the second sequence.

Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of this collection.

Implements [IIndexScanner\(T,TKey\).GroupJoin](#)

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function GroupJoin
    (Of T2,TResult)( _
    ByVal source As System.Collections.Generic.IEnumerable(Of T2), _
    ByVal keySelector As System.Func(Of T2,TKey), _
    ByVal resultSelector As System.Func(Of T,IEnumerable(Of T2),TResult) _
) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
public System.Collections.Generic.IEnumerable<TResult> GroupJoin<T2,TResult>(
    System.Collections.Generic.IEnumerable<T2> source,
    System.Func<T2,TKey> keySelector,
    System.Func<T,IEnumerable<T2>,TResult> resultSelector
)
```

## Parameters

*source*

The second sequence to join to this collection.

*keySelector*

A function to extract from an item of the second sequence the value to match against this collection's key value.

*resultSelector*

A function to create a result element from an element from this collection and a collection of matching elements from the second sequence.

## Type Parameters

*T2*

The type of the elements of the second sequence.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to group pairs, where each pair consists of an item of this collection and the corresponding enumeration of the items of the second sequence joined to it.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T,TKey> Class](#)

[Index<T,TKey> Members](#)

[Overload List](#)

## Join Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) : Join Method

Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.

Implements [IIndexScanner\(T,TKey\).Join](#)

## Overload List

Overload	Description
<a href="#">Join&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,TKey&gt;,Func&lt;T,T2,TResult&gt;,JoinOperator)</a>	<p>Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.</p> <p>Implements <a href="#">IIndexScanner(T,TKey).Join</a></p>
<a href="#">Join&lt;T2,TResult&gt;(IIndexScanner&lt;T2,TKey&gt;,Func&lt;T,T2,TResult&gt;,JoinOperator)</a>	<p>Correlates the items of this indexed collection with the items of another indexed collection and returns the combined items with matching keys.</p> <p>Implements <a href="#">IIndexScanner(T,TKey).Join</a></p>
<a href="#">Join&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,Object&gt;,Func&lt;T,T2,TResult&gt;,JoinOperator)</a>	<p>Correlates the items of this indexed collection with the items of another sequence and</p>

	<p>returns the combined items with matching keys.</p> <p>Implements <a href="#">IIndexScanner(T).Join</a></p> <p>(Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a>)</p>
<a href="#">Join&lt;T2,TResult&gt;(IIndexScanner&lt;T2&gt;,Func&lt;T,T2,TResult&gt;,JoinOperator)</a>	<p>Correlates the items of this indexed collection with the items of another indexed collection and returns the combined items with matching keys.</p> <p>Implements <a href="#">IIndexScanner(T).Join</a></p> <p>(Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a>)</p>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)

[Index<T,TKey> Members](#)

[Join<T2,TResult>\(IEnumerable<T2>,Func<T2,TKey>,Func<T,T2,TResult>,JoinOperator\)](#) Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) > [Join Method](#) :

[Join<T2,TResult>\(IEnumerable<T2>,Func<T2,TKey>,Func<T,T2,TResult>,JoinOperator\)](#) Method



The type of the elements of the second sequence.

The type of the result elements.

The second sequence to join to this collection.

A function to extract from a second sequence's item the value to match against this collection's key value.

A function to create a result element from two matching elements.

A comparison operator to match elements.

Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.

Implements [IIndexScanner\(T,TKey\).Join](#)

## Syntax

### Visual Basic (Declaration)

```
Public Overloads Function Join
    (Of T2,TResult)( _
    ByVal source As System.Collections.Generic.IEnumerable(Of T2), _
    ByVal keySelector As System.Func(Of T2,TKey), _
    ByVal resultSelector As System.Func(Of T,T2,TResult), _
    ByVal op As JoinOperator _
) As System.Collections.Generic.IEnumerable(Of TResult)
```

### C#

```
public System.Collections.Generic.IEnumerable<TResult> Join<T2,TResult>(
    System.Collections.Generic.IEnumerable<T2> source,
    System.Func<T2,TKey> keySelector,
    System.Func<T,T2,TResult> resultSelector,
    JoinOperator op
)
```

### Parameters

*source*

The second sequence to join to this collection.

*keySelector*

A function to extract from a second sequence's item the value to match against this collection's key value.

*resultSelector*

A function to create a result element from two matching elements.

*op*

A comparison operator to match elements.

## Type Parameters

*T2*

The type of the elements of the second sequence.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to pairs of joined elements of the two collections.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T,TKey> Class](#)

[Index<T,TKey> Members](#)

[Overload List](#)

[Join<T2,TResult>\(IIndexScanner<T2,TKey>,Func<T,T2,TResult>,JoinOperator\) Method](#)

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) > [Join Method](#) :

[Join<T2,TResult>\(IIndexScanner<T2,TKey>,Func<T,T2,TResult>,JoinOperator\) Method](#)

The type of the elements of the second collection.

The type of the result elements.

The second indexed collection to join to this collection.

A function to create a result element from two matching elements.

A comparison operator to match elements.

Correlates the items of this indexed collection with the items of another indexed collection and returns the combined items with matching keys.

Implements [IIndexScanner\(T,TKey\).Join](#)

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Join
    (Of T2,TResult)( _
    ByVal source As IIndexScanner(Of T2,TKey), _
    ByVal resultSelector As System.Func(Of T,T2,TResult), _
    ByVal op As JoinOperator _
) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
public System.Collections.Generic.IEnumerable<TResult> Join<T2,TResult>(
    IIndexScanner<T2,TKey> source,
    System.Func<T,T2,TResult> resultSelector,
    JoinOperator op
)
```

## Parameters

*source*

The second indexed collection to join to this collection.

*resultSelector*

A function to create a result element from two matching elements.

*op*

A comparison operator to match elements.

## Type Parameters

*T2*

The type of the elements of the second collection.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to pairs of joined elements of the two collections.

## Remarks

Matching of two elements is performed by matching their keys.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)  
[Index<T,TKey> Members](#)  
[Overload List](#)

### Keys Method

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) : Keys Method

Specifies the order of the key values to sort the result.

Gets distinct key values in all items of the indexed collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public MustOverride Function Keys( _     ByVal order As Order _ ) As System.Collections.Generic.IEnumerable(Of TKey)</pre>	
C#	
<pre>public abstract System.Collections.Generic.IEnumerable&lt;TKey&gt; Keys(     Order order )</pre>	

### Parameters

*order*

Specifies the order of the key values to sort the result.

## Return Value

All distinct key values contained in the items of the collection in the specified order.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[Index<T,TKey> Class](#)






[Index<T,TKey> Members](#)





## Properties

[C1.LiveLinq.Indexing Namespace](#) : [Index<T,TKey> Class](#)

For a list of all members of this type, see [Index<T,TKey> members](#).

## Public Properties

	Name	Description
	<a href="#">Algorithm</a>	Gets the indexing algorithm used by the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">ItemCount</a>	Gets the number of elements in the indexed collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a> )
	<a href="#">KeyCount</a>	Gets the number of distinct key values in all items of this collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a> )
	<a href="#">KeyIsUnique</a>	Gets a value that indicates whether the key used in this index is a unique key for the collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeySelector</a>	Gets the expression used to obtain key value from an element of the indexed collection.

	<b>KeyType</b>	Gets the type of the index key. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<b>Locale</b>	Gets the locale information used to compare strings in the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<b>Root</b>	Gets the root index in an index/subindex hierarchy. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<b>Subindexes</b>	Gets the collection of subindexes added to this index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[Index<T,TKey> Class](#)  
[C1.LiveLinq.Indexing Namespace](#)

### KeySelector Property

[C1.LiveLinq.Indexing Namespace](#) > [Index<T,TKey> Class](#) : KeySelector Property

Gets the expression used to obtain key value from an element of the indexed collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public Shadows ReadOnly Property KeySelector As System.Linq.Expressions.Expression(Of Func(Of T,TKey))</pre>	
C#	
<pre>public new System.Linq.Expressions.Expression&lt;Func&lt;T,TKey&gt;&gt; KeySelector {get;}</pre>	

### Property Value

An expression calculating the key value from an item (element of the collection). Typically, this is a field or a property in the item class, although more complex expressions can also be used.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Index<T,TKey> Class](#)

[Index<T,TKey> Members](#)

## IndexCollection<T>

[C1.LiveLinq.Indexing Namespace](#) : IndexCollection<T> Class

The type of the elements of the indexed collection.

Represents a collection of indexes attached to an indexed collection.

## Object Model

IndexCollection<T>

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>
Public Class IndexCollection(Of T)
    Inherits ScannerCollection(Of T)
```

C#

```
[System.Reflection.DefaultMember("Item")]
public class IndexCollection<T> : ScannerCollection<T>
```

## Type Parameters

*T*

The type of the elements of the indexed collection.

## Remarks

Any indexed collection (implementing the [IIndexedSource<T>](#) interface) has a collection of indexes attached to it.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.ScannerCollection<T>](#)

**C1.LiveLinq.Indexing.IndexCollection<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexCollection<T> Members](#)

[C1.LiveLinq.Indexing Namespace](#)

[Indexes Property](#)

[Indexes Property](#)

[Indexes Property](#)

[Indexes Property](#)

[Indexes Property](#)

### Overview

[C1.LiveLinq.Indexing Namespace](#) : [IndexCollection<T>](#) Class

The type of the elements of the indexed collection.

Represents a collection of indexes attached to an indexed collection.

## Object Model

[IndexCollection<T>](#)

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>
Public Class IndexCollection(Of T)
    Inherits ScannerCollection(Of T)
```

C#

```
[System.Reflection.DefaultMember("Item")]
public class IndexCollection<T> : ScannerCollection<T>
```



# Type Parameters

T

The type of the elements of the indexed collection.

## Remarks

Any indexed collection (implementing the [IIndexedSource<T>](#) interface) has a collection of indexes attached to it.

## Inheritance Hierarchy

System.Object  
[C1.LiveLinq.Indexing.ScannerCollection<T>](#)  
**C1.LiveLinq.Indexing.IndexCollection<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

- [IndexCollection<T> Members](#)
- [C1.LiveLinq.Indexing Namespace](#)
- [Indexes Property](#)
- [Indexes Property](#)
- [Indexes Property](#)
- [Indexes Property](#)
- [Indexes Property](#)

## Members


[Properties](#) [Methods](#)

[C1.LiveLinq.Indexing Namespace](#) : [IndexCollection<T>](#) Class

The following tables list the members exposed by [IndexCollection<T>](#).



## Public Constructors

Name	Description
------	-------------

	<a href="#">IndexCollection&lt;T&gt; Constructor</a>	Initializes a new instance of the <a href="#">IndexCollection&lt;T&gt;</a> class.
---	--	---







[Top](#)

## Public Properties

	Name	Description
	<a href="#">Count</a>	Overridden. Gets the number of indexes in the collection.
	<a href="#">Item</a>	Gets the index with the specified key selector.

[Top](#)

## Public Methods

	Name	Description
	<a href="#">Add</a>	Overloaded. Creates a new index and adds it to the collection of indexes.
	<a href="#">Clear</a>	Overridden. Clears the collection of all indexes. All indexes are detached from the indexed collection and destroyed.
	<a href="#">Contains</a>	Overloaded. Determines whether an index with the specified key selector exists in the collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.ScannerCollection&lt;T&gt;</a> )
	<a href="#">Find</a>	Overloaded. Finds an index in the collection by its key selector.
	<a href="#">GetEnumerator</a>	Overridden. Returns an enumerator that iterates through the <a href="#">IndexCollection&lt;T&gt;</a> .
	<a href="#">Remove</a>	Overloaded. Overridden. Removes an index from the collection.

[Top](#)

## See Also

### Reference

[IndexCollection<T> Class](#)  
[C1.LiveLinq.Indexing Namespace](#)  
[Indexes Property](#)  
[Indexes Property](#)  
[Indexes Property](#)  
[Indexes Property](#)  
[Indexes Property](#)

## IndexCollection<T> Constructor

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) : [IndexCollection<T> Constructor](#)

The collection to be indexed.

Initializes a new instance of the [IndexCollection<T>](#) class.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function New( _     ByVal source As IObservableSource(Of T) _ )</pre>	
C#	
<pre>public IndexCollection&lt;T&gt;(     IObservableSource&lt;T&gt; source )</pre>	

### Parameters

*source*

The collection to be indexed.

## Remarks

Normally, you don't need to create instances of the [IndexCollection<T>](#) class in your code, they already exist in the instances of LiveLinq indexed collection classes. You need to create them explicitly in code only if you define your own indexable collection class and then only if it does not inherit from [C1.LiveLinq.Collections.IndexedCollection<T>](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexCollection<T> Class](#)

[IndexCollection<T> Members](#)

### Methods

[C1.LiveLinq.Indexing Namespace](#) : [IndexCollection<T> Class](#)

For a list of all members of this type, see [IndexCollection<T> members](#).

### Public Methods

	Name	Description
≡	<a href="#">Add</a>	Overloaded. Creates a new index and adds it to the collection of indexes.
≡	<a href="#">Clear</a>	Overridden. Clears the collection of all indexes. All indexes are detached from the indexed collection and destroyed.
≡	<a href="#">Contains</a>	Overloaded. Determines whether an index with the specified key selector exists in the collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.ScannerCollection&lt;T&gt;</a> )
≡	<a href="#">Find</a>	Overloaded. Finds an index in the collection by its key selector.
≡	<a href="#">GetEnumerator</a>	Overridden. Returns an enumerator that iterates through the <a href="#">IndexCollection&lt;T&gt;</a> .
≡	<a href="#">Remove</a>	Overloaded. Overridden. Removes an index from the collection.

[Top](#)

## See Also

### Reference

[IndexCollection<T> Class](#)

[C1.LiveLinq.Indexing Namespace](#)

[Indexes Property](#)

[Indexes Property](#)

Indexes Property  
 Indexes Property  
 Indexes Property

## Add Method

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) : Add Method

Creates a new index and adds it to the collection of indexes.

## Overload List

Overload	Description
<a href="#">Add&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;)</a>	Creates a new index and adds it to the collection of indexes.
<a href="#">Add&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Boolean)</a>	Creates a new index and adds it to the collection of indexes.
<a href="#">Add&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Boolean,Boolean,IndexingAlgorithm,CultureInfo)</a>	Creates a new index and adds it to the collection of indexes. (Inherited from <a href="#">C1.LiveLinq.Indexing.ScannerCollection&lt;T&gt;</a> )
<a href="#">Add&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Boolean,Boolean,CultureInfo)</a>	Creates a new index and adds it to the collection of indexes. (Inherited from <a href="#">C1.LiveLinq.Indexing.ScannerCollection&lt;T&gt;</a> )
<a href="#">Add&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Boolean,Boolean)</a>	Creates a new index and adds it to the collection of indexes. (Inherited from <a href="#">C1.LiveLinq.Indexing.ScannerCollection&lt;T&gt;</a> )
<a href="#">Add(LambdaExpression,Boolean,Boolean,IndexingAlgorithm,CultureInfo)</a>	Creates a new index and adds it to the collection of indexes. (Inherited from

	<a href="#">C1.LiveLinq.Indexing.ScannerCollection&lt;T&gt;)</a>
--	--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexCollection<T> Class](#)

[IndexCollection<T> Members](#)

Add<TKey>(Expression<Func<T,TKey>>) Method

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) > [Add Method](#) :

Add<TKey>(Expression<Func<T,TKey>>) Method

The type of the index key.

Key selector expression of the index, see [IndexDefinition<T>.KeySelector](#).

Creates a new index and adds it to the collection of indexes.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Add(Of TKey)( _     ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of T,TKey)) _ ) As Index(Of T,TKey)</pre>	
C#	
<pre>public Index&lt;T,TKey&gt; Add&lt;TKey&gt;(     System.Linq.Expressions.Expression&lt;Func&lt;T,TKey&gt;&gt; keySelector )</pre>	

### Parameters

*keySelector*

Key selector expression of the index, see [IndexDefinition<T>.KeySelector](#).

### Type Parameters

*TKey*

The type of the index key.

## Return Value

The new index added to the collection of indexes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexCollection<T> Class](#)  
[IndexCollection<T> Members](#)  
[Overload List](#)

Add<TKey>(Expression<Func<T,TKey>>,Boolean) Method

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) > [Add Method](#) :

Add<TKey>(Expression<Func<T,TKey>>,Boolean) Method

The type of the index key.

Key selector expression of the index, see [IndexDefinition<T>.KeySelector](#).

**true** if a unique index must be created.

Creates a new index and adds it to the collection of indexes.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Add(Of TKey)( _     ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of T, TKey))), _     ByVal keyIsUnique As System.Boolean _ ) As Index(Of T, TKey)</pre>	
C#	
<pre>public Index&lt;T, TKey&gt; Add&lt;TKey&gt;(     System.Linq.Expressions.Expression&lt;Func&lt;T, TKey&gt;&gt; keySelector,</pre>	

```
System.bool keyIsUnque  
)
```

## Parameters

*keySelector*

Key selector expression of the index, see [IndexDefinition<T>.KeySelector](#).

*keyIsUnque*

**true** if a unique index must be created.

## Type Parameters

*TKey*

The type of the index key.

## Return Value

The new index added to the collection of indexes.

## Remarks

A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique.

But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the indexed collection. An attempt to modify the indexed collection violating the uniqueness throws an **System.InvalidOperationException**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexCollection<T> Class](#)

[IndexCollection<T> Members](#)

[Overload List](#)



## Clear Method

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) : Clear Method

Clears the collection of all indexes. All indexes are detached from the indexed collection and destroyed.

## Syntax

Visual Basic (Declaration)	
<code>Public Overrides Sub Clear()</code>	
C#	
<code>public override void Clear()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexCollection<T> Class](#)

[IndexCollection<T> Members](#)

## Find Method

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) : Find Method

Finds an index in the collection by its key selector.

## Overload List

Overload	Description
<a href="#">Find(LambdaExpression)</a>	Finds an index in the collection by its key selector.
<a href="#">Find&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;)</a>	Finds an index in the collection by its key selector.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexCollection<T> Class](#)

[IndexCollection<T> Members](#)

[Find\(LambdaExpression\) Method](#)

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) > [Find Method](#) : Find(LambdaExpression) Method

Key selector expression of an index, see [IndexDefinition<T>.KeySelector](#).

Finds an index in the collection by its key selector.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Shadows Function Find( _     ByVal keySelector As System.Linq.Expressions.LambdaExpression _ ) As Index(Of T)</pre>	
C#	
<pre>public new Index&lt;T&gt; Find(     System.Linq.Expressions.LambdaExpression keySelector )</pre>	

### Parameters

*keySelector*

Key selector expression of an index, see [IndexDefinition<T>.KeySelector](#).

### Return Value

An index with the given key selector, if it is found; otherwise, **null**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexCollection<T> Class](#)  
[IndexCollection<T> Members](#)  
[Overload List](#)

[Find<TKey>\(Expression<Func<T,TKey>>\) Method](#)

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) > [Find Method](#) :

[Find<TKey>\(Expression<Func<T,TKey>>\) Method](#)

The type of the index key.

Key selector expression of an index, see [IndexDefinition<T>.KeySelector](#).

Finds an index in the collection by its key selector.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Find(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
) As Index(Of T, TKey)
```

C#

```
public Index<T, TKey> Find<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector  
)
```

### Parameters

*keySelector*

Key selector expression of an index, see [IndexDefinition<T>.KeySelector](#).

### Type Parameters

*TKey*

The type of the index key.

### Return Value

An index with the given key selector, if it is found; otherwise, **null**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexCollection<T> Class](#)  
[IndexCollection<T> Members](#)  
[Overload List](#)

### GetEnumerator Method

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) : GetEnumerator Method

Returns an enumerator that iterates through the [IndexCollection<T>](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Overrides Function GetEnumerator() As System.Collections.Generic.IEnumerator(Of IIndexScanner(Of T))</pre>	
C#	
<pre>public override System.Collections.Generic.IEnumerator&lt;IIndexScanner&lt;T&gt;&gt; GetEnumerator()</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexCollection<T> Class](#)  
[IndexCollection<T> Members](#)

### Remove Method

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) : Remove Method

Removes an index from the collection.

## Overload List

Overload	Description
<a href="#">Remove(LambdaExpression)</a>	Removes an index from the collection.
<a href="#">Remove(Index&lt;T&gt;)</a>	Removes an index from the collection.
<a href="#">Remove&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;)</a>	Removes an index from the collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.ScannerCollection&lt;T&gt;</a> )

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexCollection<T> Class](#)

[IndexCollection<T> Members](#)

[Remove\(LambdaExpression\) Method](#)

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) > [Remove Method](#) :

[Remove\(LambdaExpression\) Method](#)

Key selector expression of an index, see [IndexDefinition<T>.KeySelector](#).

Removes an index from the collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Overrides Function Remove( _     ByVal keySelector As System.Linq.Expressions.LambdaExpression _ ) As System.Boolean</pre>	
C#	
<pre>public override System.bool Remove(</pre>	

```
System.Linq.Expressions.LambdaExpression keySelector  
)
```

## Parameters

*keySelector*

Key selector expression of an index, see [IndexDefinition<T>.KeySelector](#).

## Return Value

**true** if an index has been removed; **false** if there is no index with the given key selector in the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexCollection<T> Class](#)

[IndexCollection<T> Members](#)

[Overload List](#)

Remove(Index<T>) Method

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) > [Remove Method](#) : Remove(Index<T>) Method

The index to remove.

Removes an index from the collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Remove( _     ByVal index As Index(Of T) _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool Remove(     Index&lt;T&gt; index</pre>	

)

## Parameters

*index*

The index to remove.

## Return Value

**true** if an index has been removed; **false** if the index does not belong to this collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexCollection<T> Class](#)

[IndexCollection<T> Members](#)



[Overload List](#)

## Properties

[C1.LiveLinq.Indexing Namespace](#) : [IndexCollection<T> Class](#)

For a list of all members of this type, see [IndexCollection<T> members](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	Overridden. Gets the number of indexes in the collection.
	<a href="#">Item</a>	Gets the index with the specified key selector.

[Top](#)

## See Also

## Reference

[IndexCollection<T> Class](#)

[C1.LiveLinq.Indexing Namespace](#)

[Indexes Property](#)

[Indexes Property](#)  
[Indexes Property](#)  
[Indexes Property](#)  
[Indexes Property](#)

Count Property

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) : Count Property

Gets the number of indexes in the collection.

Syntax

Visual Basic (Declaration)	
<code>Public Overrides ReadOnly Property Count As System.Integer</code>	
C#	
<code>public override System.int Count {get;}</code>	

Property Value

The number of indexes in the collection.

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[IndexCollection<T> Class](#)  
[IndexCollection<T> Members](#)

Item Property

[C1.LiveLinq.Indexing Namespace](#) > [IndexCollection<T> Class](#) : Item Property

Key selector expression of an index, see [IndexDefinition<T>.KeySelector](#).

Gets the index with the specified key selector.

Syntax

Visual Basic (Declaration)	
----------------------------	--



```
Public ReadOnly Default Property Item( _
    ByVal key As System.Linq.Expressions.LambdaExpression _
) As Index(Of T)
```

C#

```
public Index<T> this[
    System.Linq.Expressions.LambdaExpression key
]; {get;}
```

## Parameters

*key*

Key selector expression of an index, see [IndexDefinition<T>.KeySelector](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexCollection<T> Class](#)

[IndexCollection<T> Members](#)

## IndexDefinition<T>

[C1.LiveLinq.Indexing Namespace](#) : [IndexDefinition<T> Class](#)

The type of the elements of indexed collection.

Contains common part of the [Index](#) and [Subindex](#) classes.

## Object Model

[IndexDefinition<T>](#)

## Syntax

Visual Basic (Declaration)

```
<System.Diagnostics.DebuggerDisplayAttribute(Value="{ Key = {KeySelector}
\}"),
```

```

    Name="",
    Type="",
    Target=,
    TargetTypeName="")>
Public MustInherit Class IndexDefinition(Of T)

```

C#

```

[System.Diagnostics.DebuggerDisplay(Value="{ Key = {KeySelector} }"),
    Name="",
    Type="",
    Target=,
    TargetTypeName="")]
public abstract class IndexDefinition<T>

```

## Type Parameters

*T*

The type of the elements of indexed collection.

## Remarks

This class serves as the base class of two classes: [Index](#) and [Subindex](#). It contains properties common to these two classes.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Indexing.IndexDefinition<T>**

[C1.LiveLinq.Indexing.Index<T>](#)

[C1.LiveLinq.Indexing.Subindex<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexDefinition<T> Members](#)

[C1.LiveLinq.Indexing Namespace](#)

[Index<T,TKey> Class](#)

## Overview

[C1.LiveLinq.Indexing Namespace](#) : [IndexDefinition<T>](#) Class

The type of the elements of indexed collection.

Contains common part of the [Index](#) and [Subindex](#) classes.

## Object Model

[IndexDefinition<T>](#)

## Syntax

Visual Basic (Declaration)

```
<System.Diagnostics.DebuggerDisplayAttribute(Value="\{ Key = {KeySelector} \}",  
Name="",  
Type="",  
Target=,  
TargetTypeName="")>  
Public MustInherit Class IndexDefinition(Of T)
```

C#

```
[System.Diagnostics.DebuggerDisplay(Value="\{ Key = {KeySelector} \}",  
Name="",  
Type="",  
Target=,  
TargetTypeName="")]  
public abstract class IndexDefinition<T>
```

## Type Parameters

*T*

The type of the elements of indexed collection.

## Remarks

This class serves as the base class of two classes: [Index](#) and [Subindex](#). It contains properties common to these two classes.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Indexing.IndexDefinition<T>**

[C1.LiveLinq.Indexing.Index<T>](#)

[C1.LiveLinq.Indexing.Subindex<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexDefinition<T> Members](#)

[C1.LiveLinq.Indexing Namespace](#)

[Index<T,TKey> Class](#)







### Members


[Properties](#)

[C1.LiveLinq.Indexing Namespace](#) : [IndexDefinition<T> Class](#)

The following tables list the members exposed by [IndexDefinition<T>](#).

### Public Properties

	Name	Description
	<a href="#">Algorithm</a>	Gets the indexing algorithm used by the index.
	<a href="#">KeyIsUnique</a>	Gets a value that indicates whether the key used in this index is a unique key for the collection.
	<a href="#">KeySelector</a>	Gets the expression used to obtain key value from an element of the indexed collection.
	<a href="#">KeyType</a>	Gets the type of the index key.
	<a href="#">Locale</a>	Gets the locale information used to compare strings in the index.
	<a href="#">Root</a>	Gets the root index in an index/subindex hierarchy.

	<a href="#">Subindexes</a>	Gets the collection of subindexes added to this index.
---	----------------------------	--

[Top](#)

## See Also

### Reference

[IndexDefinition<T> Class](#)

[C1.LiveLinq.Indexing Namespace](#)








[Index<T,TKey> Class](#)

## Properties

[C1.LiveLinq.Indexing Namespace](#) : [IndexDefinition<T> Class](#)

For a list of all members of this type, see [IndexDefinition<T> members](#).

## Public Properties

	Name	Description
	<a href="#">Algorithm</a>	Gets the indexing algorithm used by the index.
	<a href="#">KeyIsUnique</a>	Gets a value that indicates whether the key used in this index is a unique key for the collection.
	<a href="#">KeySelector</a>	Gets the expression used to obtain key value from an element of the indexed collection.
	<a href="#">KeyType</a>	Gets the type of the index key.
	<a href="#">Locale</a>	Gets the locale information used to compare strings in the index.
	<a href="#">Root</a>	Gets the root index in an index/subindex hierarchy.
	<a href="#">Subindexes</a>	Gets the collection of subindexes added to this index.

[Top](#)

## See Also

### Reference

[IndexDefinition<T> Class](#)  
[C1.LiveLinq.Indexing Namespace](#)  
[Index<T,TKey> Class](#)

## Algorithm Property

[C1.LiveLinq.Indexing Namespace](#) > [IndexDefinition<T> Class](#) : Algorithm Property

Gets the indexing algorithm used by the index.

## Syntax

Visual Basic (Declaration)	
<code>Public MustOverride ReadOnly Property Algorithm As IndexingAlgorithm</code>	
C#	
<code>public abstract IndexingAlgorithm Algorithm {get;}</code>	

### Property Value

An [IndexingAlgorithm](#) used by the index. In the current version, only one algorithm is supported, RedBlackTree. Later versions may support other algorithms, such as bitmap or hash indexes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexDefinition<T> Class](#)  
[IndexDefinition<T> Members](#)

## KeysIsUnique Property

[C1.LiveLinq.Indexing Namespace](#) > [IndexDefinition<T> Class](#) : KeysIsUnique Property

Gets a value that indicates whether the key used in this index is a unique key for the collection.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<b>Public ReadOnly Property</b> KeyIsUnique <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool KeyIsUnique { <b>get</b> ;}	

## Property Value

**true** if the key is unique; otherwise, **false**

## Remarks

A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique in the [IndexCollection.Add](#) method.

But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the collection. An attempt to modify the collection violating the uniqueness throws an **System.InvalidOperationException**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexDefinition<T> Class](#)

[IndexDefinition<T> Members](#)

### KeySelector Property

[C1.LiveLinq.Indexing Namespace](#) > [IndexDefinition<T> Class](#) : KeySelector Property

Gets the expression used to obtain key value from an element of the indexed collection.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> KeySelector <b>As</b> System.Linq.Expressions.LambdaExpression	
C#	
<b>public</b> System.Linq.Expressions.LambdaExpression KeySelector { <b>get</b> ;}	

## Property Value

An expression calculating the key value from an item (element of the collection). Typically, this is a field or a property in the item class, although more complex expressions can also be used.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexDefinition<T> Class](#)

[IndexDefinition<T> Members](#)

### KeyType Property

[C1.LiveLinq.Indexing Namespace](#) > [IndexDefinition<T> Class](#) : KeyType Property

Gets the type of the index key.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> KeyType <b>As</b> System.Type	
C#	
<b>public</b> System.Type KeyType { <b>get</b> ;}	

### Property Value

The type of the result of the [KeySelector](#) expression.

## Remarks

This type is the same as the TKey type parameter for [Index<T,TKey>](#) and [Subindex<T,TKey>](#) objects.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2



## See Also

### Reference

[IndexDefinition<T> Class](#)

[IndexDefinition<T> Members](#)

### Locale Property

[C1.LiveLinq.Indexing Namespace](#) > [IndexDefinition<T> Class](#) : Locale Property

Gets the locale information used to compare strings in the index.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Locale As System.Globalization.CultureInfo</code>	
C#	
<code>public System.Globalization.CultureInfo Locale {get;}</code>	

### Property Value

A **System.Globalization.CultureInfo** that contains data about the user's locale. The default is **System.Globalization.CultureInfo.CurrentCulture**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexDefinition<T> Class](#)

[IndexDefinition<T> Members](#)

### Root Property

[C1.LiveLinq.Indexing Namespace](#) > [IndexDefinition<T> Class](#) : Root Property

Gets the root index in an index/subindex hierarchy.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Root As IndexDefinition(Of T)</code>	
C#	
<code>public IndexDefinition&lt;T&gt; Root {get;}</code>	

### Property Value

For a subindex, this is an [Index<T>](#) to which the subindex belongs (maybe indirectly, through several levels of parent subindexes). For an index, this is always the same object as the index itself.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexDefinition<T> Class](#)

[IndexDefinition<T> Members](#)

### Subindexes Property

[C1.LiveLinq.Indexing Namespace](#) > [IndexDefinition<T> Class](#) : Subindexes Property

Gets the collection of subindexes added to this index.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Subindexes As SubindexCollection(Of T)</code>	
C#	
<code>public SubindexCollection&lt;T&gt; Subindexes {get;}</code>	

### Property Value

The [SubindexCollection<T>](#) that contains the subindexes of this index.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexDefinition<T> Class](#)  
[IndexDefinition<T> Members](#)

## IndexingAlgorithm

[C1.LiveLinq.Indexing Namespace](#) : IndexingAlgorithm Class

Defines the kind of an index, the algorithm used by that index. Currently, the RedBlackTree algorithm is always used.

## Object Model

IndexingAlgorithm

## Syntax

Visual Basic (Declaration)	
<code>Public MustInherit Class IndexingAlgorithm</code>	
C#	
<code>public abstract class IndexingAlgorithm</code>	

## Remarks

In the current version, only one algorithm is supported, RedBlackTree. Later versions may support other algorithms, such as bitmap or hash indexing.

## Inheritance Hierarchy

System.Object  
**C1.LiveLinq.Indexing.IndexingAlgorithm**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[IndexingAlgorithm Members](#)  
[C1.LiveLinq.Indexing Namespace](#)

## Overview

[C1.LiveLinq.Indexing Namespace](#) : IndexingAlgorithm Class

Defines the kind of an index, the algorithm used by that index. Currently, the RedBlackTree algorithm is always used.

# Object Model

IndexingAlgorithm

## Syntax

Visual Basic (Declaration)	
<code>Public MustInherit Class IndexingAlgorithm</code>	
C#	
<code>public abstract class IndexingAlgorithm</code>	

## Remarks

In the current version, only one algorithm is supported, RedBlackTree. Later versions may support other algorithms, such as bitmap or hash indexing.

## Inheritance Hierarchy

System.Object  
**C1.LiveLinq.Indexing.IndexingAlgorithm**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference


## Members

[Fields](#) [Methods](#)

[C1.LiveLinq.Indexing Namespace](#) : IndexingAlgorithm Class


The following tables list the members exposed by [IndexingAlgorithm](#).

### Public Fields

	Name	Description
 <b>S</b>	<a href="#">RedBlackTree</a>	The red-black tree algorithm, a type of self-balancing binary search tree widely used in computer science because it has good performance that does not significantly degrade even in worst cases.

[Top](#)

### Public Methods

	Name	Description
	<a href="#">CreateIndex&lt;T,TKey&gt;</a>	Creates a new index.

[Top](#)

## See Also

### Reference

[IndexingAlgorithm Class](#)  
[C1.LiveLinq.Indexing Namespace](#)


## Methods

[C1.LiveLinq.Indexing Namespace](#) : IndexingAlgorithm Class

For a list of all members of this type, see [IndexingAlgorithm members](#).

### Public Methods

	Name	Description
--	------	-------------

	<code>CreateIndex&lt;T,TKey&gt;</code>	Creates a new index.
---	--	----------------------

[Top](#)

## See Also

### Reference

[IndexingAlgorithm Class](#)

[C1.LiveLinq.Indexing Namespace](#)

### CreateIndex<T,TKey> Method

[C1.LiveLinq.Indexing Namespace](#) > [IndexingAlgorithm Class](#) :

CreateIndex<T,TKey>(IObservableSource<T>,Expression<Func<T,TKey>>,Boolean,CultureInfo) Method

The type of the elements of the collection to index.

The type of the index key.

The collection to be indexed.

Key selector expression of the index, see [KeySelector](#).

Specifies whether the key used in this index is a unique key for the indexed collection.

Locale information used to compare strings in the index (default: **CultureInfo.CurrentCulture**).

Creates a new index.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function CreateIndex     (Of T,TKey)( _     ByVal source As IObservableSource(Of T), _     ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of T,TKey)), _     ByVal keyIsUnique As System.Boolean, _     ByVal locale As System.Globalization.CultureInfo _ ) As Index(Of T,TKey)</pre>	
C#	
<pre>public Index&lt;T,TKey&gt; CreateIndex&lt;T,TKey&gt;(     IObservableSource&lt;T&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;T,TKey&gt;&gt; keySelector,</pre>	

```
System.bool keyIsUnique,  
System.Globalization.CultureInfo locale  
)
```

## Parameters

*source*

The collection to be indexed.

*keySelector*

Key selector expression of the index, see [KeySelector](#).

*keyIsUnique*

Specifies whether the key used in this index is a unique key for the indexed collection.

*locale*

Locale information used to compare strings in the index (default: **CultureInfo.CurrentCulture**).

## Type Parameters

*T*

The type of the elements of the collection to index.

*TKey*

The type of the index key.

## Return Value

The new index.

## Remarks

Normally, you don't need to use this method. Indexes are usually created in code by calling [IndexCollection.Add](#), or their creation can be enforced in LINQ queries by using the [Indexed](#) hint. This method should be used only in special situations where all you want is to index a collection without an overhead of maintaining a collection of indexes, or you need an index that exists separately from the collection of indexes maintained by the source. Unlike the standard ways of creating indexes, this method only creates an index object and attaches it to the source (so it will be automatically synchronized with the source when the source is modified), but the created index is not added to [IndexCollection<T>](#).

A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique. But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the indexed collection. An attempt to modify the indexed collection violating the uniqueness throws an **System.InvalidOperationException**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexingAlgorithm Class](#)


[IndexingAlgorithm Members](#)

### Fields

[C1.LiveLinq.Indexing Namespace](#) : IndexingAlgorithm Class

For a list of all members of this type, see [IndexingAlgorithm members](#).

## Public Fields

	Name	Description
	<a href="#">RedBlackTree</a>	The red-black tree algorithm, a type of self-balancing binary search tree widely used in computer science because it has good performance that does not significantly degrade even in worst cases.

[Top](#)

## See Also

### Reference

[IndexingAlgorithm Class](#)

[C1.LiveLinq.Indexing Namespace](#)



## RedBlackTree Field

[C1.LiveLinq.Indexing Namespace](#) > [IndexingAlgorithm Class](#) : RedBlackTree Field

The red-black tree algorithm, a type of self-balancing binary search tree widely used in computer science because it has good performance that does not significantly degrade even in worst cases.

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared ReadOnly RedBlackTree As IndexingAlgorithm</pre>	
C#	
<pre>public static readonly IndexingAlgorithm RedBlackTree</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexingAlgorithm Class](#)

[IndexingAlgorithm Members](#)

## IndexingException

[C1.LiveLinq.Indexing Namespace](#) : IndexingException Class

Represents an exception that is thrown when errors are generated using LiveLinq components.

## Object Model

IndexingException

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.SerializableAttribute(&gt;&gt; Public Class IndexingException</pre>	

<b>Inherits</b> System.Exception	
C#	
[System.Serializable()] <b>public class</b> IndexingException : System.Exception	

## Inheritance Hierarchy

System.Object  
System.Exception  
**C1.LiveLinq.Indexing.IndexingException**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexingException Members](#)  
[C1.LiveLinq.Indexing Namespace](#)

## Overview

[C1.LiveLinq.Indexing Namespace](#) : IndexingException Class

Represents an exception that is thrown when errors are generated using LiveLinq components.

## Object Model

IndexingException

## Syntax

Visual Basic (Declaration)	
<System.SerializableAttribute(>> <b>Public Class</b> IndexingException <b>Inherits</b> System.Exception	
C#	
[System.Serializable()]	

```
public class IndexingException : System.Exception
```

## Inheritance Hierarchy

System.Object

System.Exception

**C1.LiveLinq.Indexing.IndexingException**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexingException Members](#)

[C1.LiveLinq.Indexing Namespace](#)


## Members

[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.Indexing Namespace](#) : [IndexingException Class](#)



The following tables list the members exposed by [IndexingException](#).







## Public Constructors

	Name	Description
	<a href="#">IndexingException Constructor</a>	Overloaded.

[Top](#)





## Public Properties

	Name	Description
	<a href="#">Data</a>	(Inherited from System.Exception)
	<a href="#">HelpLink</a>	(Inherited from System.Exception)

 <a href="#">HResult</a>	(Inherited from System.Exception)
 <a href="#">InnerException</a>	(Inherited from System.Exception)
 <a href="#">Message</a>	(Inherited from System.Exception)
 <a href="#">Source</a>	(Inherited from System.Exception)
 <a href="#">StackTrace</a>	(Inherited from System.Exception)
 <a href="#">TargetSite</a>	(Inherited from System.Exception)


[Top](#)

## Public Methods

	Name	Description
	<a href="#">GetBaseException</a>	(Inherited from System.Exception)
	<a href="#">GetObjectData</a>	(Inherited from System.Exception)
	<a href="#">GetType</a>	(Inherited from System.Exception)
	<a href="#">ToString</a>	(Inherited from System.Exception)

[Top](#)

## Protected Events

	Name	Description
	<a href="#">SerializeObjectState</a>	(Inherited from System.Exception)

[Top](#)

## See Also

### Reference

[IndexingException Class](#)

[C1.LiveLinq.Indexing Namespace](#)

## IndexingException Constructor

[C1.LiveLinq.Indexing Namespace](#) > [IndexingException Class](#) : IndexingException Constructor

### Overload List

Overload	Description
<a href="#">IndexingException Constructor()</a>	Initializes a new instance of the <a href="#">IndexingException</a> class. This is the default constructor.
<a href="#">IndexingException Constructor(String)</a>	Initializes a new instance of the <a href="#">IndexingException</a> class with the specified string.
<a href="#">IndexingException Constructor(String,Exception)</a>	Initializes a new instance of the <a href="#">IndexingException</a> class with the specified string and inner exception.
<a href="#">IndexingException Constructor(SerializationInfo,StreamingContext)</a>	Initializes a new instance of the <a href="#">IndexingException</a> class with the specified serialization information and context.

### Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

### See Also

#### Reference

[IndexingException Class](#)

[IndexingException Members](#)

## IndexingException Constructor()

[C1.LiveLinq.Indexing Namespace](#) > [IndexingException Class](#) > [IndexingException Constructor](#) :

IndexingException Constructor()

Initializes a new instance of the [IndexingException](#) class. This is the default constructor.

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public IndexingException()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexingException Class](#)

[IndexingException Members](#)

[Overload List](#)

## IndexingException Constructor(String)

[C1.LiveLinq.Indexing Namespace](#) > [IndexingException Class](#) > [IndexingException Constructor](#) :

IndexingException Constructor(String)

The string to display when the exception is thrown.

Initializes a new instance of the [IndexingException](#) class with the specified string.

## Syntax

Visual Basic (Declaration)	
<code>Public Function New( _     ByVal message As System.String _ )</code>	
C#	

```
public IndexingException(
    System.string message
)
```

## Parameters

*message*

The string to display when the exception is thrown.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexingException Class](#)  
[IndexingException Members](#)  
[Overload List](#)

## IndexingException Constructor(String,Exception)

[C1.LiveLinq.Indexing Namespace](#) > [IndexingException Class](#) > [IndexingException Constructor](#) :  
 IndexingException Constructor(String,Exception)

The string to display when the exception is thrown.

A reference to an inner exception.

Initializes a new instance of the [IndexingException](#) class with the specified string and inner exception.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function New( _     ByVal message As System.String, _     ByVal inner As System.Exception _ )</pre>	
C#	
<pre>public IndexingException(</pre>	

```
System.string message,  
System.Exception inner  
)
```

## Parameters

*message*

The string to display when the exception is thrown.

*inner*

A reference to an inner exception.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IndexingException Class](#)

[IndexingException Members](#)

[Overload List](#)

## IndexingException Constructor(SerializationInfo,StreamingContext)

[C1.LiveLinq.Indexing Namespace](#) > [IndexingException Class](#) > [IndexingException Constructor](#) :

IndexingException Constructor(SerializationInfo,StreamingContext)

The data necessary to serialize or deserialize an object.

Description of the source and destination of the specified serialized stream.

Initializes a new instance of the [IndexingException](#) class with the specified serialization information and context.

## Syntax

Visual Basic (Declaration)

```
Protected Function New( _  
    ByVal info As System.Runtime.Serialization.SerializationInfo, _  
    ByVal context As System.Runtime.Serialization.StreamingContext _  
)
```



C#

```
protected IndexingException(  
    System.Runtime.Serialization.SerializationInfo info,  
    System.Runtime.Serialization.StreamingContext context  
)
```

### Parameters

*info*

The data necessary to serialize or deserialize an object.

*context*

Description of the source and destination of the specified serialized stream.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexingException Class](#)  
[IndexingException Members](#)  
[Overload List](#)

## ScannerCollection<T>

[C1.LiveLinq.Indexing Namespace](#) : ScannerCollection<T> Class

The type of the elements of the indexed collection.

Represents a collection of indexes or subindexes.

## Object Model

ScannerCollection<T>

## Syntax

Visual Basic (Declaration)

```
Public MustInherit Class ScannerCollection(Of T)
```

C#

```
public abstract class ScannerCollection<T>
```

## Type Parameters

*T*

The type of the elements of the indexed collection.

## Remarks

Any indexed collection (implementing the [IIndexedSource<T>](#) interface) has a collection of indexes attached to it. **ScannerCollection<T>** is the base class for the collection of indexes, [IndexCollection<T>](#)

The [ScannerCollection<T>](#) class is also used in the [Indexes](#) property of the [C1.LiveLinq.Indexing.Search.IndexQuery<T>](#) class, the result of an indexing search operation such as [Index.Find](#) and others, where it contains subindexes.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Indexing.ScannerCollection<T>**

[C1.LiveLinq.Indexing.IndexCollection<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ScannerCollection<T> Members](#)

[C1.LiveLinq.Indexing Namespace](#)

### Overview

[C1.LiveLinq.Indexing Namespace](#) : [ScannerCollection<T>](#) Class

The type of the elements of the indexed collection.

Represents a collection of indexes or subindexes.

## Object Model

## Syntax

Visual Basic (Declaration)

```
Public MustInherit Class ScannerCollection(Of T)
```

C#

```
public abstract class ScannerCollection<T>
```

## Type Parameters

*T*

The type of the elements of the indexed collection.

## Remarks

Any indexed collection (implementing the [IIndexedSource<T>](#) interface) has a collection of indexes attached to it. **ScannerCollection<T>** is the base class for the collection of indexes, [IndexCollection<T>](#)

The [ScannerCollection<T>](#) class is also used in the [Indexes](#) property of the [C1.LiveLinq.Indexing.Search.IndexQuery<T>](#) class, the result of an indexing search operation such as [Index.Find](#) and others, where it contains subindexes.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Indexing.ScannerCollection<T>**

[C1.LiveLinq.Indexing.IndexCollection<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ScannerCollection<T> Members](#)

[C1.LiveLinq.Indexing Namespace](#)


## Members

[Properties](#) [Methods](#)

[C1.LiveLinq.Indexing Namespace](#) : [ScannerCollection<T>](#) Class







The following tables list the members exposed by [ScannerCollection<T>](#).

### Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the number of indexes in the collection.

[Top](#)

### Public Methods

	Name	Description
	<a href="#">Add</a>	Overloaded. Creates a new index and adds it to the collection of indexes.
	<a href="#">Clear</a>	Clears the collection of all indexes. All indexes are detached from the indexed collection and destroyed.
	<a href="#">Contains</a>	Overloaded. Determines whether an index with the specified key selector exists in the collection.
	<a href="#">Find</a>	Overloaded. Finds an index in the collection by its key selector.
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">ScannerCollection&lt;T&gt;</a> .
	<a href="#">Remove</a>	Overloaded. Removes an index from the collection.

[Top](#)

## See Also

### Reference

[ScannerCollection<T>](#) Class

[C1.LiveLinq.Indexing Namespace](#)

## Methods

[C1.LiveLinq.Indexing Namespace](#) : [ScannerCollection<T> Class](#)

For a list of all members of this type, see [ScannerCollection<T> members](#).

## Public Methods

	Name	Description
≡💎	<a href="#">Add</a>	Overloaded. Creates a new index and adds it to the collection of indexes.
≡💎	<a href="#">Clear</a>	Clears the collection of all indexes. All indexes are detached from the indexed collection and destroyed.
≡💎	<a href="#">Contains</a>	Overloaded. Determines whether an index with the specified key selector exists in the collection.
≡💎	<a href="#">Find</a>	Overloaded. Finds an index in the collection by its key selector.
≡💎	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">ScannerCollection&lt;T&gt;</a> .
≡💎	<a href="#">Remove</a>	Overloaded. Removes an index from the collection.

[Top](#)

## See Also

### Reference

[ScannerCollection<T> Class](#)

[C1.LiveLinq.Indexing Namespace](#)

### Add Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) : Add Method

Creates a new index and adds it to the collection of indexes.

## Overload List

Overload	Description
----------	-------------

Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean,IndexingAlgorithm,CultureInfo)	Creates a new index and adds it to the collection of indexes.
Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean,CultureInfo)	Creates a new index and adds it to the collection of indexes.
Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean)	Creates a new index and adds it to the collection of indexes.
Add<TKey>(Expression<Func<T,TKey>>,Boolean)	Creates a new index and adds it to the collection of indexes.
Add<TKey>(Expression<Func<T,TKey>>)	Creates a new index and adds it to the collection

	of indexes.
<a href="#">Add(LambdaExpression,Boolean,Boolean,IndexingAlgorithm,CultureInfo)</a>	Creates a new index and adds it to the collection of indexes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ScannerCollection<T> Class](#)

[ScannerCollection<T> Members](#)

[Add<TKey>\(Expression<Func<T,TKey>>,Boolean,Boolean,IndexingAlgorithm,CultureInfo\)](#)  
Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) > [Add Method](#) :

[Add<TKey>\(Expression<Func<T,TKey>>,Boolean,Boolean,IndexingAlgorithm,CultureInfo\)](#) Method

The type of the index key.

Key selector expression of the index, see [KeySelector](#).

Specifies whether the key used in this index is a unique key for the indexed collection (default: **false**).

Specifies whether it is required that the index does not exist prior to this method call (default: **false**). If an index with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

An [IndexingAlgorithm](#) used by the index. In the current version, only one algorithm is supported, RedBlackTree. Later versions may support other algorithms, such as bitmap or hash indexes.

Locale information used to compare strings in the index (default: **CultureInfo.CurrentCulture**).

Creates a new index and adds it to the collection of indexes.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Add(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)), _  
    ByVal keyIsUnique As System.Boolean, _  
    ByVal onlyOnce As System.Boolean, _  
    ByVal algorithm As IndexingAlgorithm, _  
    ByVal locale As System.Globalization.CultureInfo _  
) As IIndexScanner(Of T, TKey)
```

C#

```
public IIndexScanner<T, TKey> Add<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector,  
    System.bool keyIsUnique,  
    System.bool onlyOnce,  
    IndexingAlgorithm algorithm,  
    System.Globalization.CultureInfo locale  
)
```

## Parameters

*keySelector*

Key selector expression of the index, see [KeySelector](#).

*keyIsUnique*

Specifies whether the key used in this index is a unique key for the indexed collection (default: **false**).

*onlyOnce*

Specifies whether it is required that the index does not exist prior to this method call (default: **false**). If an index with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

*algorithm*

An [IndexingAlgorithm](#) used by the index. In the current version, only one algorithm is supported, RedBlackTree. Later versions may support other algorithms, such as bitmap or hash indexes.



*locale*

Locale information used to compare strings in the index (default: **CultureInfo.CurrentCulture**).

## Type Parameters

*TKey*

The type of the index key.

## Return Value

The new index added to the collection of indexes.

## Remarks

A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique.

But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the indexed collection. An attempt to modify the indexed collection violating the uniqueness throws an **System.InvalidOperationException**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ScannerCollection<T> Class](#)

[ScannerCollection<T> Members](#)

[Overload List](#)

Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean,CultureInfo) Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) > [Add Method](#) :

Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean,CultureInfo) Method

The type of the index key.

Key selector expression of the index, see [KeySelector](#).

Specifies whether the key used in this index is a unique key for the indexed collection (default: **false**).

Specifies whether it is required that the index does not exist prior to this method call (default: **false**). If an index with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

Locale information used to compare strings in the index (default: **CultureInfo.CurrentCulture**).

Creates a new index and adds it to the collection of indexes.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Add(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)), _  
    ByVal keyIsUnique As System.Boolean, _  
    ByVal onlyOnce As System.Boolean, _  
    ByVal locale As System.Globalization.CultureInfo _  
) As IIndexScanner(Of T, TKey)
```

C#

```
public IIndexScanner<T, TKey> Add<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector,  
    System.bool keyIsUnique,  
    System.bool onlyOnce,  
    System.Globalization.CultureInfo locale  
)
```

## Parameters

*keySelector*

Key selector expression of the index, see [KeySelector](#).

*keyIsUnique*

Specifies whether the key used in this index is a unique key for the indexed collection (default: **false**).

*onlyOnce*

Specifies whether it is required that the index does not exist prior to this method call (default: **false**). If an index with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

*locale*

Locale information used to compare strings in the index (default: **CultureInfo.CurrentCulture**).

## Type Parameters

*TKey*

The type of the index key.

## Return Value

The new index added to the collection of indexes.

## Remarks

A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique.

But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the indexed collection. An attempt to modify the indexed collection violating the uniqueness throws an **System.InvalidOperationException**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ScannerCollection<T> Class](#)  
[ScannerCollection<T> Members](#)  
[Overload List](#)

Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean) Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) > [Add Method](#) :

Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean) Method

The type of the index key.

Key selector expression of the index, see [KeySelector](#).

Specifies whether the key used in this index is a unique key for the indexed collection (default: **false**).

Specifies whether it is required that the index does not exist prior to this method call (default: **false**). If an index with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

Creates a new index and adds it to the collection of indexes.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Add(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T,TKey))), _  
    ByVal keyIsUnique As System.Boolean, _  
    ByVal onlyOnce As System.Boolean _  
) As IIndexScanner(Of T,TKey)
```

C#

```
public IIndexScanner<T,TKey> Add<TKey>(  
    System.Linq.Expressions.Expression<Func<T,TKey>> keySelector,  
    System.bool keyIsUnique,  
    System.bool onlyOnce  
)
```

## Parameters

*keySelector*

Key selector expression of the index, see [KeySelector](#).

*keyIsUnique*

Specifies whether the key used in this index is a unique key for the indexed collection (default: **false**).

*onlyOnce*

Specifies whether it is required that the index does not exist prior to this method call (default: **false**). If an index with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

## Type Parameters

*TKey*

The type of the index key.

## Return Value

The new index added to the collection of indexes.

## Remarks

A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique.

But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the indexed collection. An attempt to modify the indexed collection violating the uniqueness throws an **System.InvalidOperationException**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ScannerCollection<T> Class](#)

[ScannerCollection<T> Members](#)

[Overload List](#)

Add<TKey>(Expression<Func<T,TKey>>,Boolean) Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) > [Add Method](#) :

Add<TKey>(Expression<Func<T,TKey>>,Boolean) Method

The type of the index key.

Key selector expression of the index, see [KeySelector](#).

Specifies whether the key used in this index is a unique key for the indexed collection (default: **false**).

Creates a new index and adds it to the collection of indexes.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Add(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)), _  
    ByVal keyIsUnique As System.Boolean _  
) As IIndexScanner(Of T, TKey)
```

C#

```
public IIndexScanner<T, TKey> Add<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector,  
    System.bool keyIsUnique  
)
```

### Parameters

*keySelector*

Key selector expression of the index, see [KeySelector](#).

*keyIsUnique*

Specifies whether the key used in this index is a unique key for the indexed collection (default: **false**).

### Type Parameters

*TKey*

The type of the index key.

### Return Value

The new index added to the collection of indexes.

## Remarks

A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique.

But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the indexed collection. An attempt to modify the indexed collection violating the uniqueness throws an **System.InvalidOperationException**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ScannerCollection<T> Class](#)  
[ScannerCollection<T> Members](#)  
[Overload List](#)

Add<TKey>(Expression<Func<T,TKey>>) Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) > [Add Method](#) :

Add<TKey>(Expression<Func<T,TKey>>) Method

The type of the index key.

Key selector expression of the index, see [KeySelector](#).

Creates a new index and adds it to the collection of indexes.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Add(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
) As IIndexScanner(Of T, TKey)
```

C#

```
public IIndexScanner<T,TKey> Add<TKey>(  
    System.Linq.Expressions.Expression<Func<T,TKey>> keySelector  
)
```

### Parameters

*keySelector*

Key selector expression of the index, see [KeySelector](#).

## Type Parameters

*TKey*

The type of the index key.

## Return Value

The new index added to the collection of indexes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ScannerCollection<T> Class](#)

[ScannerCollection<T> Members](#)

[Overload List](#)

Add(LambdaExpression, Boolean, Boolean, IndexingAlgorithm, CultureInfo) Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) > [Add Method](#) :

Add(LambdaExpression, Boolean, Boolean, IndexingAlgorithm, CultureInfo) Method

Key selector expression of the index, see [KeySelector](#).

Specifies whether the key used in this index is a unique key for the indexed collection (default: **false**).

Specifies whether it is required that the index does not exist prior to this method call. If an index with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

An [IndexingAlgorithm](#) used by the index. In the current version, only one algorithm is supported, RedBlackTree. Later versions may support other algorithms, such as bitmap or hash indexes.

Locale information used to compare strings in the index (default: **CultureInfo.CurrentCulture**).

Creates a new index and adds it to the collection of indexes.

## Syntax



## Visual Basic (Declaration)

```
Public Overloads Function Add( _  
    ByVal keySelector As System.Linq.Expressions.LambdaExpression, _  
    ByVal keyIsUnique As System.Boolean, _  
    ByVal onlyOnce As System.Boolean, _  
    ByVal algorithm As IndexingAlgorithm, _  
    ByVal locale As System.Globalization.CultureInfo _  
) As IIndexScanner(Of T)
```

## C#

```
public IIndexScanner<T> Add(  
    System.Linq.Expressions.LambdaExpression keySelector,  
    System.bool keyIsUnique,  
    System.bool onlyOnce,  
    IndexingAlgorithm algorithm,  
    System.Globalization.CultureInfo locale  
)
```

## Parameters

*keySelector*

Key selector expression of the index, see [KeySelector](#).

*keyIsUnique*

Specifies whether the key used in this index is a unique key for the indexed collection (default: **false**).

*onlyOnce*

Specifies whether it is required that the index does not exist prior to this method call. If an index with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

*algorithm*

An [IndexingAlgorithm](#) used by the index. In the current version, only one algorithm is supported, RedBlackTree. Later versions may support other algorithms, such as bitmap or hash indexes.

*locale*

Locale information used to compare strings in the index (default: **CultureInfo.CurrentCulture**).

## Return Value

The new index added to the collection of indexes.

## Remarks

A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique.

But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the indexed collection. An attempt to modify the indexed collection violating the uniqueness throws an **System.InvalidOperationException**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ScannerCollection<T> Class](#)  
[ScannerCollection<T> Members](#)  
[Overload List](#)

### Clear Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) : Clear Method

Clears the collection of all indexes. All indexes are detached from the indexed collection and destroyed.

## Syntax

Visual Basic (Declaration)	
<b>Public MustOverride Sub</b> Clear()	
C#	
<b>public abstract void</b> Clear()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ScannerCollection<T> Class](#)

[ScannerCollection<T> Members](#)

### Contains Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) : Contains Method

Determines whether an index with the specified key selector exists in the collection.

## Overload List

Overload	Description
<a href="#">Contains(LambdaExpression)</a>	Determines whether an index with the specified key selector exists in the collection.
<a href="#">Contains&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;)</a>	Determines whether an index with the specified key selector exists in the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ScannerCollection<T> Class](#)

[ScannerCollection<T> Members](#)

### Contains(LambdaExpression) Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) > [Contains Method](#) :

Contains(LambdaExpression) Method

Key selector expression of an index, see [KeySelector](#).

Determines whether an index with the specified key selector exists in the collection.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Contains( _  
    ByVal keySelector As System.Linq.Expressions.LambdaExpression _  
) As System.Boolean
```

C#

```
public System.bool Contains(  
    System.Linq.Expressions.LambdaExpression keySelector  
)
```

### Parameters

*keySelector*

Key selector expression of an index, see [KeySelector](#).

### Return Value

**true** if an index with the specified key selector is found in the collection; otherwise, **false**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ScannerCollection<T> Class](#)

[ScannerCollection<T> Members](#)

[Overload List](#)

Contains<TKey>(Expression<Func<T,TKey>>) Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) > [Contains Method](#) :

Contains<TKey>(Expression<Func<T,TKey>>) Method

The type of the index key.

Key selector expression of an index, see [KeySelector](#).

Determines whether an index with the specified key selector exists in the collection.

## Syntax

#### Visual Basic (Declaration)

```
Public Overloads Function Contains(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
) As System.Boolean
```

#### C#

```
public System.bool Contains<TKey>(   
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector  
)
```

#### Parameters

*keySelector*

Key selector expression of an index, see [KeySelector](#).

#### Type Parameters

*TKey*

The type of the index key.

#### Return Value

**true** if an index with the specified key selector is found in the collection; otherwise, **false**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

#### Reference

[ScannerCollection<T> Class](#)  
[ScannerCollection<T> Members](#)  
[Overload List](#)

#### Find Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) : Find Method

Finds an index in the collection by its key selector.

## Overload List

Overload	Description
<a href="#">Find(LambdaExpression)</a>	Finds an index in the collection by its key selector.
<a href="#">Find&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;)</a>	Finds an index in the collection by its key selector.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ScannerCollection<T> Class](#)

[ScannerCollection<T> Members](#)

[Find\(LambdaExpression\) Method](#)

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) > [Find Method](#) : [Find\(LambdaExpression\) Method](#)

Key selector expression of an index, see [KeySelector](#).

Finds an index in the collection by its key selector.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Find( _     ByVal keySelector As System.Linq.Expressions.LambdaExpression _ ) As IIndexScanner(Of T)</pre>	
C#	
<pre>public IIndexScanner&lt;T&gt; Find(     System.Linq.Expressions.LambdaExpression keySelector )</pre>	

### Parameters

*keySelector*

Key selector expression of an index, see [KeySelector](#).

## Return Value

An index with the given key selector, if it is found; otherwise, **null**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ScannerCollection<T> Class](#)  
[ScannerCollection<T> Members](#)  
[Overload List](#)

[Find<TKey>\(Expression<Func<T,TKey>>\) Method](#)

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) > [Find Method](#) :

[Find<TKey>\(Expression<Func<T,TKey>>\) Method](#)

The type of the index key.

Key selector expression of an index, see [KeySelector](#).

Finds an index in the collection by its key selector.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Find(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
) As IIndexScanner(Of T, TKey)
```

C#

```
public IIndexScanner<T,TKey> Find<TKey>(  
    System.Linq.Expressions.Expression<Func<T,TKey>> keySelector  
)
```

## Parameters

*keySelector*

Key selector expression of an index, see [KeySelector](#).

## Type Parameters

*TKey*

The type of the index key.

## Return Value

An index with the given key selector, if it is found; otherwise, **null**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ScannerCollection<T> Class](#)  
[ScannerCollection<T> Members](#)  
[Overload List](#)

## GetEnumerator Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) : GetEnumerator Method

Returns an enumerator that iterates through the [ScannerCollection<T>](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public MustOverride Function GetEnumerator() As System.Collections.Generic.IEnumerator(Of IIndexScanner(Of T))</pre>	
C#	
<pre>public abstract System.Collections.Generic.IEnumerator&lt;IIndexScanner&lt;T&gt;&gt; GetEnumerator()</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2



## See Also

### Reference

[ScannerCollection<T> Class](#)

[ScannerCollection<T> Members](#)

### Remove Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) : Remove Method

Removes an index from the collection.

## Overload List

Overload	Description
<a href="#">Remove(LambdaExpression)</a>	Removes an index from the collection.
<a href="#">Remove&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;)</a>	Removes an index from the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ScannerCollection<T> Class](#)

[ScannerCollection<T> Members](#)

### Remove(LambdaExpression) Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) > [Remove Method](#) :

Remove(LambdaExpression) Method

Key selector expression of an index, see [KeySelector](#).

Removes an index from the collection.

## Syntax

Visual Basic (Declaration)	
<b>Public Overloads MustOverride Function</b> Remove( _	

```
ByVal keySelector As System.Linq.Expressions.LambdaExpression _
) As System.Boolean
```

C#

```
public abstract System.bool Remove(
    System.Linq.Expressions.LambdaExpression keySelector
)
```

## Parameters

*keySelector*

Key selector expression of an index, see [KeySelector](#).

## Return Value

**true** if an index has been removed; **false** if there is no index with the given key selector in the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ScannerCollection<T> Class](#)  
[ScannerCollection<T> Members](#)  
[Overload List](#)

Remove<TKey>(Expression<Func<T,TKey>>) Method

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) > [Remove Method](#) :

Remove<TKey>(Expression<Func<T,TKey>>) Method

The type of the index key.

Key selector expression of an index, see [KeySelector](#).

Removes an index from the collection.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Remove(Of TKey)( _
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of
T, TKey)) _
) As System.Boolean
```

C#

```
public System.bool Remove<TKey>(
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector
)
```

## Parameters

*keySelector*

Key selector expression of an index, see [KeySelector](#).

## Type Parameters

*TKey*

The type of the index key.

## Return Value

**true** if an index has been removed; **false** if there is no index with the given key selector in the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference


[ScannerCollection<T> Class](#)  
[ScannerCollection<T> Members](#)  
[Overload List](#)

## Properties

[C1.LiveLinq.Indexing Namespace](#) : [ScannerCollection<T> Class](#)

For a list of all members of this type, see [ScannerCollection<T> members](#).

## Public Properties

	Name	Description
	Count	Gets the number of indexes in the collection.

[Top](#)

## See Also

### Reference

[ScannerCollection<T> Class](#)

[C1.LiveLinq.Indexing Namespace](#)

### Count Property

[C1.LiveLinq.Indexing Namespace](#) > [ScannerCollection<T> Class](#) : Count Property

Gets the number of indexes in the collection.

## Syntax

Visual Basic (Declaration)	
<b>Public MustOverride ReadOnly Property</b> Count <b>As</b> System.Integer	
C#	
<b>public abstract</b> System.int Count { <b>get</b> ;}	

### Property Value

The number of indexes in the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ScannerCollection<T> Class](#)

[ScannerCollection<T> Members](#)

# Subindex<T>

C1.LiveLinq.Indexing.Namespace : Subindex<T> Class

The type of the elements of the collection to index.

Base class for the Subindex<T,TKey> class.

## Object Model

Subindex<T>

## Syntax

Visual Basic (Declaration)	
<pre>Public MustInherit Class Subindex(Of T)     Inherits IndexDefinition(Of T)</pre>	
C#	
<pre>public abstract class Subindex&lt;T&gt; : IndexDefinition&lt;T&gt;</pre>	

## Type Parameters

T

The type of the elements of the collection to index.

## Remarks

You don't typically use the Subindex<T> class directly. It provides functionality of the Subindex<T,TKey> class that does not depend on the index key type. The base class Subindex<T> is needed only if the index key type is not known, usually in general-purpose code intended for reuse with different key types.

## Inheritance Hierarchy

System.Object  
C1.LiveLinq.Indexing.IndexDefinition<T>  
    **C1.LiveLinq.Indexing.Subindex<T>**  
        C1.LiveLinq.Indexing.Subindex<T,TKey>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Subindex<T> Members](#)  
[C1.LiveLinq.Indexing Namespace](#)

### Overview

[C1.LiveLinq.Indexing Namespace](#) : [Subindex<T> Class](#)

The type of the elements of the collection to index.

Base class for the [Subindex<T,TKey>](#) class.

## Object Model

[Subindex<T>](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public MustInherit Class Subindex(Of T)     Inherits IndexDefinition(Of T)</pre>	
C#	
<pre>public abstract class Subindex&lt;T&gt; : IndexDefinition&lt;T&gt;</pre>	

## Type Parameters

*T*

The type of the elements of the collection to index.

## Remarks

You don't typically use the [Subindex<T>](#) class directly. It provides functionality of the [Subindex<T,TKey>](#) class that does not depend on the index key type. The base class [Subindex<T>](#) is needed only if the index key type is not known, usually in general-purpose code intended for reuse with different key types.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.IndexDefinition<T>](#)

**C1.LiveLinq.Indexing.Subindex<T>**

[C1.LiveLinq.Indexing.Subindex<T,TKey>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Subindex<T> Members](#)

[C1.LiveLinq.Indexing Namespace](#)





### Members





[Properties](#)

[C1.LiveLinq.Indexing Namespace](#) : [Subindex<T>](#) Class

The following tables list the members exposed by [Subindex<T>](#).

### Public Properties

	Name	Description
	<a href="#">Algorithm</a>	Gets the indexing algorithm used by the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeyIsUnique</a>	Gets a value that indicates whether the key used in this index is a unique key for the collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeySelector</a>	Gets the expression used to obtain key value from an element of the indexed collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeyType</a>	Gets the type of the index key. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )

	<a href="#">Locale</a>	Gets the locale information used to compare strings in the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Parent</a>	Parent of this subindex definition in the index/subindexes hierarchy.
	<a href="#">Root</a>	Gets the root index in an index/subindex hierarchy. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Subindexes</a>	Gets the collection of subindexes added to this index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[Subindex<T> Class](#)





[C1.LiveLinq.Indexing Namespace](#)

## Properties





[C1.LiveLinq.Indexing Namespace](#) : [Subindex<T> Class](#)

For a list of all members of this type, see [Subindex<T> members](#).

## Public Properties

	Name	Description
	<a href="#">Algorithm</a>	Gets the indexing algorithm used by the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeyIsUnique</a>	Gets a value that indicates whether the key used in this index is a unique key for the collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeySelector</a>	Gets the expression used to obtain key value from an element of the indexed collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeyType</a>	Gets the type of the index key. (Inherited from



		<a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Locale</a>	Gets the locale information used to compare strings in the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Parent</a>	Parent of this subindex definition in the index/subindexes hierarchy.
	<a href="#">Root</a>	Gets the root index in an index/subindex hierarchy. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Subindexes</a>	Gets the collection of subindexes added to this index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[Subindex<T> Class](#)

[C1.LiveLinq.Indexing Namespace](#)

### Parent Property

[C1.LiveLinq.Indexing Namespace](#) > [Subindex<T> Class](#) : Parent Property

Parent of this subindex definition in the index/subindexes hierarchy.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Parent As IndexDefinition(Of T)</code>	
C#	
<code>public IndexDefinition&lt;T&gt; Parent {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

Reference

[Subindex<T> Class](#)  
[Subindex<T> Members](#)

Subindex<T,TKey>

[C1.LiveLinq.Indexing Namespace](#) : Subindex<T,TKey> Class

The type of the elements of the collection to index.

The type of the index key.

Defines a subindex, an index definition subordinate to another index definition, its parent.

Object Model

Subindex<T,TKey>

Syntax

Visual Basic (Declaration)	
<code>Public MustInherit Class Subindex     (Of T,TKey)     Inherits Subindex(Of T)</code>	
C#	
<code>public abstract class Subindex&lt;T,TKey&gt; : Subindex&lt;T&gt;</code>	

Type Parameters

*T*

The type of the elements of the collection to index.

*TKey*

The type of the index key.

Remarks

An index ([Index<T,TKey>](#)) can have subindexes. Subindexes are optional, not required for any indexing tasks, but can provide additional optimization and help minimize memory requirements when a collection is indexed by multi-level (multi-field) keys.

Suppose we want to index a Customers table by two fields, (City, Rating), perhaps for speeding up queries like `from c in Customers where c.City == "London" && c.Rating == 1 select c`. We can do it by defining an index with key selector `c => new { c.City, c.Rating }`, that will index the table by two fields, creating a multi-field index. Such index will suffice for optimizing the query above, but it will not optimize, for example, the following query: `from c in Customers where c.City == "London" && c.Rating > 2 select c`. Also, multi-field indexes occupy more memory than necessary because they have to store repeated field values.

Subindexes provide a better alternative for optimizing multi-field searches. In the example above, we can define an index by City and create a subindex of that index, by Rating. Using subindexes becomes even more effective when, as it is often happens, we also need queries to search by additional fields, like, for example, if we need to search by ContactTitle inside a city in addition to the search by Rating inside a city: `from c in Customers where c.City == "London" && c.ContactTitle == "Owner" select c`. All we have to do now is to add a second subindex to the index by City, a subindex by ContactTitle.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.IndexDefinition<T>](#)

[C1.LiveLinq.Indexing.Subindex<T>](#)

**C1.LiveLinq.Indexing.Subindex<T,TKey>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Subindex<T,TKey> Members](#)

[C1.LiveLinq.Indexing Namespace](#)

### Overview

[C1.LiveLinq.Indexing Namespace](#) : Subindex<T,TKey> Class

The type of the elements of the collection to index.

The type of the index key.

Defines a subindex, an index definition subordinate to another index definition, its parent.

# Object Model

Subindex<T,TKey>

## Syntax

Visual Basic (Declaration)	
<pre>Public MustInherit Class Subindex     (Of T,TKey)     Inherits Subindex(Of T)</pre>	
C#	
<pre>public abstract class Subindex&lt;T,TKey&gt; : Subindex&lt;T&gt;</pre>	

## Type Parameters

*T*

The type of the elements of the collection to index.

*TKey*

The type of the index key.

## Remarks

An index ([Index<T,TKey>](#)) can have subindexes. Subindexes are optional, not required for any indexing tasks, but can provide additional optimization and help minimize memory requirements when a collection is indexed by multi-level (multi-field) keys.

Suppose we want to index a Customers table by two fields, (City, Rating), perhaps for speeding up queries like `from c in Customers where c.City == "London" && c.Rating == 1 select c` We can do it by defining an index with key selector `c => new { c.City, c.Rating }`, that will index the table by two fields, creating a multi-field index. Such index will suffice for optimizing the query above, but it will not optimize, for example, the following query: `from c in Customers where c.City == "London" && c.Rating > 2 select c` Also, multi-field indexes occupy more memory than necessary because they have to store repeated field values.

Subindexes provide a better alternative for optimizing multi-field searches. In the example above, we can define an index by City and create a subindex of that index, by Rating. Using subindexes becomes even more effective when, as it is often happens, we also need queries to search by additional fields, like, for example, if we

need to search by ContactTitle inside a city in addition to the search by Rating inside a city: `from c in Customers where c.City == "London" && c.ContactTitle == "Owner" select c` All we have to do now is to add a second subindex to the index by City, a subindex by ContactTitle.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.IndexDefinition<T>](#)

[C1.LiveLinq.Indexing.Subindex<T>](#)

**C1.LiveLinq.Indexing.Subindex<T,TKey>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Subindex<T,TKey> Members](#)

[C1.LiveLinq.Indexing Namespace](#)




## Members






[Properties](#)

[C1.LiveLinq.Indexing Namespace](#) : [Subindex<T,TKey>](#) Class

The following tables list the members exposed by [Subindex<T,TKey>](#).

## Public Properties

	Name	Description
	<a href="#">Algorithm</a>	Gets the indexing algorithm used by the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeyIsUnique</a>	Gets a value that indicates whether the key used in this index is a unique key for the collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeySelector</a>	Gets the expression used to obtain key value from an element of the

		indexed collection.
	<a href="#">KeyType</a>	Gets the type of the index key. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Locale</a>	Gets the locale information used to compare strings in the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Parent</a>	Parent of this subindex definition in the index/subindexes hierarchy. (Inherited from <a href="#">C1.LiveLinq.Indexing.Subindex&lt;T&gt;</a> )
	<a href="#">Root</a>	Gets the root index in an index/subindex hierarchy. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Subindexes</a>	Gets the collection of subindexes added to this index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference



[Subindex<T,TKey> Class](#)  
[C1.LiveLinq.Indexing Namespace](#)







## Properties

[C1.LiveLinq.Indexing Namespace](#) : [Subindex<T,TKey> Class](#)

For a list of all members of this type, see [Subindex<T,TKey> members](#).

## Public Properties

	Name	Description
	<a href="#">Algorithm</a>	Gets the indexing algorithm used by the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">KeyIsUnique</a>	Gets a value that indicates whether the key used in this index is a unique key for the collection. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )

	<a href="#">KeySelector</a>	Gets the expression used to obtain key value from an element of the indexed collection.
	<a href="#">KeyType</a>	Gets the type of the index key. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Locale</a>	Gets the locale information used to compare strings in the index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Parent</a>	Parent of this subindex definition in the index/subindexes hierarchy. (Inherited from <a href="#">C1.LiveLinq.Indexing.Subindex&lt;T&gt;</a> )
	<a href="#">Root</a>	Gets the root index in an index/subindex hierarchy. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )
	<a href="#">Subindexes</a>	Gets the collection of subindexes added to this index. (Inherited from <a href="#">C1.LiveLinq.Indexing.IndexDefinition&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[Subindex<T,TKey> Class](#)  
[C1.LiveLinq.Indexing Namespace](#)

### KeySelector Property

[C1.LiveLinq.Indexing Namespace](#) > [Subindex<T,TKey> Class](#) : KeySelector Property

Gets the expression used to obtain key value from an element of the indexed collection.

## Syntax

Visual Basic (Declaration)	
<pre><b>Public Shadows ReadOnly Property</b> KeySelector <b>As</b> System.Linq.Expressions.Expression(Of Func(Of T,TKey))</pre>	
C#	
<pre><b>public new</b> System.Linq.Expressions.Expression&lt;Func&lt;T,TKey&gt;&gt; KeySelector {<b>get</b>;}</pre>	

## Property Value

An expression calculating the key value from an item (element of the collection). Typically, this is a field or a property in the item class, although more complex expressions can also be used.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[Subindex<T,TKey> Class](#)

[Subindex<T,TKey> Members](#)

## SubindexCollection<T>

[C1.LiveLinq.Indexing Namespace](#) : SubindexCollection<T> Class

The type of the elements of the collection to index.

Represents a collection of subindexes attached to an [IndexDefinition<T>](#).

## Object Model

**SubindexCollection<T>**

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>  
Public NotInheritable Class SubindexCollection(Of T)
```

C#

```
[System.Reflection.DefaultMember("Item")]  
public sealed class SubindexCollection<T>
```

## Type Parameters

*T*

The type of the elements of the collection to index.



## Remarks

A [Subindex<T,TKey>](#) is attached to its parent index definition, which is an [Index<T,TKey>](#) or another [Subindex<T,TKey>](#). The subindexes collection is stored in the parent's [IndexDefinition<T>.Subindexes](#) property.

Subindexes are optional, not required for any indexing tasks, but can provide additional optimization and help minimize memory requirements when a collection is indexed by multi-level (multi-field) keys.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Indexing.SubindexCollection<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Members](#)

[C1.LiveLinq.Indexing Namespace](#)

### Overview

[C1.LiveLinq.Indexing Namespace](#) : [SubindexCollection<T>](#) Class

The type of the elements of the collection to index.

Represents a collection of subindexes attached to an [IndexDefinition<T>](#).

## Object Model

[SubindexCollection<T>](#)

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>
Public NotInheritable Class SubindexCollection(Of T)
```

C#

```
[System.Reflection.DefaultMember("Item")]  
public sealed class SubindexCollection<T>
```

## Type Parameters

*T*

The type of the elements of the collection to index.

## Remarks

A [Subindex<T,TKey>](#) is attached to its parent index definition, which is an [Index<T,TKey>](#) or another [Subindex<T,TKey>](#). The subindexes collection is stored in the parent's [IndexDefinition<T>.Subindexes](#) property.

Subindexes are optional, not required for any indexing tasks, but can provide additional optimization and help minimize memory requirements when a collection is indexed by multi-level (multi-field) keys.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Indexing.SubindexCollection<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Members](#)  
[C1.LiveLinq.Indexing Namespace](#)



## Members

[Properties](#) [Methods](#)

[C1.LiveLinq.Indexing Namespace](#) : [SubindexCollection<T>](#) Class







The following tables list the members exposed by [SubindexCollection<T>](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the number of subindexes in the collection.
	<a href="#">Item</a>	Gets the subindex object at the specified ordinal position in the collection.

[Top](#)

## Public Methods

	Name	Description
	<a href="#">Add</a>	Overloaded. Creates a new subindex and attaches it to its parent's <a href="#">IndexDefinition&lt;T&gt;.Subindexes</a> collection.
	<a href="#">Clear</a>	Clears the collection of all subindexes. All subindexes are detached from the parent and destroyed.
	<a href="#">Contains</a>	Overloaded. Determines whether a subindex with the specified key selector exists in the collection.
	<a href="#">Find</a>	Overloaded. Finds a subindex in the collection by its key selector.
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">SubindexCollection&lt;T&gt;</a> .
	<a href="#">Remove</a>	Overloaded. Removes a subindex from the collection.

[Top](#)

## See Also

### Reference

[SubindexCollection<T> Class](#)  
[C1.LiveLinq.Indexing Namespace](#)

## Methods

[C1.LiveLinq.Indexing Namespace](#) : [SubindexCollection<T> Class](#)

For a list of all members of this type, see [SubindexCollection<T> members](#).

## Public Methods

	Name	Description
⇒	<a href="#">Add</a>	Overloaded. Creates a new subindex and attaches it to its parent's <a href="#">IndexDefinition&lt;T&gt;.Subindexes</a> collection.
⇒	<a href="#">Clear</a>	Clears the collection of all subindexes. All subindexes are detached from the parent and destroyed.
⇒	<a href="#">Contains</a>	Overloaded. Determines whether a subindex with the specified key selector exists in the collection.
⇒	<a href="#">Find</a>	Overloaded. Finds a subindex in the collection by its key selector.
⇒	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">SubindexCollection&lt;T&gt;</a> .
⇒	<a href="#">Remove</a>	Overloaded. Removes a subindex from the collection.

[Top](#)

## See Also

### Reference

[SubindexCollection<T> Class](#)  
[C1.LiveLinq.Indexing Namespace](#)

### Add Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) : Add Method

Creates a new subindex and attaches it to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Overload List

Overload	Description
<a href="#">Add&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Boolean,Boolean,IndexingAlgorithm,CultureInfo)</a>	Creates a new subindex and attaches it to its parent's <a href="#">IndexDefinition&lt;T&gt;.Subi</a>

	<a href="#">ndexes</a> collection.
<a href="#">Add&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Boolean,Boolean,CultureInfo)</a>	Creates a new subindex and attaches it to its parent's <a href="#">IndexDefinition&lt;T&gt;.Subindexes</a> collection.
<a href="#">Add&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Boolean,Boolean)</a>	Creates a new subindex and attaches it to its parent's <a href="#">IndexDefinition&lt;T&gt;.Subindexes</a> collection.
<a href="#">Add&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Boolean)</a>	Creates a new subindex and attaches it to its parent's <a href="#">IndexDefinition&lt;T&gt;.Subindexes</a> collection.
<a href="#">Add&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;)</a>	Creates a new subindex and attaches it to its parent's <a href="#">IndexDefinition&lt;T&gt;.Subindexes</a> collection.
<a href="#">Add(LambdaExpression,Boolean,Boolean,IndexingAlgorithm,CultureInfo)</a>	Creates a new subindex and attaches it to its parent's <a href="#">IndexDefinition&lt;T&gt;.Subindexes</a> collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[SubindexCollection<T> Class](#)

[SubindexCollection<T> Members](#)

Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean,IndexingAlgorithm,CultureInfo)  
Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) > [Add Method](#) :

Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean,IndexingAlgorithm,CultureInfo) Method

The type of the subindex key.

Key selector expression of the subindex, see [KeySelector](#).

Specifies whether the key used in this subindex is unique for any given value of the parent key (default: **false**).

Specifies whether it is required that the subindex does not exist prior to this method call (default: **false**). If a subindex with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

An [IndexingAlgorithm](#) used by the subindex. In the current version, only one algorithm is supported, RedBlackTree. Later versions may support other algorithms, such as bitmap or hash indexing.

Locale information used to compare strings in the subindex (default: **CultureInfo.CurrentCulture**).

Creates a new subindex and attaches it to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Add(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T,TKey))), _  
    ByVal keyIsUnique As System.Boolean, _  
    ByVal onlyOnce As System.Boolean, _  
    ByVal algorithm As IndexingAlgorithm, _  
    ByVal locale As System.Globalization.CultureInfo _  
) As Subindex(Of T,TKey)
```

C#

```
public Subindex<T,TKey> Add<TKey>(  
    System.Linq.Expressions.Expression<Func<T,TKey>> keySelector,
```

```
System.bool keyIsUnique,  
System.bool onlyOnce,  
IndexingAlgorithm algorithm,  
System.Globalization.CultureInfo locale  
)
```

## Parameters

*keySelector*

Key selector expression of the subindex, see [KeySelector](#).

*keyIsUnique*

Specifies whether the key used in this subindex is unique for any given value of the parent key (default: **false**).

*onlyOnce*

Specifies whether it is required that the subindex does not exist prior to this method call (default: **false**). If a subindex with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

*algorithm*

An [IndexingAlgorithm](#) used by the subindex. In the current version, only one algorithm is supported, `RedBlackTree`. Later versions may support other algorithms, such as `bitmap` or `hash indexing`.

*locale*

Locale information used to compare strings in the subindex (default: **CultureInfo.CurrentCulture**).

## Type Parameters

*TKey*

The type of the subindex key.

## Return Value

The new subindex added to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Remarks

A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique.

But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the indexed collection. An attempt to modify the indexed collection violating the uniqueness throws an **System.InvalidOperationException**.

For a subindex, uniqueness means that any given pair of parent key and subindex key values uniquely determines an item in the indexed collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)  
[SubindexCollection<T> Members](#)  
[Overload List](#)

Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean,CultureInfo) Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) > [Add Method](#) :

Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean,CultureInfo) Method

The type of the subindex key.

Key selector expression of the subindex, see [KeySelector](#).

Specifies whether the key used in this subindex is unique for any given value of the parent key (default: **false**).

Specifies whether it is required that the subindex does not exist prior to this method call (default: **false**). If a subindex with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

Locale information used to compare strings in the subindex (default: **CultureInfo.CurrentCulture**).

Creates a new subindex and attaches it to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Syntax

Visual Basic (Declaration)	
----------------------------	--



```
Public Overloads Function Add(Of TKey)( _
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of
T,TKey)), _
    ByVal keyIsUnique As System.Boolean, _
    ByVal onlyOnce As System.Boolean, _
    ByVal locale As System.Globalization.CultureInfo _
) As Subindex(Of T,TKey)
```

C#

```
public Subindex<T,TKey> Add<TKey>(
    System.Linq.Expressions.Expression<Func<T,TKey>> keySelector,
    System.bool keyIsUnique,
    System.bool onlyOnce,
    System.Globalization.CultureInfo locale
)
```

## Parameters

*keySelector*

Key selector expression of the subindex, see [KeySelector](#).

*keyIsUnique*

Specifies whether the key used in this subindex is unique for any given value of the parent key (default: **false**).

*onlyOnce*

Specifies whether it is required that the subindex does not exist prior to this method call (default: **false**). If a subindex with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

*locale*

Locale information used to compare strings in the subindex (default: **CultureInfo.CurrentCulture**).

## Type Parameters

*TKey*

The type of the subindex key.

## Return Value

The new subindex added to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Remarks

A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique.

But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the indexed collection. An attempt to modify the indexed collection violating the uniqueness throws an **System.InvalidOperationException**.

For a subindex, uniqueness means that any given pair of parent key and subindex key values uniquely determines an item in the indexed collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)  
[SubindexCollection<T> Members](#)  
[Overload List](#)

Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean) Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) > [Add Method](#) :

Add<TKey>(Expression<Func<T,TKey>>,Boolean,Boolean) Method

The type of the subindex key.

Key selector expression of the subindex, see [KeySelector](#).

Specifies whether the key used in this subindex is unique for any given value of the parent key (default: **false**).

Specifies whether it is required that the subindex does not exist prior to this method call (default: **false**). If a subindex with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

Creates a new subindex and attaches it to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Add(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)), _  
    ByVal keyIsUnique As System.Boolean, _  
    ByVal onlyOnce As System.Boolean _  
) As Subindex(Of T, TKey)
```

C#

```
public Subindex<T, TKey> Add<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector,  
    System.bool keyIsUnique,  
    System.bool onlyOnce  
)
```

### Parameters

*keySelector*

Key selector expression of the subindex, see [KeySelector](#).

*keyIsUnique*

Specifies whether the key used in this subindex is unique for any given value of the parent key (default: **false**).

*onlyOnce*

Specifies whether it is required that the subindex does not exist prior to this method call (default: **false**). If a subindex with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

### Type Parameters

*TKey*

The type of the subindex key.

### Return Value

The new subindex added to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Remarks

A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique.

But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the indexed collection. An attempt to modify the indexed collection violating the uniqueness throws an **System.InvalidOperationException**.

For a subindex, uniqueness means that any given pair of parent key and subindex key values uniquely determines an item in the indexed collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)  
[SubindexCollection<T> Members](#)  
[Overload List](#)

Add<TKey>(Expression<Func<T,TKey>>,Boolean) Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) > [Add Method](#) :

Add<TKey>(Expression<Func<T,TKey>>,Boolean) Method

The type of the subindex key.

Key selector expression of the subindex, see [KeySelector](#).

Specifies whether the key used in this subindex is unique for any given value of the parent key (default: **false**).

Creates a new subindex and attaches it to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Add(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of
```

```
T, TKey)), _
    ByVal keyIsUnique As System.Boolean _
) As Subindex(Of T, TKey)
```

C#

```
public Subindex<T, TKey> Add<TKey>(
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector,
    System.bool keyIsUnique
)
```

## Parameters

*keySelector*

Key selector expression of the subindex, see [KeySelector](#).

*keyIsUnique*

Specifies whether the key used in this subindex is unique for any given value of the parent key (default: **false**).

## Type Parameters

*TKey*

The type of the subindex key.

## Return Value

The new subindex added to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Remarks

A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique.

But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the indexed collection. An attempt to modify the indexed collection violating the uniqueness throws an

**System.InvalidOperationException**.

For a subindex, uniqueness means that any given pair of parent key and subindex key values uniquely determines an item in the indexed collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)  
[SubindexCollection<T> Members](#)  
[Overload List](#)

Add<TKey>(Expression<Func<T,TKey>>) Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) > [Add Method](#) :

Add<TKey>(Expression<Func<T,TKey>>) Method

The type of the subindex key.

Key selector expression of the subindex, see [KeySelector](#).

Creates a new subindex and attaches it to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Add(Of TKey)( _     ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of T, TKey)) _     ) As Subindex(Of T, TKey)</pre>	
C#	
<pre>public Subindex&lt;T, TKey&gt; Add&lt;TKey&gt;(     System.Linq.Expressions.Expression&lt;Func&lt;T, TKey&gt;&gt; keySelector )</pre>	

### Parameters

*keySelector*

Key selector expression of the subindex, see [KeySelector](#).

### Type Parameters

*TKey*

The type of the subindex key.

## Return Value

The new subindex added to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)  
[SubindexCollection<T> Members](#)  
[Overload List](#)

Add(LambdaExpression, Boolean, Boolean, IndexingAlgorithm, CultureInfo) Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) > [Add Method](#) :

Add(LambdaExpression, Boolean, Boolean, IndexingAlgorithm, CultureInfo) Method

Key selector expression of the subindex, see [KeySelector](#).

Specifies whether the key used in this subindex is unique for any given value of the parent key (default: **false**).

Specifies whether it is required that the subindex does not exist prior to this method call. If a subindex with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

An [IndexingAlgorithm](#) used by the subindex. In the current version, only one algorithm is supported, RedBlackTree. Later versions may support other algorithms, such as bitmap or hash indexing.

Locale information used to compare strings in the subindex (default: **CultureInfo.CurrentCulture**).

Creates a new subindex and attaches it to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Add( _     ByVal keySelector As System.Linq.Expressions.LambdaExpression, _     ByVal keyIsUnique As System.Boolean, _     ByVal onlyOnce As System.Boolean, _</pre>	

```

    ByVal algorithm As IndexingAlgorithm, _
    ByVal locale As System.Globalization.CultureInfo _
) As Subindex(Of T)

```

C#

```

public Subindex<T> Add(
    System.Linq.Expressions.LambdaExpression keySelector,
    System.bool keyIsUnique,
    System.bool onlyOnce,
    IndexingAlgorithm algorithm,
    System.Globalization.CultureInfo locale
)

```

## Parameters

*keySelector*

Key selector expression of the subindex, see [KeySelector](#).

*keyIsUnique*

Specifies whether the key used in this subindex is unique for any given value of the parent key (default: **false**).

*onlyOnce*

Specifies whether it is required that the subindex does not exist prior to this method call. If a subindex with this *keySelector* already exists, an exception is thrown if it is **true**, and this method call is ignored if it is **false**.

*algorithm*

An [IndexingAlgorithm](#) used by the subindex. In the current version, only one algorithm is supported, RedBlackTree. Later versions may support other algorithms, such as bitmap or hash indexing.

*locale*

Locale information used to compare strings in the subindex (default: **CultureInfo.CurrentCulture**).

## Return Value

The new subindex added to its parent's [IndexDefinition<T>.Subindexes](#) collection.

## Remarks



A unique index occupies less memory and performs better than a non-unique index (although the difference isn't dramatic). Therefore, for unique keys, it's recommended to specify the corresponding index as unique.

But do that only if you are sure that the key is indeed unique, as it imposes a uniqueness constraint on the indexed collection. An attempt to modify the indexed collection violating the uniqueness throws an **System.InvalidOperationException**.

For a subindex, uniqueness means that any given pair of parent key and subindex key values uniquely determines an item in the indexed collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)  
[SubindexCollection<T> Members](#)  
[Overload List](#)

### Clear Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) : Clear Method

Clears the collection of all subindexes. All subindexes are detached from the parent and destroyed.

## Syntax

Visual Basic (Declaration)	
<code>Public Sub Clear()</code>	
C#	
<code>public void Clear()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)

[SubindexCollection<T> Members](#)

### Contains Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) : Contains Method

Determines whether a subindex with the specified key selector exists in the collection.

## Overload List

Overload	Description
<a href="#">Contains(LambdaExpression)</a>	Determines whether a subindex with the specified key selector exists in the collection.
<a href="#">Contains&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;)</a>	Determines whether a subindex with the specified key selector exists in the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)

[SubindexCollection<T> Members](#)

### Contains(LambdaExpression) Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) > [Contains Method](#) :

[Contains\(LambdaExpression\) Method](#)

Key selector expression of a subindex, see [KeySelector](#).

Determines whether a subindex with the specified key selector exists in the collection.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Contains( _  
    ByVal keySelector As System.Linq.Expressions.LambdaExpression _  
) As System.Boolean
```

C#

```
public System.bool Contains(  
    System.Linq.Expressions.LambdaExpression keySelector  
)
```

### Parameters

*keySelector*

Key selector expression of a subindex, see [KeySelector](#).

### Return Value

**true** if a subindex with the specified key selector is found in the collection; otherwise, **false**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)

[SubindexCollection<T> Members](#)

[Overload List](#)

[Contains<TKey>\(Expression<Func<T,TKey>>\) Method](#)

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) > [Contains Method](#) :

[Contains<TKey>\(Expression<Func<T,TKey>>\) Method](#)

The type of the subindex key.

Key selector expression of a subindex, see [KeySelector](#).

Determines whether a subindex with the specified key selector exists in the collection.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Contains(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
    ) As System.Boolean
```

C#

```
public System.bool Contains<TKey>(   
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector  
    )
```

### Parameters

*keySelector*

Key selector expression of a subindex, see [KeySelector](#).

### Type Parameters

*TKey*

The type of the subindex key.

### Return Value

**true** if a subindex with the specified key selector is found in the collection; otherwise, **false**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)  
[SubindexCollection<T> Members](#)  
[Overload List](#)

## Find Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) : Find Method

Finds a subindex in the collection by its key selector.

## Overload List

Overload	Description
<a href="#">Find&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;)</a>	Finds a subindex in the collection by its key selector.
<a href="#">Find(LambdaExpression)</a>	Finds a subindex in the collection by its key selector.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)

[SubindexCollection<T> Members](#)

[Find<TKey>\(Expression<Func<T,TKey>>\)](#) Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) > [Find Method](#) :

[Find<TKey>\(Expression<Func<T,TKey>>\)](#) Method

The type of the subindex key.

Key selector expression of an subindex, see [KeySelector](#).

Finds a subindex in the collection by its key selector.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Find(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _
```

```
) As Subindex(Of T,TKey)
```

C#

```
public Subindex<T,TKey> Find<TKey>(  
    System.Linq.Expressions.Expression<Func<T,TKey>> keySelector  
)
```

## Parameters

*keySelector*

Key selector expression of an subindex, see [KeySelector](#).

## Type Parameters

*TKey*

The type of the subindex key.

## Return Value

A subindex with the given key selector, if it is found; otherwise, **null**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[SubindexCollection<T> Class](#)

[SubindexCollection<T> Members](#)

[Overload List](#)

Find(LambdaExpression) Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) > [Find Method](#) : Find(LambdaExpression) Method

Key selector expression of an subindex, see [KeySelector](#).

Finds a subindex in the collection by its key selector.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Find( _
    ByVal keySelector As System.Linq.Expressions.LambdaExpression _
) As Subindex(Of T)
```

C#

```
public Subindex<T> Find(
    System.Linq.Expressions.LambdaExpression keySelector
)
```

## Parameters

*keySelector*

Key selector expression of an subindex, see [KeySelector](#).

## Return Value

A subindex with the given key selector, if it is found; otherwise, **null**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[SubindexCollection<T> Class](#)  
[SubindexCollection<T> Members](#)  
[Overload List](#)

## GetEnumerator Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) : GetEnumerator Method

Returns an enumerator that iterates through the [SubindexCollection<T>](#).

## Syntax

Visual Basic (Declaration)

```
Public Function GetEnumerator() As System.Collections.Generic.IEnumerator(Of
Subindex(Of T))
```

C#

```
public System.Collections.Generic.IEnumerator<Subindex<T>> GetEnumerator()
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)

[SubindexCollection<T> Members](#)

### Remove Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) : Remove Method

Removes a subindex from the collection.

## Overload List

Overload	Description
<a href="#">Remove(Subindex&lt;T&gt;)</a>	Removes a subindex from the collection.
<a href="#">Remove(LambdaExpression)</a>	Removes a subindex from the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)

[SubindexCollection<T> Members](#)

### Remove(Subindex<T>) Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) > Remove Method :

Remove(Subindex<T>) Method

The subindex to remove.



Removes a subindex from the collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Remove( _     ByVal definition As Subindex(Of T) _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool Remove(     Subindex&lt;T&gt; definition )</pre>	

### Parameters

*definition*

The subindex to remove.

### Return Value

**true** if a subindex has been removed; **false** if the subindex does not belong to this collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)  
[SubindexCollection<T> Members](#)  
[Overload List](#)

Remove(LambdaExpression) Method

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) > [Remove Method](#) :  
Remove(LambdaExpression) Method

Key selector expression of a subindex, see [KeySelector](#).

Removes a subindex from the collection.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Remove( _  
    ByVal keySelector As System.Linq.Expressions.LambdaExpression _  
) As System.Boolean
```

C#

```
public System.bool Remove(  
    System.Linq.Expressions.LambdaExpression keySelector  
)
```

### Parameters

*keySelector*

Key selector expression of a subindex, see [KeySelector](#).

### Return Value

**true** if a subindex has been removed; **false** if there is no subindex with the given key selector in the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



[SubindexCollection<T> Class](#)  
[SubindexCollection<T> Members](#)  
[Overload List](#)

### Properties

[C1.LiveLinq.Indexing Namespace](#) : [SubindexCollection<T> Class](#)

For a list of all members of this type, see [SubindexCollection<T> members](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the number of subindexes in the collection.
	<a href="#">Item</a>	Gets the subindex object at the specified ordinal position in the collection.

[Top](#)

## See Also

### Reference

[SubindexCollection<T> Class](#)

[C1.LiveLinq.Indexing Namespace](#)

### Count Property

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) : Count Property

Gets the number of subindexes in the collection.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> Count <b>As</b> System.Integer	
C#	
<b>public</b> System.int Count { <b>get</b> ;}	

### Property Value

The number of subindexes in the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SubindexCollection<T> Class](#)

[SubindexCollection<T> Members](#)

## Item Property

[C1.LiveLinq.Indexing Namespace](#) > [SubindexCollection<T> Class](#) : Item Property

The zero-based ordinal position of the subindex to find

Gets the subindex object at the specified ordinal position in the collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public ReadOnly Default Property Item( _     ByVal ordinal As System.Integer _ ) As Subindex(Of T)</pre>	
C#	
<pre>public Subindex&lt;T&gt; this[     System.int ordinal ]; {get;}</pre>	

## Parameters

*ordinal*

The zero-based ordinal position of the subindex to find

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[SubindexCollection<T> Class](#)

[SubindexCollection<T> Members](#)

## Interfaces

## IIndexedSource<T>

[C1.LiveLinq.Indexing Namespace](#) : [IIndexedSource<T> Interface](#)

The type of the elements of the collection.

Represents an indexed collection.

## Object Model

**IIndexedSource<T>**

## Syntax

Visual Basic (Declaration)

```
Public Interface IIndexedSource(Of T)
```

C#

```
public interface IIndexedSource<T>
```

## Type Parameters

*T*

The type of the elements of the collection.

## Remarks

An indexed collection has a collection of indexes, [ScannerCollection<T>](#) that are maintained up-to-date on every change made to the collection.

This interface is implemented by all main LiveLinq collection classes:

[C1.LiveLinq.Collections.IndexedCollection<T>](#),

[C1.LiveLinq.AdoNet.IndexedDataTable<TRow>](#), [C1.LiveLinq.LiveViews.View<T>](#).

You need to implement this interface only if you want to define your own indexable collection classes and then only if they don't inherit from

[C1.LiveLinq.Collections.IndexedCollection<T>](#), see LiveLinq to Objects:

[IndexedCollection\(T\)](#) and other collection classes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

## Overview

[C1.LiveLinq.Indexing Namespace](#) : [IIndexedSource<T> Interface](#)

The type of the elements of the collection.

Represents an indexed collection.

## Object Model

**IIndexedSource<T>**

## Syntax

Visual Basic (Declaration)	
<b>Public Interface</b> IIndexedSource( <b>Of</b> T)	
C#	
<b>public interface</b> IIndexedSource<T>	

## Type Parameters

*T*

The type of the elements of the collection.

## Remarks

An indexed collection has a collection of indexes, [ScannerCollection<T>](#) that are maintained up-to-date on every change made to the collection.

This interface is implemented by all main LiveLinq collection classes:

[C1.LiveLinq.Collections.IndexedCollection<T>](#),  
[C1.LiveLinq.AdoNet.IndexedDataTable<TRow>](#), [C1.LiveLinq.LiveViews.View<T>](#).

You need to implement this interface only if you want to define your own indexable collection classes and then only if they don't inherit from [C1.LiveLinq.Collections.IndexedCollection<T>](#), see LiveLinq to Objects: [IndexedCollection\(T\)](#) and other collection classes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexedSource<T> Members](#)  
[C1.LiveLinq.Indexing Namespace](#)


## Members

[Properties](#)

[C1.LiveLinq.Indexing Namespace](#) : [IIndexedSource<T>](#) Interface

The following tables list the members exposed by [IIndexedSource<T>](#).

## Public Properties

	Name	Description
	<a href="#">Indexes</a>	Gets the collection of indexes attached to this collection.

[Top](#)

## See Also

### Reference


[IIndexedSource<T> Interface](#)  
[C1.LiveLinq.Indexing Namespace](#)

## Properties

[C1.LiveLinq.Indexing Namespace](#) : [IIndexedSource<T>](#) Interface

For a list of all members of this type, see [IIndexedSource<T> members](#).

## Public Properties

	Name	Description
	<a href="#">Indexes</a>	Gets the collection of indexes attached to this collection.

[Top](#)

## See Also

### Reference

[IIndexedSource<T> Interface](#)  
[C1.LiveLinq.Indexing Namespace](#)

### Indexes Property

[C1.LiveLinq.Indexing Namespace](#) > [IIndexedSource<T> Interface](#) : Indexes Property

Gets the collection of indexes attached to this collection.

## Syntax

Visual Basic (Declaration)	
<b>ReadOnly Property</b> Indexes <b>As</b> <a href="#">ScannerCollection(Of T)</a>	
C#	
<a href="#">ScannerCollection&lt;T&gt;</a> Indexes { <a href="#">get</a> ;}	

### Property Value

A collection of indexes attached to this collection. If this is an independent collection, not the result of a LiveLinq indexing search, then its [Indexes](#) collection contains [Index<T>](#) objects. Otherwise, that is, if it is the result of an indexing search operation such as [Index.Find](#) and others, it contains subindexes implementing [C1.LiveLinq.Indexing.Search.IIndexScanner<T>](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference







[IIndexedSource<T> Interface](#)  
[IIndexedSource<T> Members](#)

## C1.LiveLinq.Indexing.Search Namespace



### Overview

### Classes




	Class	Description
	<a href="#">GroupingQuery&lt;T&gt;</a>	Represents a collection of <a href="#">IndexedGroup&lt;T&gt;</a> , groups of elements with the same key, resulting from a search operation with grouping. This class has a derived class <a href="#">GroupingQuery&lt;TKey,T&gt;</a> with specific type of the key used for the index search.
	<a href="#">GroupingQuery&lt;TKey,T&gt;</a>	Represents a collection of <a href="#">IndexedGroup&lt;TKey,T&gt;</a> , groups of elements with the same key, resulting from a search operation with grouping.
	<a href="#">IndexedGroup&lt;T&gt;</a>	Represents a group of elements with the same key belonging to a collection of groups resulting from a search operation with grouping. This class has a derived class <a href="#">IndexedGroup&lt;TKey,T&gt;</a> with specific key type.
	<a href="#">IndexedGroup&lt;TKey,T&gt;</a>	Represents a group of elements with the same key belonging to a collection of groups resulting from a search operation with grouping.
	<a href="#">IndexQuery&lt;T&gt;</a>	Represents a collection that is the result of an index search. Objects of this class are returned by the <a href="#">IIndexScanner&lt;T&gt;</a> search methods. This class has a derived class <a href="#">IndexQuery&lt;T,TKey&gt;</a> with specific type of the key used for the index search.
	<a href="#">IndexQuery&lt;T,TKey&gt;</a>	Represents a collection that is the result of an index search. Objects of this class are returned by the <a href="#">IIndexScanner&lt;T,TKey&gt;</a> search methods.

## Interfaces

	Interface	Description
	<a href="#">IIndexScanner&lt;T&gt;</a>	Represents an index or a subindex in its capacity of scanning through data. Provides methods for searching data items.
	<a href="#">IIndexScanner&lt;T,TKey&gt;</a>	Represents an index or a subindex in its capacity of scanning

		through data. Provides methods for searching data items.
--	--	--

## Enumerations

	Enumeration	Description
	<a href="#">JoinOperator</a>	A comparison operator to match elements in a join operation.

## See Also

### Reference

[C1.LiveLinq.4 Assembly](#)

## Classes

### GroupingQuery<T>

[C1.LiveLinq.Indexing.Search Namespace](#) : [GroupingQuery<T>](#) Class

The type of the elements of the indexed collection.

Represents a collection of [IndexedGroup<T>](#), groups of elements with the same key, resulting from a search operation with grouping. This class has a derived class [GroupingQuery<TKey,T>](#) with specific type of the key used for the index search.

## Object Model

[GroupingQuery<T>](#)

## Syntax

Visual Basic (Declaration)	
<code>Public MustInherit Class GroupingQuery(Of T)</code>	
C#	
<code>public abstract class GroupingQuery&lt;T&gt;</code>	

## Type Parameters

*T*

The type of the elements of the indexed collection.

## Remarks

The result of any index search operation, [IndexQuery<T>](#) can be grouped by applying [IndexQuery<T>.GroupByUntypedKey](#). It is grouped by the key that was used to perform the search operation. For example, `customers.Indexes(c => c.City).All().GroupByUntypedKey(); customers.Indexes(c => c.City).FindBetween("A", "Z").GroupByUntypedKey();`

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Indexing.Search.GroupingQuery<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupingQuery<T> Members](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[IndexedGroup<TKey,T> Class](#)

### Overview

[C1.LiveLinq.Indexing.Search Namespace](#) : [GroupingQuery<T> Class](#)

The type of the elements of the indexed collection.

Represents a collection of [IndexedGroup<T>](#), groups of elements with the same key, resulting from a search operation with grouping. This class has a derived class [GroupingQuery<TKey,T>](#) with specific type of the key used for the index search.

## Object Model

[GroupingQuery<T>](#)

## Syntax

Visual Basic (Declaration)

```
Public MustInherit Class GroupingQuery(Of T)
```

C#

```
public abstract class GroupingQuery<T>
```

## Type Parameters

*T*

The type of the elements of the indexed collection.

## Remarks

The result of any index search operation, [IndexQuery<T>](#) can be grouped by applying [IndexQuery<T>.GroupByUntypedKey](#). It is grouped by the key that was used to perform the search operation. For example, `customers.Indexes(c => c.City).All().GroupByUntypedKey();` `customers.Indexes(c => c.City).FindBetween("A", "Z").GroupByUntypedKey();`

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Indexing.Search.GroupingQuery<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupingQuery<T> Members](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[IndexedGroup<TKey,T> Class](#)


## Members

[Methods](#)

[C1.LiveLinq.Indexing.Search Namespace](#) : [GroupingQuery<T>](#) Class

The following tables list the members exposed by [GroupingQuery<T>](#).

## Public Methods

	Name	Description
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">GroupingQuery&lt;T&gt;</a> .

[Top](#)

## See Also

### Reference

[GroupingQuery<T> Class](#)

[C1.LiveLinq.Indexing.Search Namespace](#)


[IndexedGroup<TKey,T> Class](#)

## Methods

[C1.LiveLinq.Indexing.Search Namespace](#) : [GroupingQuery<T> Class](#)

For a list of all members of this type, see [GroupingQuery<T> members](#).

## Public Methods

	Name	Description
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">GroupingQuery&lt;T&gt;</a> .

[Top](#)

## See Also

### Reference

[GroupingQuery<T> Class](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[IndexedGroup<TKey,T> Class](#)

### GetEnumerator Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [GroupingQuery<T> Class](#) : GetEnumerator Method

Returns an enumerator that iterates through the [GroupingQuery<T>](#).

## Syntax

Visual Basic (Declaration)	
<b>Public MustOverride Function</b> GetEnumerator() <b>As</b>	

System.Collections.Generic.IEnumerator(Of IndexedGroup(Of T))	
C#	
<pre>public abstract System.Collections.Generic.IEnumerator&lt;IndexedGroup&lt;T&gt;&gt; GetEnumerator()</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupingQuery<T> Class](#)

[GroupingQuery<T> Members](#)

## GroupingQuery<TKey,T>

[C1.LiveLinq.Indexing.Search Namespace](#) : GroupingQuery<TKey,T> Class

The type of the index key.

The type of the elements of the indexed collection.

Represents a collection of [IndexedGroup<TKey,T>](#), groups of elements with the same key, resulting from a search operation with grouping.

## Object Model

GroupingQuery<TKey,T>

## Syntax

Visual Basic (Declaration)	
<pre>Public MustInherit Class GroupingQuery     (Of TKey,T)</pre>	
C#	
<pre>public abstract class GroupingQuery&lt;TKey,T&gt;</pre>	

## Type Parameters

*TKey*

The type of the index key.

*T*

The type of the elements of the indexed collection.

## Remarks

The result of any index search operation, [IndexQuery<T,TKey>](#) can be grouped by applying [IndexQuery<T,TKey>.GroupByKey](#). It is grouped by the key that was used to perform the search operation. For example, `customers.Indexes(c => c.City).All().GroupByKey(); customers.Indexes(c => c.City).FindBetween("A", "Z").GroupByKey();`

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Indexing.Search.GroupingQuery<TKey,T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupingQuery<TKey,T> Members](#)  
[C1.LiveLinq.Indexing.Search Namespace](#)  
[IndexedGroup<T> Class](#)

### Overview

[C1.LiveLinq.Indexing.Search Namespace](#) : [GroupingQuery<TKey,T> Class](#)

The type of the index key.

The type of the elements of the indexed collection.

Represents a collection of [IndexedGroup<TKey,T>](#), groups of elements with the same key, resulting from a search operation with grouping.

## Object Model

**GroupingQuery<TKey,T>**

## Syntax

Visual Basic (Declaration)	
<b>Public MustInherit Class</b> GroupingQuery (Of TKey,T)	
C#	
<b>public abstract class</b> GroupingQuery<TKey,T>	

## Type Parameters

*TKey*

The type of the index key.

*T*

The type of the elements of the indexed collection.

## Remarks

The result of any index search operation, [IndexQuery<T,TKey>](#) can be grouped by applying [IndexQuery<T,TKey>.GroupByKey](#). It is grouped by the key that was used to perform the search operation. For example, `customers.Indexes(c => c.City).All().GroupByKey(); customers.Indexes(c => c.City).FindBetween("A", "Z").GroupByKey();`

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Indexing.Search.GroupingQuery<TKey,T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupingQuery<TKey,T> Members](#)  
[C1.LiveLinq.Indexing.Search Namespace](#)  
[IndexedGroup<T> Class](#)




# Members

## Methods

[C1.LiveLinq.Indexing.Search Namespace](#) : [GroupingQuery<TKey,T>](#) Class

The following tables list the members exposed by [GroupingQuery<TKey,T>](#).

## Public Methods

	Name	Description
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">GroupingQuery&lt;TKey,T&gt;</a> .

[Top](#)

## See Also

### Reference


[GroupingQuery<TKey,T>](#) Class  
[C1.LiveLinq.Indexing.Search Namespace](#)  
[IndexedGroup<T>](#) Class

## Methods

[C1.LiveLinq.Indexing.Search Namespace](#) : [GroupingQuery<TKey,T>](#) Class

For a list of all members of this type, see [GroupingQuery<TKey,T>](#) members.

## Public Methods

	Name	Description
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">GroupingQuery&lt;TKey,T&gt;</a> .

[Top](#)

## See Also

### Reference

[GroupingQuery<TKey,T>](#) Class  
[C1.LiveLinq.Indexing.Search Namespace](#)  
[IndexedGroup<T>](#) Class

## GetEnumerator Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [GroupingQuery<TKey,T> Class](#) : GetEnumerator Method

Returns an enumerator that iterates through the [GroupingQuery<TKey,T>](#).

## Syntax

Visual Basic (Declaration)

```
Public MustOverride Function GetEnumerator() As  
System.Collections.Generic.IEnumerator(Of IndexedGroup(Of TKey,T))
```

C#

```
public abstract System.Collections.Generic.IEnumerator<IndexedGroup<TKey,T>>  
GetEnumerator()
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupingQuery<TKey,T> Class](#)

[GroupingQuery<TKey,T> Members](#)

## IndexedGroup<T>

[C1.LiveLinq.Indexing.Search Namespace](#) : IndexedGroup<T> Class

The type of the elements of the indexed collection.

Represents a group of elements with the same key belonging to a collection of groups resulting from a search operation with grouping. This class has a derived class [IndexedGroup<TKey,T>](#) with specific key type.

## Object Model

IndexedGroup<T>

## Syntax

Visual Basic (Declaration)	
<pre>Public MustInherit Class IndexedGroup(Of T)     Inherits IndexQuery(Of T)     Implements C1.LiveLinq.Indexing.IIndexedSource(Of T)</pre>	
C#	
<pre>public abstract class IndexedGroup&lt;T&gt; : IndexQuery&lt;T&gt;, C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</pre>	

## Type Parameters

*T*

The type of the elements of the indexed collection.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.Search.IndexQuery<T>](#)

**C1.LiveLinq.Indexing.Search.IndexedGroup<T>**

[C1.LiveLinq.Indexing.Search.IndexedGroup<TKey,T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedGroup<T> Members](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[GroupingQuery<T> Class](#)

[IndexedGroup<TKey,T> Class](#)

### Overview

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexedGroup<T> Class](#)

The type of the elements of the indexed collection.

Represents a group of elements with the same key belonging to a collection of groups resulting from a search operation with grouping. This class has a derived class [IndexedGroup<TKey,T>](#) with specific key type.

# Object Model

IndexedGroup<T>

## Syntax

Visual Basic (Declaration)

```
Public MustInherit Class IndexedGroup(Of T)
    Inherits IndexQuery(Of T)
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T)
```

C#

```
public abstract class IndexedGroup<T> : IndexQuery<T>,
    C1.LiveLinq.Indexing.IIndexedSource<T>
```

## Type Parameters

*T*

The type of the elements of the indexed collection.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.Search.IndexQuery<T>](#)

**C1.LiveLinq.Indexing.Search.IndexedGroup<T>**

[C1.LiveLinq.Indexing.Search.IndexedGroup<TKey,T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedGroup<T> Members](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[GroupingQuery<T> Class](#)

[IndexedGroup<TKey,T> Class](#)




## Members

[Properties](#) [Methods](#)

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexedGroup<T>](#) Class




The following tables list the members exposed by [IndexedGroup<T>](#).

### Public Properties

	Name	Description
	<a href="#">Indexes</a>	The collection of subindexes for this <a href="#">IndexQuery&lt;T&gt;</a> . (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )
	<a href="#">Key</a>	Gets the key of the <a href="#">IndexedGroup&lt;T&gt;</a> .
	<a href="#">KeyType</a>	Gets the type of the key of the <a href="#">IndexedGroup&lt;T&gt;</a> .

[Top](#)

### Public Methods

	Name	Description
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">IndexQuery&lt;T&gt;</a> . (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )
	<a href="#">GroupByUntypedKey</a>	Groups the collection of search results by its search key. (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )
	<a href="#">Subindex</a>	Overloaded. Used to apply subindex search to the result of a search operation. (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[IndexedGroup<T> Class](#)

[C1.LiveLinq.Indexing.Search Namespace](#)




[GroupingQuery<T> Class](#)  
[IndexedGroup<TKey,T> Class](#)

## Properties

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexedGroup<T> Class](#)

For a list of all members of this type, see [IndexedGroup<T> members](#).

## Public Properties

	Name	Description
	<a href="#">Indexes</a>	The collection of subindexes for this <a href="#">IndexQuery&lt;T&gt;</a> . (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )
	<a href="#">Key</a>	Gets the key of the <a href="#">IndexedGroup&lt;T&gt;</a> .
	<a href="#">KeyType</a>	Gets the type of the key of the <a href="#">IndexedGroup&lt;T&gt;</a> .

[Top](#)

## See Also

### Reference

[IndexedGroup<T> Class](#)  
[C1.LiveLinq.Indexing.Search Namespace](#)  
[GroupingQuery<T> Class](#)  
[IndexedGroup<TKey,T> Class](#)

### Key Property

[C1.LiveLinq.Indexing.Search Namespace](#) > [IndexedGroup<T> Class](#) : Key Property

Gets the key of the [IndexedGroup<T>](#).

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Key As System.Object</code>	
C#	
<code>public System.object Key {get;}</code>	

### Property Value

The key of the [IndexedGroup<T>](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedGroup<T> Class](#)

[IndexedGroup<T> Members](#)

### KeyType Property

[C1.LiveLinq.Indexing.Search Namespace](#) > [IndexedGroup<T> Class](#) : KeyType Property

Gets the type of the key of the [IndexedGroup<T>](#).

## Syntax

Visual Basic (Declaration)	
<code>Public MustOverride ReadOnly Property KeyType As System.Type</code>	
C#	
<code>public abstract System.Type KeyType {get;}</code>	

### Property Value

Type of the key of the [IndexedGroup<T>](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedGroup<T> Class](#)

[IndexedGroup<T> Members](#)

# IndexedGroup<TKey,T>

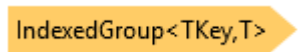
[C1.LiveLinq.Indexing.Search Namespace](#) : IndexedGroup<TKey,T> Class

The type of the index key.

The type of the elements of the indexed collection.

Represents a group of elements with the same key belonging to a collection of groups resulting from a search operation with grouping.

## Object Model



## Syntax

Visual Basic (Declaration)	
<pre>Public MustInherit Class IndexedGroup     (Of TKey,T)     Inherits IndexedGroup(Of T)     Implements C1.LiveLinq.Indexing.IIndexedSource(Of T)</pre>	
C#	
<pre>public abstract class IndexedGroup&lt;TKey,T&gt; : IndexedGroup&lt;T&gt;, C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</pre>	

## Type Parameters

*TKey*

The type of the index key.

*T*

The type of the elements of the indexed collection.

## Inheritance Hierarchy

- System.Object
  - [C1.LiveLinq.Indexing.Search.IndexQuery<T>](#)
    - [C1.LiveLinq.Indexing.Search.IndexedGroup<T>](#)
      - C1.LiveLinq.Indexing.Search.IndexedGroup<TKey,T>**

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedGroup<TKey,T> Members](#)  
[C1.LiveLinq.Indexing.Search Namespace](#)  
[GroupingQuery<TKey,T> Class](#)  
[IndexedGroup<T> Class](#)

### Overview

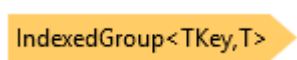
[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexedGroup<TKey,T> Class](#)

The type of the index key.

The type of the elements of the indexed collection.

Represents a group of elements with the same key belonging to a collection of groups resulting from a search operation with grouping.

## Object Model



### Syntax

Visual Basic (Declaration)	
<pre>Public MustInherit Class IndexedGroup     (Of TKey,T)     Inherits IndexedGroup(Of T)     Implements C1.LiveLinq.Indexing.IIndexedSource(Of T)</pre>	
C#	
<pre>public abstract class IndexedGroup&lt;TKey,T&gt; : IndexedGroup&lt;T&gt;,     C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</pre>	

### Type Parameters

*TKey*

The type of the index key.

*T*

The type of the elements of the indexed collection.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.Search.IndexQuery<T>](#)

[C1.LiveLinq.Indexing.Search.IndexedGroup<T>](#)

**C1.LiveLinq.Indexing.Search.IndexedGroup<TKey,T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedGroup<TKey,T> Members](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[GroupingQuery<TKey,T> Class](#)

[IndexedGroup<T> Class](#)




## Members

[Properties](#) [Methods](#)

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexedGroup<TKey,T> Class](#)




The following tables list the members exposed by [IndexedGroup<TKey,T>](#).

## Public Properties

	Name	Description
	<a href="#">Indexes</a>	The collection of subindexes for this <a href="#">IndexQuery&lt;T&gt;</a> . (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )
	<a href="#">Key</a>	Gets the key of the <a href="#">IndexedGroup&lt;TKey,T&gt;</a> .
	<a href="#">KeyType</a>	Overridden. Gets the type of the key of the <a href="#">IndexedGroup&lt;TKey,T&gt;</a> .

[Top](#)

## Public Methods

	Name	Description
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">IndexQuery&lt;T&gt;</a> . (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )
	<a href="#">GroupByUntypedKey</a>	Groups the collection of search results by its search key. (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )
	<a href="#">Subindex</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[IndexedGroup<TKey,T> Class](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[GroupingQuery<TKey,T> Class](#)




[IndexedGroup<T> Class](#)

## Properties

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexedGroup<TKey,T> Class](#)

For a list of all members of this type, see [IndexedGroup<TKey,T> members](#).

## Public Properties

	Name	Description
	<a href="#">Indexes</a>	The collection of subindexes for this <a href="#">IndexQuery&lt;T&gt;</a> . (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )
	<a href="#">Key</a>	Gets the key of the <a href="#">IndexedGroup&lt;TKey,T&gt;</a> .
	<a href="#">KeyType</a>	Overridden. Gets the type of the key of the <a href="#">IndexedGroup&lt;TKey,T&gt;</a> .

[Top](#)

## See Also

## Reference

[IndexedGroup<TKey,T> Class](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[GroupingQuery<TKey,T> Class](#)

[IndexedGroup<T> Class](#)

## Key Property

[C1.LiveLinq.Indexing.Search Namespace](#) > [IndexedGroup<TKey,T> Class](#) : Key Property

Gets the key of the [IndexedGroup<TKey,T>](#).

## Syntax

Visual Basic (Declaration)	
<code>Public MustOverride Shadows ReadOnly Property Key As TKey</code>	
C#	
<code>public abstract new TKey Key {get;}</code>	

## Property Value

The key of the [IndexedGroup<TKey,T>](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedGroup<TKey,T> Class](#)

[IndexedGroup<TKey,T> Members](#)

## KeyType Property

[C1.LiveLinq.Indexing.Search Namespace](#) > [IndexedGroup<TKey,T> Class](#) : KeyType Property

Gets the type of the key of the [IndexedGroup<TKey,T>](#).

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<code>Public Overrides ReadOnly Property KeyType As System.Type</code>	
C#	
<code>public override System.Type KeyType {get;}</code>	

### Property Value

The same type as the *TKey* type parameter of the class.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexedGroup<TKey,T> Class](#)

[IndexedGroup<TKey,T> Members](#)

## IndexQuery<T>

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexQuery<T> Class](#)

The type of the elements of the indexed collection.

Represents a collection that is the result of an index search. Objects of this class are returned by the [IndexScanner<T>](#) search methods. This class has a derived class [IndexQuery<T,TKey>](#) with specific type of the key used for the index search.

## Object Model

[IndexQuery<T>](#)

## Syntax

Visual Basic (Declaration)	
<code>Public MustInherit Class IndexQuery(Of T)     Implements C1.LiveLinq.Indexing.IIndexedSource(Of T)</code>	
C#	
<code>public abstract class IndexQuery&lt;T&gt; : C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</code>	

## Type Parameters

*T*

The type of the elements of the indexed collection.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Indexing.Search.IndexQuery<T>**

[C1.LiveLinq.Indexing.Search.IndexedGroup<T>](#)

[C1.LiveLinq.Indexing.Search.IndexQuery<T,TKey>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexQuery<T> Members](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[IndexQuery<T,TKey> Class](#)

### Overview

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexQuery<T>](#) Class

The type of the elements of the indexed collection.

Represents a collection that is the result of an index search. Objects of this class are returned by the [IIndexScanner<T>](#) search methods. This class has a derived class [IndexQuery<T,TKey>](#) with specific type of the key used for the index search.

## Object Model

[IndexQuery<T>](#)

## Syntax

Visual Basic (Declaration)

```
Public MustInherit Class IndexQuery(Of T)  
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T)
```

C#	
<pre>public abstract class IndexQuery&lt;T&gt; : C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</pre>	

Type Parameters

T

The type of the elements of the indexed collection.

Inheritance Hierarchy

System.Object

**C1.LiveLinq.Indexing.Search.IndexQuery<T>**

[C1.LiveLinq.Indexing.Search.IndexedGroup<T>](#)

[C1.LiveLinq.Indexing.Search.IndexQuery<T,TKey>](#)

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[IndexQuery<T> Members](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[IndexQuery<T,TKey> Class](#)


Members

[Properties](#) [Methods](#)

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexQuery<T>](#) Class

The following tables list the members exposed by [IndexQuery<T>](#).

Public Properties

	Name	Description
	<a href="#">Indexes</a>	The collection of subindexes for this <a href="#">IndexQuery&lt;T&gt;</a> .

[Top](#)

Public Methods

	Name	Description
≡	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">IndexQuery&lt;T&gt;</a> .
≡	<a href="#">GroupByUntypedKey</a>	Groups the collection of search results by its search key.
≡	<a href="#">Subindex</a>	Overloaded. Used to apply subindex search to the result of a search operation.

[Top](#)

## See Also

### Reference

[IndexQuery<T> Class](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[IndexQuery<T,TKey> Class](#)

## Methods

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexQuery<T> Class](#)

For a list of all members of this type, see [IndexQuery<T> members](#).

## Public Methods

	Name	Description
≡	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">IndexQuery&lt;T&gt;</a> .
≡	<a href="#">GroupByUntypedKey</a>	Groups the collection of search results by its search key.
≡	<a href="#">Subindex</a>	Overloaded. Used to apply subindex search to the result of a search operation.

[Top](#)

## See Also

### Reference

[IndexQuery<T> Class](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[IndexQuery<T,TKey> Class](#)



## GetEnumerator Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IndexQuery<T> Class](#) : GetEnumerator Method

Returns an enumerator that iterates through the [IndexQuery<T>](#).

## Syntax

Visual Basic (Declaration)	
<b>Public MustOverride Function</b> GetEnumerator() <b>As</b> System.Collections.Generic.IEnumerator( <b>Of</b> T)	
C#	
<b>public abstract</b> System.Collections.Generic.IEnumerator<T> GetEnumerator()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexQuery<T> Class](#)

[IndexQuery<T> Members](#)

## GroupByUntypedKey Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IndexQuery<T> Class](#) : GroupByUntypedKey Method

Groups the collection of search results by its search key.

## Syntax

Visual Basic (Declaration)	
<b>Public Overridable Function</b> GroupByUntypedKey() <b>As</b> GroupingQuery( <b>Of</b> T)	
C#	
<b>public virtual</b> GroupingQuery<T> GroupByUntypedKey()	

### Return Value

Search result collection grouped by its search key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexQuery<T> Class](#)

[IndexQuery<T> Members](#)

[IndexedGroup<T> Class](#)

[GroupByKey Method](#)

### Subindex Method

[C1.LiveInq.Indexing.Search Namespace](#) > [IndexQuery<T> Class](#) : Subindex Method

Used to apply subindex search to the result of a search operation.

## Overload List

Overload	Description
<a href="#">Subindex&lt;TKey&gt;(Subindex&lt;T,TKey&gt;)</a>	Used to apply subindex search to the result of a search operation.
<a href="#">Subindex(Subindex&lt;T&gt;)</a>	Used to apply subindex search to the result of a search operation.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexQuery<T> Class](#)

[IndexQuery<T> Members](#)

[Subindex<TKey>\(Subindex<T,TKey>\) Method](#)

[C1.LiveInq.Indexing.Search Namespace](#) > [IndexQuery<T> Class](#) > [Subindex Method](#) :

Subindex<TKey>(Subindex<T,TKey>) Method

Type of the subindex key.

Subindex to use for narrowing the search.

Used to apply subindex search to the result of a search operation.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Overridable Function Subindex(Of TKey)( _     ByVal definition As Subindex(Of T,TKey) _ ) As IIndexScanner(Of T,TKey)</pre>	
C#	
<pre>public virtual IIndexScanner&lt;T,TKey&gt; Subindex&lt;TKey&gt;(     Subindex&lt;T,TKey&gt; definition )</pre>	

### Parameters

*definition*

Subindex to use for narrowing the search.

### Type Parameters

*TKey*

Type of the subindex key.

### Return Value

An [IIndexScanner<T,TKey>](#) collection indexed by the subindex that can be used to perform search operations narrowing the collection.

## Remarks

A subindex can be used to further narrow the result of a search operation, if the corresponding subindex exists in the index or subindex used to perform that search operation. For example, `var idxByCity = customers.Indexes(c => c.City); var subindexByContactTitle = idxByCity.Subindexes(c => c.ContactTitle); var ownersInLondon = idxByCity.Find("London").Subindex(subindexByContactTitle).Find("Owner");`

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexQuery<T> Class](#)  
[IndexQuery<T> Members](#)  
[Overload List](#)

Subindex(Subindex<T>) Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IndexQuery<T> Class](#) > [Subindex Method](#) :

Subindex(Subindex<T>) Method

Subindex to use for narrowing the search.

Used to apply subindex search to the result of a search operation.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Subindex( _     ByVal definition As Subindex(Of T) _ ) As IIndexScanner(Of T)</pre>	
C#	
<pre>public IIndexScanner&lt;T&gt; Subindex(     Subindex&lt;T&gt; definition )</pre>	

### Parameters

*definition*

Subindex to use for narrowing the search.

### Return Value

An [IIndexScanner<T>](#) collection indexed by the subindex that can be used to perform search operations narrowing the collection.

## Remarks

A subindex can be used to further narrow the result of a search operation, if the corresponding subindex exists in the index or subindex used to perform that search

```
operation. For example, var idxByCity = customers.Indexes(c => c.City);
var subindexByContactTitle = idxByCity.Subindexes(c =>
c.ContactTitle); var ownersInLondon =
idxByCity.Find("London").Subindex<string>(subindexByContactTitle).
Find("Owner");
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexQuery<T> Class](#)

[IndexQuery<T> Members](#)


[Overload List](#)

## Properties

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexQuery<T> Class](#)

For a list of all members of this type, see [IndexQuery<T> members](#).

## Public Properties

	Name	Description
	<a href="#">Indexes</a>	The collection of subindexes for this <a href="#">IndexQuery&lt;T&gt;</a> .

[Top](#)

## See Also

### Reference

[IndexQuery<T> Class](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[IndexQuery<T,TKey> Class](#)

### Indexes Property

[C1.LiveLinq.Indexing.Search Namespace](#) > [IndexQuery<T> Class](#) : Indexes Property

The collection of subindexes for this [IndexQuery<T>](#).

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Indexes As ScannerCollection(Of T)</code>	
C#	
<code>public ScannerCollection&lt;T&gt; Indexes {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexQuery<T> Class](#)

[IndexQuery<T> Members](#)

## IndexQuery<T,TKey>

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexQuery<T,TKey> Class](#)

The type of the elements of the indexed collection.

The type of the index key.

Represents a collection that is the result of an index search. Objects of this class are returned by the [IndexScanner<T,TKey>](#) search methods.

## Object Model

**IndexQuery<T,TKey>**

## Syntax

Visual Basic (Declaration)	
<code>Public MustInherit Class IndexQuery</code> <code>(Of T, TKey)</code> <code>Inherits IndexQuery(Of T)</code> <code>Implements C1.LiveLinq.Indexing.IIndexedSource(Of T)</code>	
C#	

```
public abstract class IndexQuery<T,TKey> : IndexQuery<T>,
C1.LiveLinq.Indexing.IIndexedSource<T>
```

## Type Parameters

*T*

The type of the elements of the indexed collection.

*TKey*

The type of the index key.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.Search.IndexQuery<T>](#)

**C1.LiveLinq.Indexing.Search.IndexQuery<T,TKey>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexQuery<T,TKey> Members](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[IndexQuery<T> Class](#)

### Overview

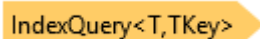
[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexQuery<T,TKey> Class](#)

The type of the elements of the indexed collection.

The type of the index key.

Represents a collection that is the result of an index search. Objects of this class are returned by the [IIndexScanner<T,TKey>](#) search methods.

## Object Model

IndexQuery<T,TKey>

## Syntax

Visual Basic (Declaration)	
<pre>Public MustInherit Class IndexQuery     (Of T,TKey)     Inherits IndexQuery(Of T)     Implements C1.LiveLinq.Indexing.IIndexedSource(Of T)</pre>	
C#	
<pre>public abstract class IndexQuery&lt;T,TKey&gt; : IndexQuery&lt;T&gt;, C1.LiveLinq.Indexing.IIndexedSource&lt;T&gt;</pre>	

## Type Parameters

*T*

The type of the elements of the indexed collection.

*TKey*

The type of the index key.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Indexing.Search.IndexQuery<T>](#)

**C1.LiveLinq.Indexing.Search.IndexQuery<T,TKey>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IndexQuery<T,TKey> Members](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[IndexQuery<T> Class](#)

## Members


[Properties](#) [Methods](#)

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexQuery<T,TKey> Class](#)

The following tables list the members exposed by [IndexQuery<T,TKey>](#).







## Public Properties

	Name	Description
	<a href="#">Indexes</a>	The collection of subindexes for this <a href="#">IndexQuery&lt;T&gt;</a> . (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )

[Top](#)

## Public Methods

	Name	Description
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">IndexQuery&lt;T&gt;</a> . (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )
	<a href="#">GroupByKey</a>	Groups the collection of search results by its search key.
	<a href="#">GroupByUntypedKey</a>	Overridden. Groups the collection of search results by its search key.
	<a href="#">Subindex</a>	Overloaded. Used to apply subindex search to the result of a search operation. (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[IndexQuery<T,TKey> Class](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[IndexQuery<T> Class](#)

### Methods

[C1.LiveLinq.Indexing.Search Namespace](#) : [IndexQuery<T,TKey> Class](#)

For a list of all members of this type, see [IndexQuery<T,TKey> members](#).

## Public Methods

	Name	Description
--	------	-------------

⇒	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the <a href="#">IndexQuery&lt;T&gt;</a> . (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )
⇒	<a href="#">GroupByKey</a>	Groups the collection of search results by its search key.
⇒	<a href="#">GroupByUntypedKey</a>	Overridden. Groups the collection of search results by its search key.
⇒	<a href="#">Subindex</a>	Overloaded. Used to apply subindex search to the result of a search operation. (Inherited from <a href="#">C1.LiveLinq.Indexing.Search.IndexQuery&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[IndexQuery<T,TKey> Class](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

[IndexQuery<T> Class](#)

### GroupByKey Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IndexQuery<T,TKey> Class](#) : GroupByKey Method

Groups the collection of search results by its search key.

## Syntax

Visual Basic (Declaration)	
<b>Public Overridable Function</b> GroupByKey() <b>As</b> GroupingQuery(Of TKey,T)	
C#	
<b>public virtual</b> GroupingQuery<TKey,T> GroupByKey()	

### Return Value

Search result collection grouped by its search key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [IndexQuery<T,TKey> Class](#)
- [IndexQuery<T,TKey> Members](#)
- [IndexedGroup<TKey,T> Class](#)
- [GroupByUntypedKey Method](#)

## GroupByUntypedKey Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IndexQuery<T,TKey> Class](#) : GroupByUntypedKey Method

Groups the collection of search results by its search key.

# Syntax

Visual Basic (Declaration)	
<code>Public Overrides Function GroupByUntypedKey() As GroupingQuery(Of T)</code>	
C#	
<code>public override GroupingQuery&lt;T&gt; GroupByUntypedKey()</code>	

## Return Value

Search result collection grouped by its search key.

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [IndexQuery<T,TKey> Class](#)
- [IndexQuery<T,TKey> Members](#)
- [IndexedGroup<T> Class](#)
- [GroupByKey Method](#)

# Enumerations

## JoinOperator

[C1.LiveLinq.Indexing.Search Namespace](#) : JoinOperator Enumeration

A comparison operator to match elements in a join operation.

## Syntax

Visual Basic (Declaration)	
<pre>Public Enum JoinOperator     Inherits System.Enum</pre>	
C#	
<pre>public enum JoinOperator : System.Enum</pre>	

## Members

Member	Description
<b>Equal</b>	<b>a</b> is equal to <b>b</b> ( <b>a == b</b> )
<b>Greater</b>	<b>a</b> is greater than <b>b</b> ( <b>a &gt; b</b> )
<b>GreaterOrEqual</b>	<b>a</b> is greater than or equal to <b>b</b> ( <b>a &gt;= b</b> )
<b>Less</b>	<b>a</b> is less than <b>b</b> ( <b>a &lt; b</b> )
<b>LessOrEqual</b>	<b>a</b> is less than or equal to <b>b</b> ( <b>a &lt;= b</b> )

## Inheritance Hierarchy

System.Object  
  System.ValueType  
    System.Enum  
      **C1.LiveLinq.Indexing.Search.JoinOperator**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[C1.LiveLinq.Indexing.Search Namespace](#)

# Interfaces

## IIndexScanner<T>

[C1.LiveLinq.Indexing.Search Namespace](#) : IIndexScanner<T> Interface

The type of the elements of the indexed collection.

Represents an index or a subindex in its capacity of scanning through data. Provides methods for searching data items.

# Object Model

IIndexScanner<T>

## Syntax

Visual Basic (Declaration)	
Public Interface IIndexScanner(Of T)	
C#	
public interface IIndexScanner<T>	

## Type Parameters

T

The type of the elements of the indexed collection.

## Remarks

This interface is implemented by [C1.LiveLinq.Indexing.Index<T>](#). It is also used by subindexes, but there it is not directly implemented by [C1.LiveLinq.Indexing.Subindex<T>](#), but rather returned by the [IndexQuery<T>.Subindex](#) method because it depends on the item found by an index or a subindex that is the parent of that subindex.

[IIndexScanner<T>](#) has a typed key counterpart [IIndexScanner<T,TKey>](#) that is used with typed key classes [C1.LiveLinq.Indexing.Index<T,TKey>](#) and [C1.LiveLinq.Indexing.Subindex<T,TKey>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T> Members](#)  
[C1.LiveLinq.Indexing.Search Namespace](#)

## Overview

[C1.LiveLinq.Indexing.Search Namespace](#) : [IIndexScanner<T> Interface](#)

The type of the elements of the indexed collection.

Represents an index or a subindex in its capacity of scanning through data. Provides methods for searching data items.

## Object Model

[IIndexScanner<T>](#)

## Syntax

Visual Basic (Declaration)	
<b>Public Interface</b> IIndexScanner( <b>Of</b> T)	
C#	
<b>public interface</b> IIndexScanner<T>	

## Type Parameters

*T*

The type of the elements of the indexed collection.

## Remarks

This interface is implemented by [C1.LiveLinq.Indexing.Index<T>](#). It is also used by subindexes, but there it is not directly implemented by [C1.LiveLinq.Indexing.Subindex<T>](#), but rather returned by the [IndexQuery<T>.Subindex](#) method because it depends on the item found by an index or a subindex that is the parent of that subindex.

[IIndexScanner<T>](#) has a typed key counterpart [IIndexScanner<T,TKey>](#) that is used with typed key classes [C1.LiveLinq.Indexing.Index<T,TKey>](#) and [C1.LiveLinq.Indexing.Subindex<T,TKey>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T> Members](#)  
[C1.LiveLinq.Indexing.Search Namespace](#)




## Members

[Properties](#) [Methods](#)

[C1.LiveLinq.Indexing.Search Namespace](#) : [IIndexScanner<T>](#) Interface


The following tables list the members exposed by [IIndexScanner<T>](#).










## Public Properties

	Name	Description
	<a href="#">Definition</a>	Gets an <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a> or a <a href="#">C1.LiveLinq.Indexing.Subindex&lt;T&gt;</a> definition on which the scanner is based.
	<a href="#">KeyCount</a>	Gets the number of distinct key values in all items of this collection.
	<a href="#">ParentScanner</a>	Gets an index or a subindex scanner that is the parent of a subindex scanner.

[Top](#)

## Public Methods

	Name	Description
	<a href="#">All</a>	Gets all items in the indexed collection.

 <a href="#">ContainsKey</a>	Returns a value that indicates whether the collection contains an item with the given key value.
 <a href="#">Find</a>	Finds items with the specified key value.
 <a href="#">FindBetween</a>	Finds items with key values in the interval between the specified values.
 <a href="#">FindGreater</a>	Finds items with keys greater than the specified value.
 <a href="#">FindKeys</a>	Finds items containing any of the specified key values.
 <a href="#">FindLess</a>	Finds items with keys less than the specified value.
 <a href="#">FindStartingWith</a>	Finds items with string key values starting with the specified string.
 <a href="#">GroupJoin</a>	Overloaded. Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.
 <a href="#">Join</a>	Overloaded. Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.

[Top](#)

## See Also

### Reference

[IIndexScanner<T> Interface](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

### Methods

[C1.LiveLinq.Indexing.Search Namespace](#) : [IIndexScanner<T> Interface](#)

For a list of all members of this type, see [IIndexScanner<T> members](#).

## Public Methods

Name	Description
------	-------------



⇒	<a href="#">All</a>	Gets all items in the indexed collection.
⇒	<a href="#">ContainsKey</a>	Returns a value that indicates whether the collection contains an item with the given key value.
⇒	<a href="#">Find</a>	Finds items with the specified key value.
⇒	<a href="#">FindBetween</a>	Finds items with key values in the interval between the specified values.
⇒	<a href="#">FindGreater</a>	Finds items with keys greater than the specified value.
⇒	<a href="#">FindKeys</a>	Finds items containing any of the specified key values.
⇒	<a href="#">FindLess</a>	Finds items with keys less than the specified value.
⇒	<a href="#">FindStartingWith</a>	Finds items with string key values starting with the specified string.
⇒	<a href="#">GroupJoin</a>	Overloaded. Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.
⇒	<a href="#">Join</a>	Overloaded. Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.

[Top](#)

## See Also

### Reference

[IIndexScanner<T> Interface](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

### All Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : All Method

Specifies the order of the key values to sort the result.

Gets all items in the indexed collection.

## Syntax

Visual Basic (Declaration)	
<pre>Function All( _     ByVal order As Order _ ) As IndexQuery(Of T)</pre>	
C#	
<pre>IndexQuery&lt;T&gt; All(     Order order )</pre>	

## Parameters

*order*

Specifies the order of the key values to sort the result.

## Return Value

An object enumerating all items of the collection in the specified order of their key values.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T> Interface](#)

[IIndexScanner<T> Members](#)

## ContainsKey Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : ContainsKey Method

The key value to search for

Returns a value that indicates whether the collection contains an item with the given key value.

## Syntax

Visual Basic (Declaration)	
<pre>Function ContainsKey( _     ByVal key As System.Object _</pre>	

```
) As System.Boolean
```

C#

```
System.bool ContainsKey(  
    System.object key  
)
```

## Parameters

*key*

The key value to search for

## Return Value

**true** if the collection contains an element with the specified key value; otherwise, **false**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T> Interface](#)

[IIndexScanner<T> Members](#)

## Find Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : Find Method

The key value to search for.

Finds items with the specified key value.

## Syntax

Visual Basic (Declaration)

```
Function Find( _  
    ByVal key As System.Object _  
) As IndexQuery(Of T)
```

C#

```
IndexQuery<T> Find(  
    System.Object key  
)
```

## Parameters

*key*

The key value to search for.

## Return Value

An object enumerating items having the specified key value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T> Interface](#)  
[IIndexScanner<T> Members](#)

## FindBetween Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : FindBetween Method

Minimum key value to search for.

If **true**, the result includes items with the minimum key value.

Maximum key value to search for.

If **true**, the result includes items with the maximum key value.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with key values in the interval between the specified values.

## Syntax

Visual Basic (Declaration)

```
Function FindBetween( _  
    ByVal min As System.Object, _
```

```

    ByVal minInclusive As System.Boolean, _
    ByVal max As System.Object, _
    ByVal maxInclusive As System.Boolean, _
    ByVal order As Order _
) As IndexQuery(Of T)

```

C#

```

IndexQuery<T> FindBetween(
    System.object min,
    System.bool minInclusive,
    System.object max,
    System.bool maxInclusive,
    Order order
)

```

## Parameters

*min*

Minimum key value to search for.

*minInclusive*

If **true**, the result includes items with the minimum key value.

*max*

Maximum key value to search for.

*maxInclusive*

If **true**, the result includes items with the maximum key value.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items with key values within the specified limits.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T> Interface](#)

[IIndexScanner<T> Members](#)

### FindGreater Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : FindGreater Method

Minimum key value to search for.

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly greater than the specified value.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with keys greater than the specified value.

## Syntax

Visual Basic (Declaration)

```
Function FindGreater( _  
    ByVal key As System.Object, _  
    ByVal inclusive As System.Boolean, _  
    ByVal order As Order _  
) As IndexQuery(Of T)
```

C#

```
IndexQuery<T> FindGreater(  
    System.object key,  
    System.bool inclusive,  
    Order order  
)
```

### Parameters

*key*

Minimum key value to search for.

*inclusive*

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly greater than the specified value.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values are greater than the specified value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T> Interface](#)  
[IIndexScanner<T> Members](#)

## FindKeys Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : FindKeys Method

The key values to search for.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items containing any of the specified key values.

## Syntax

Visual Basic (Declaration)

```
Function FindKeys( _  
    ByVal keys As System.Collections.IEnumerable, _  
    ByVal order As Order _  
) As IndexQuery(Of T)
```

C#

```
IndexQuery<T> FindKeys(  
    System.Collections.IEnumerable keys,  
    Order order
```

```
)
```

## Parameters

*keys*

The key values to search for.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values belong to the specified key value collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T> Interface](#)

[IIndexScanner<T> Members](#)

## FindLess Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : FindLess Method

Maximum key value to search for.

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly less than the specified value.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with keys less than the specified value.

## Syntax

Visual Basic (Declaration)

```
Function FindLess( _
```



```

    ByVal key As System.Object, _
    ByVal inclusive As System.Boolean, _
    ByVal order As Order _
) As IndexQuery(Of T)

```

C#

```

IndexQuery<T> FindLess(
    System.object key,
    System.bool inclusive,
    Order order
)

```

## Parameters

*key*

Maximum key value to search for.

*inclusive*

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly less than the specified value.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveInq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values are less than the specified value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T> Interface](#)  
[IIndexScanner<T> Members](#)

## FindStartingWith Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : FindStartingWith Method

The string to search for as the beginning of key value strings.

An optional condition that found items must satisfy.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with string key values starting with the specified string.

## Syntax

### Visual Basic (Declaration)

```
Function FindStartingWith( _  
    ByVal value As System.String, _  
    ByVal keyPredicate As System.Func(Of String,Boolean), _  
    ByVal order As Order _  
) As IndexQuery(Of T,String)
```

### C#

```
IndexQuery<T,string> FindStartingWith(  
    System.string value,  
    System.Func<string,bool> keyPredicate,  
    Order order  
)
```

## Parameters

*value*

The string to search for as the beginning of key value strings.

*keyPredicate*

An optional condition that found items must satisfy.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values are strings that have a beginning matching the specified string and satisfy the optional condition.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T> Interface](#)  
[IIndexScanner<T> Members](#)

### GroupJoin Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : GroupJoin Method

Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.

## Overload List

Overload	Description
<a href="#">GroupJoin&lt;T2,TResult&gt;(IIndexScanner&lt;T2&gt;,Func&lt;T,IEnumerable&lt;T2&gt;,TResult&gt;)</a>	Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this

	collection .
GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<IEnumerable<T>,T2,TResult>)	Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of the second sequence.
GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<T,IEnumerable<T2>,TResult>)	Correlates the items of this indexed collection with the items of another sequence and groups the results by

	the item of this collection .
--	--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T> Interface](#)  
[IIndexScanner<T> Members](#)

GroupJoin<T2,TResult>(IIndexScanner<T2>,Func<T,IEnumerable<T2>,TResult>) Method  
[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) > [GroupJoin Method](#) :  
 GroupJoin<T2,TResult>(IIndexScanner<T2>,Func<T,IEnumerable<T2>,TResult>) Method

The type of the elements of the second collection.

The type of the result elements.

The second indexed collection to join to this collection.

A function to create a result element from an element from this collection and a collection of matching elements from the second collection.

Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.

## Syntax

Visual Basic (Declaration)	
<b>Overloads</b> <b>Function</b> GroupJoin (Of T2,TResult)( _ ByVal source As IIndexScanner(Of T2), _ ByVal resultSelector As System.Func(Of T,IEnumerable(Of T2),TResult) _ ) As System.Collections.Generic.IEnumerable(Of TResult)	
C#	

```
System.Collections.Generic.IEnumerable<TResult> GroupJoin<T2,TResult>(
    IIndexScanner<T2> source,
    System.Func<T,IEnumerable<T2>,TResult> resultSelector
)
```

## Parameters

*source*

The second indexed collection to join to this collection.

*resultSelector*

A function to create a result element from an element from this collection and a collection of matching elements from the second collection.

## Type Parameters

*T2*

The type of the elements of the second collection.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to group pairs, where each pair consists of an item of this collection and the corresponding enumeration of the items of the second collection joined to it.

## Remarks

Matching of two elements is performed by matching their keys.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T> Interface](#)  
[IIndexScanner<T> Members](#)  
[Overload List](#)

GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<IEnumerable<T>,T2,TResult>  
) Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) > [GroupJoin Method](#) :

GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<IEnumerable<T>,T2,TResult>) Method

The type of the elements of the second sequence.

The type of the result elements.

The second sequence to join to this collection.

A function to extract from an item of the second sequence the value to match against this collection's key value.

A function to create a result element from an element of the second sequence and the collection of matching elements from this collection.

Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of the second sequence.

## Syntax

Visual Basic (Declaration)

**Overloads** **Function** GroupJoin

```
(Of T2,TResult)( _
    ByVal source As System.Collections.Generic.IEnumerable(Of T2), _
    ByVal keySelector As System.Func(Of T2,Object), _
    ByVal resultSelector As System.Func(Of IEnumerable(Of T),T2,TResult) _
) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
System.Collections.Generic.IEnumerable<TResult> GroupJoin<T2,TResult>(
    System.Collections.Generic.IEnumerable<T2> source,
    System.Func<T2,object> keySelector,
    System.Func<IEnumerable<T>,T2,TResult> resultSelector
)
```

### Parameters

*source*

The second sequence to join to this collection.

*keySelector*

A function to extract from an item of the second sequence the value to match against this collection's key value.

*resultSelector*

A function to create a result element from an element of the second sequence and the collection of matching elements from this collection.

## Type Parameters

*T2*

The type of the elements of the second sequence.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to group pairs, where each pair consists of an item of the second collection and the corresponding enumeration of the items of this collection joined to it.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T> Interface](#)

[IIndexScanner<T> Members](#)

[Overload List](#)

GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<T,IEnumerable<T2>,TResult>)  
 ) Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) > [GroupJoin Method](#) :

GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<T,IEnumerable<T2>,TResult>) Method

The type of the elements of the second sequence.

The type of the result elements.



The second sequence to join to this collection.

A function to extract from an item of the second sequence the value to match against this collection's key value.

A function to create a result element from an element from this collection and a collection of matching elements from the second sequence.

Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of this collection.

## Syntax

Visual Basic (Declaration)

```
Overloads Function GroupJoin  
    (Of T2,TResult)( _  
        ByVal source As System.Collections.Generic.IEnumerable(Of T2), _  
        ByVal keySelector As System.Func(Of T2,Object), _  
        ByVal resultSelector As System.Func(Of T,IEnumerable(Of T2),TResult) _  
    ) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
System.Collections.Generic.IEnumerable<TResult> GroupJoin<T2,TResult>(  
    System.Collections.Generic.IEnumerable<T2> source,  
    System.Func<T2,object> keySelector,  
    System.Func<T,IEnumerable<T2>,TResult> resultSelector  
)
```

## Parameters

*source*

The second sequence to join to this collection.

*keySelector*

A function to extract from an item of the second sequence the value to match against this collection's key value.

*resultSelector*

A function to create a result element from an element from this collection and a collection of matching elements from the second sequence.

## Type Parameters

*T2*

The type of the elements of the second sequence.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to group pairs, where each pair consists of an item of this collection and the corresponding enumeration of the items of the second sequence joined to it.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T> Interface](#)  
[IIndexScanner<T> Members](#)  
[Overload List](#)

## Join Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : Join Method

Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.

## Overload List

Overload	Description
<a href="#">Join&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,Object&gt;,Func&lt;T,T2,TResult&gt;,JoinOperator)</a>	Correlates the items of this indexed collection with the items of

	another sequence and returns the combined items with matching keys.
<a href="#">Join&lt;T2,TResult&gt;(IIndexScanner&lt;T2&gt;,Func&lt;T,T2,TResult&gt;,JoinOperator)</a>	Correlates the items of this indexed collection with the items of another indexed collection and returns the combined items with matching keys.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T> Interface](#)  
[IIndexScanner<T> Members](#)

Join<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<T,T2,TResult>,JoinOperator)  
Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) > [Join Method](#) :

Join<T2,TResult>(IEnumerable<T2>,Func<T2,Object>,Func<T,T2,TResult>,JoinOperator) Method

The type of the elements of the second sequence.

The type of the result elements.

The second sequence to join to this collection.

A function to extract from an item of the second sequence the value to match against this collection's key value.

A function to create a result element from two matching elements.

A comparison operator to match elements.

Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.

## Syntax

Visual Basic (Declaration)	
<b>Overloads Function Join</b> (Of T2,TResult)( _ ByVal source As System.Collections.Generic.IEnumerable(Of T2), _ ByVal keySelector As System.Func(Of T2,Object), _ ByVal resultSelector As System.Func(Of T,T2,TResult), _ ByVal op As JoinOperator _ ) As System.Collections.Generic.IEnumerable(Of TResult)	
C#	
System.Collections.Generic.IEnumerable<TResult> Join<T2,TResult>( System.Collections.Generic.IEnumerable<T2> source, System.Func<T2,object> keySelector, System.Func<T,T2,TResult> resultSelector, JoinOperator op )	

### Parameters

*source*

The second sequence to join to this collection.

*keySelector*

A function to extract from an item of the second sequence the value to match against this collection's key value.

*resultSelector*

A function to create a result element from two matching elements.

*op*

A comparison operator to match elements.

## Type Parameters

*T2*

The type of the elements of the second sequence.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to pairs of joined elements of the two collections.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T> Interface](#)

[IIndexScanner<T> Members](#)

[Overload List](#)

[Join<T2,TResult>\(IIndexScanner<T2>,Func<T,T2,TResult>,JoinOperator\) Method](#)

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) > [Join Method](#) :

[Join<T2,TResult>\(IIndexScanner<T2>,Func<T,T2,TResult>,JoinOperator\) Method](#)

The type of the elements of the second collection.

The type of the result elements.

The second indexed collection to join to this collection.

A function to create a result element from two matching elements.

A comparison operator to match elements.

Correlates the items of this indexed collection with the items of another indexed collection and returns the combined items with matching keys.

## Syntax

Visual Basic (Declaration)

Overloads Function Join

```
(Of T2,TResult)( _  
    ByVal source As IIndexScanner(Of T2), _  
    ByVal resultSelector As System.Func(Of T,T2,TResult), _  
    ByVal op As JoinOperator _  
) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
System.Collections.Generic.IEnumerable<TResult> Join<T2,TResult>(   
    IIndexScanner<T2> source,   
    System.Func<T,T2,TResult> resultSelector,   
    JoinOperator op   
)
```

## Parameters

*source*

The second indexed collection to join to this collection.

*resultSelector*

A function to create a result element from two matching elements.

*op*

A comparison operator to match elements.

## Type Parameters

*T2*

The type of the elements of the second collection.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to pairs of joined elements of the two collections.

## Remarks

Matching of two elements is performed by matching their keys.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference




[IIndexScanner<T> Interface](#)  
[IIndexScanner<T> Members](#)  
[Overload List](#)

## Properties

[C1.LiveLinq.Indexing.Search Namespace](#) : [IIndexScanner<T> Interface](#)

For a list of all members of this type, see [IIndexScanner<T> members](#).

## Public Properties

	Name	Description
	<a href="#">Definition</a>	Gets an <a href="#">C1.LiveLinq.Indexing.Index&lt;T&gt;</a> or a <a href="#">C1.LiveLinq.Indexing.Subindex&lt;T&gt;</a> definition on which the scanner is based.
	<a href="#">KeyCount</a>	Gets the number of distinct key values in all items of this collection.
	<a href="#">ParentScanner</a>	Gets an index or a subindex scanner that is the parent of a subindex scanner.

[Top](#)

## See Also

## Reference

[IIndexScanner<T> Interface](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

## Definition Property

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : Definition Property

Gets an [C1.LiveLinq.Indexing.Index<T>](#) or a [C1.LiveLinq.Indexing.Subindex<T>](#) definition on which the scanner is based.

## Syntax

Visual Basic (Declaration)	
<a href="#">ReadOnly Property</a> Definition <a href="#">As IndexDefinition(Of T)</a>	
C#	
<a href="#">IndexDefinition&lt;T&gt;</a> Definition { <a href="#">get</a> ;}	

## Property Value

For a subindex scanner, returns a [C1.LiveLinq.Indexing.Subindex<T>](#). For an index scanner, returns the same [C1.LiveLinq.Indexing.Index<T>](#) object as the scanner itself.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T> Interface](#)

[IIndexScanner<T> Members](#)

## KeyCount Property

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : KeyCount Property

Gets the number of distinct key values in all items of this collection.

## Syntax

Visual Basic (Declaration)	
----------------------------	--



ReadOnly Property KeyCount As System.Integer	
C#	
System.int KeyCount {get;}	

### Property Value

Number of distinct key values in the collection.

## Remarks

This number is not the same as the number of elements in the collection, unless the index key is a unique key of that collection, see [C1.LiveLinq.Indexing.IndexDefinition<T>.KeyIsUnique](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T> Interface](#)

[IIndexScanner<T> Members](#)

### ParentScanner Property

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T> Interface](#) : ParentScanner Property

Gets an index or a subindex scanner that is the parent of a subindex scanner.

## Syntax

Visual Basic (Declaration)	
ReadOnly Property ParentScanner As IIndexScanner(Of T)	
C#	
IIndexScanner<T> ParentScanner {get;}	

### Property Value

Parent scanner for a subindex scanner; **null** for an index scanner.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[IIndexScanner<T> Interface](#)  
[IIndexScanner<T> Members](#)

IIndexScanner<T,TKey>

[C1.LiveLinq.Indexing.Search Namespace](#) : IIndexScanner<T,TKey> Interface

The type of the elements of the indexed collection.

The type of the index key.

Represents an index or a subindex in its capacity of scanning through data. Provides methods for searching data items.

Object Model

IIndexScanner<T,TKey>

Syntax

Visual Basic (Declaration)	
<pre>Public Interface IIndexScanner     (Of T,TKey)     Inherits IIndexScanner(Of T)</pre>	
C#	
<pre>public interface IIndexScanner&lt;T,TKey&gt; : IIndexScanner&lt;T&gt;</pre>	

Type Parameters

- T*  
The type of the elements of the indexed collection.
- TKey*  
The type of the index key.

Remarks

This interface is implemented by [C1.LiveLinq.Indexing.Index<T,TKey>](#). It is also used by subindexes, but there it is not directly implemented by [C1.LiveLinq.Indexing.Subindex<T,TKey>](#), but rather returned by the [IndexQuery<T>.Subindex](#) method because it depends on the item found by an index or a subindex that is the parent of that subindex.

[IIndexScanner<T,TKey>](#) has an untyped key counterpart [IIndexScanner<T>](#) that is used with untyped key classes [C1.LiveLinq.Indexing.Index<T>](#) and [C1.LiveLinq.Indexing.Subindex<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T,TKey> Members](#)  
[C1.LiveLinq.Indexing.Search Namespace](#)

### Overview

[C1.LiveLinq.Indexing.Search Namespace](#) : [IIndexScanner<T,TKey>](#) Interface

The type of the elements of the indexed collection.

The type of the index key.

Represents an index or a subindex in its capacity of scanning through data. Provides methods for searching data items.

## Object Model

[IIndexScanner<T,TKey>](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Interface IIndexScanner     (Of T,TKey)     Inherits IIndexScanner(Of T)</pre>	
C#	

`public interface IIndexScanner<T,TKey> : IIndexScanner<T>`

## Type Parameters

*T*

The type of the elements of the indexed collection.

*TKey*

The type of the index key.

## Remarks

This interface is implemented by [C1.LiveLinq.Indexing.Index<T,TKey>](#). It is also used by subindexes, but there it is not directly implemented by [C1.LiveLinq.Indexing.Subindex<T,TKey>](#), but rather returned by the [IndexQuery<T>.Subindex](#) method because it depends on the item found by an index or a subindex that is the parent of that subindex.

[IIndexScanner<T,TKey>](#) has an untyped key counterpart [IIndexScanner<T>](#) that is used with untyped key classes [C1.LiveLinq.Indexing.Index<T>](#) and [C1.LiveLinq.Indexing.Subindex<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T,TKey> Members](#)  
[C1.LiveLinq.Indexing.Search Namespace](#)

## Members











[Methods](#)

[C1.LiveLinq.Indexing.Search Namespace](#) : [IIndexScanner<T,TKey>](#) Interface

The following tables list the members exposed by [IIndexScanner<T,TKey>](#).

## Public Methods

	Name	Description
--	------	-------------

 <a href="#">All</a>	Gets all items in the indexed collection.
 <a href="#">ContainsKey</a>	Returns a value that indicates whether the collection contains an item with the given key value.
 <a href="#">Find</a>	Finds items with the specified key value.
 <a href="#">FindBetween</a>	Finds items with key values in the interval between the specified values.
 <a href="#">FindGreater</a>	Finds items with keys greater than the specified value.
 <a href="#">FindKeys</a>	Finds items containing any of the specified key values.
 <a href="#">FindLess</a>	Finds items with keys less than the specified value.
 <a href="#">GroupJoin</a>	Overloaded. Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.
 <a href="#">Join</a>	Overloaded. Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.
 <a href="#">Keys</a>	Gets distinct key values in all items of this collection.

[Top](#)

## See Also

### Reference

[IIndexScanner<T,TKey> Interface](#)

[C1.LiveLinq.Indexing.Search Namespace](#)

### Methods

[C1.LiveLinq.Indexing.Search Namespace](#) : [IIndexScanner<T,TKey> Interface](#)

For a list of all members of this type, see [IIndexScanner<T,TKey> members](#).

## Public Methods

	Name	Description
≡	<a href="#">All</a>	Gets all items in the indexed collection.
≡	<a href="#">ContainsKey</a>	Returns a value that indicates whether the collection contains an item with the given key value.
≡	<a href="#">Find</a>	Finds items with the specified key value.
≡	<a href="#">FindBetween</a>	Finds items with key values in the interval between the specified values.
≡	<a href="#">FindGreater</a>	Finds items with keys greater than the specified value.
≡	<a href="#">FindKeys</a>	Finds items containing any of the specified key values.
≡	<a href="#">FindLess</a>	Finds items with keys less than the specified value.
≡	<a href="#">GroupJoin</a>	Overloaded. Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.
≡	<a href="#">Join</a>	Overloaded. Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.
≡	<a href="#">Keys</a>	Gets distinct key values in all items of this collection.

[Top](#)

## See Also

### Reference

[IIndexScanner<T,TKey> Interface](#)  
[C1.LiveLinq.Indexing.Search Namespace](#)

### All Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) : All Method

Specifies the order of the key values to sort the result.

Gets all items in the indexed collection.

## Syntax

Visual Basic (Declaration)	
<pre>Function All( _     ByVal order As Order _ ) As IndexQuery(Of T,TKey)</pre>	
C#	
<pre>IndexQuery&lt;T,TKey&gt; All(     Order order )</pre>	

### Parameters

*order*

Specifies the order of the key values to sort the result.

### Return Value

All items of the collection in the specified order of their key values.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T,TKey> Interface](#)

[IIndexScanner<T,TKey> Members](#)

### ContainsKey Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) : ContainsKey Method

The key value to search for

Returns a value that indicates whether the collection contains an item with the given key value.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

```
Function ContainsKey( _  
    ByVal key As TKey _  
) As System.Boolean
```

C#

```
System.bool ContainsKey(  
    TKey key  
)
```

## Parameters

*key*

The key value to search for

## Return Value

**true** if the collection contains an element with the specified key value; otherwise, **false**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T,TKey> Interface](#)

[IIndexScanner<T,TKey> Members](#)

## Find Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) : Find Method

The key value to search for.

Finds items with the specified key value.

## Syntax

Visual Basic (Declaration)

```
Function Find( _  
    ByVal key As TKey _  
) As IndexQuery(Of T,TKey)
```



C#

```
IndexQuery<T,TKey> Find(  
    TKey key  
)
```

### Parameters

*key*

The key value to search for.

### Return Value

An object enumerating items having the specified key value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T,TKey> Interface](#)

[IIndexScanner<T,TKey> Members](#)

### FindBetween Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) : FindBetween Method

Minimum key value to search for.

If **true**, the result includes items with the minimum key value.

Maximum key value to search for.

If **true**, the result includes items with the maximum key value.

An optional condition that found items must satisfy.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with key values in the interval between the specified values.

## Syntax

## Visual Basic (Declaration)

```
Function FindBetween( _  
    ByVal min As TKey, _  
    ByVal minInclusive As System.Boolean, _  
    ByVal max As TKey, _  
    ByVal maxInclusive As System.Boolean, _  
    ByVal keyPredicate As System.Func(Of TKey, Boolean), _  
    ByVal order As Order _  
) As IndexQuery(Of T, TKey)
```

## C#

```
IndexQuery<T, TKey> FindBetween(  
    TKey min,  
    System.bool minInclusive,  
    TKey max,  
    System.bool maxInclusive,  
    System.Func<TKey, bool> keyPredicate,  
    Order order  
)
```

## Parameters

*min*

Minimum key value to search for.

*minInclusive*

If **true**, the result includes items with the minimum key value.

*max*

Maximum key value to search for.

*maxInclusive*

If **true**, the result includes items with the maximum key value.

*keyPredicate*

An optional condition that found items must satisfy.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveInq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items with key values within the specified limits.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T,TKey> Interface](#)

[IIndexScanner<T,TKey> Members](#)

## FindGreater Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) : FindGreater Method

Minimum key value to search for.

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly greater than the specified value.

An optional condition that found items must satisfy.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with keys greater than the specified value.

## Syntax

Visual Basic (Declaration)

```
Function FindGreater( _  
    ByVal key As TKey, _  
    ByVal inclusive As System.Boolean, _  
    ByVal keyPredicate As System.Func(Of TKey, Boolean), _  
    ByVal order As Order _  
) As IndexQuery(Of T, TKey)
```

C#

```
IndexQuery<T,TKey> FindGreater(  
    TKey key,  
    System.bool inclusive,  
    System.Func<TKey,bool> keyPredicate,  
    Order order  
)
```

## Parameters

*key*

Minimum key value to search for.

*inclusive*

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly greater than the specified value.

*keyPredicate*

An optional condition that found items must satisfy.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values are greater than the specified value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T,TKey> Interface](#)

[IIndexScanner<T,TKey> Members](#)

## FindKeys Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) : FindKeys Method

The key values to search for.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items containing any of the specified key values.

## Syntax

Visual Basic (Declaration)

```
Function FindKeys( _  
    ByVal keys As System.Collections.Generic.IEnumerable(Of TKey), _  
    ByVal order As Order _  
) As IndexQuery(Of T, TKey)
```

C#

```
IndexQuery<T, TKey> FindKeys(  
    System.Collections.Generic.IEnumerable<TKey> keys,  
    Order order  
)
```

## Parameters

*keys*

The key values to search for.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values belong to the specified key value collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T, TKey> Interface](#)  
[IIndexScanner<T, TKey> Members](#)

## FindLess Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) : FindLess Method

Maximum key value to search for.

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly less than the specified value.

An optional condition that found items must satisfy.

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

Finds items with keys less than the specified value.

## Syntax

Visual Basic (Declaration)

```
Function FindLess( _  
    ByVal key As TKey, _  
    ByVal inclusive As System.Boolean, _  
    ByVal keyPredicate As System.Func(Of TKey, Boolean), _  
    ByVal order As Order _  
) As IndexQuery(Of T, TKey)
```

C#

```
IndexQuery<T,TKey> FindLess(  
    TKey key,  
    System.bool inclusive,  
    System.Func<TKey,bool> keyPredicate,  
    Order order  
)
```

## Parameters

*key*

Maximum key value to search for.

*inclusive*

If **true**, the result includes items with the specified key value. Otherwise, the result only includes those with keys strictly less than the specified value.

*keyPredicate*

An optional condition that found items must satisfy.

*order*

Optionally specifies the order of the key values to sort the result ([C1.LiveLinq.Order.Unordered](#) if sorting is not required).

## Return Value

An object enumerating all items whose key values are less than the specified value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T,TKey> Interface](#)

[IIndexScanner<T,TKey> Members](#)

## GroupJoin Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) : GroupJoin Method

Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.

## Overload List

Overload	Description
<a href="#">GroupJoin&lt;T2,TResult&gt;(IIndexScanner&lt;T2,TKey&gt;,Func&lt;T,IEnumerable&lt;T2&gt;,TResult&gt;)</a>	Correlates the items of this indexed collection with the items of another indexed collection

	and groups the results by the item of this collection.
<a href="#">GroupJoin&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,TKey&gt;,Func&lt;IEnumerable&lt;T&gt;,T2,TResult&gt;)</a>	Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of the second sequence.
<a href="#">GroupJoin&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,TKey&gt;,Func&lt;T,IEnumerable&lt;T2&gt;,TResult&gt;)</a>	Correlates the items of this indexed collection with the items of another sequence



	and groups the results by the item of this collection.
--	--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T,TKey> Interface](#)

[IIndexScanner<T,TKey> Members](#)

[GroupJoin<T2,TResult>\(IIndexScanner<T2,TKey>,Func<T,IEnumerable<T2>,TResult>\) Method](#)

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) > [GroupJoin Method](#) :

[GroupJoin<T2,TResult>\(IIndexScanner<T2,TKey>,Func<T,IEnumerable<T2>,TResult>\) Method](#)

The type of the elements of the second collection.

The type of the result elements.

The second indexed collection to join to this collection.

A function to create a result element from an element from this collection and a collection of matching elements from the second collection.

Correlates the items of this indexed collection with the items of another indexed collection and groups the results by the item of this collection.

## Syntax

Visual Basic (Declaration)	
<b>Overloads</b> <b>Function</b> GroupJoin (Of T2,TResult)( _ ByVal source As IIndexScanner(Of T2,TKey), _	

```
ByVal resultSelector As System.Func(Of T, IEnumerable(Of T2), TResult) _
) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
System.Collections.Generic.IEnumerable<TResult> GroupJoin<T2,TResult>(
    IIndexScanner<T2,TKey> source,
    System.Func<T,IEnumerable<T2>,TResult> resultSelector
)
```

## Parameters

*source*

The second indexed collection to join to this collection.

*resultSelector*

A function to create a result element from an element from this collection and a collection of matching elements from the second collection.

## Type Parameters

*T2*

The type of the elements of the second collection.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to group pairs, where each pair consists of an item of this collection and the corresponding enumeration of the items of the second collection joined to it.

## Remarks

Matching of two elements is performed by matching their keys.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T,TKey> Interface](#)  
[IIndexScanner<T,TKey> Members](#)  
[Overload List](#)

`GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<IEnumerable<T>,T2,TResult>) Method`

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) > [GroupJoin Method](#) :  
`GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<IEnumerable<T>,T2,TResult>) Method`

The type of the elements of the second sequence.

The type of the result elements.

The second sequence to join to this collection.

A function to extract from an item of the second sequence the value to match against this collection's key value.

A function to create a result element from an element of the second sequence and the collection of matching elements from this collection.

Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of the second sequence.

## Syntax

Visual Basic (Declaration)

```
Overloads Function GroupJoin
    (Of T2,TResult)( _
    ByVal source As System.Collections.Generic.IEnumerable(Of T2), _
    ByVal keySelector As System.Func(Of T2,TKey), _
    ByVal resultSelector As System.Func(Of IEnumerable(Of T),T2,TResult) _
) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
System.Collections.Generic.IEnumerable<TResult> GroupJoin<T2,TResult>(
    System.Collections.Generic.IEnumerable<T2> source,
    System.Func<T2,TKey> keySelector,
    System.Func<IEnumerable<T>,T2,TResult> resultSelector
)
```

### Parameters

*source*

The second sequence to join to this collection.

*keySelector*

A function to extract from an item of the second sequence the value to match against this collection's key value.

*resultSelector*

A function to create a result element from an element of the second sequence and the collection of matching elements from this collection.

## Type Parameters

*T2*

The type of the elements of the second sequence.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to group pairs, where each pair consists of an item of the second collection and the corresponding enumeration of the items of this collection joined to it.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T,TKey> Interface](#)

[IIndexScanner<T,TKey> Members](#)

[Overload List](#)

`GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<T,IEnumerable<T2>,TResult>)`  
Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) > [GroupJoin Method](#) :

`GroupJoin<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<T,IEnumerable<T2>,TResult>)` Method

The type of the elements of the second sequence.

The type of the result elements.

The second sequence to join to this collection.

A function to extract from an item of the second sequence the value to match against this collection's key value.

A function to create a result element from an element from this collection and a collection of matching elements from the second sequence.

Correlates the items of this indexed collection with the items of another sequence and groups the results by the item of this collection.

## Syntax

Visual Basic (Declaration)

```
Overloads Function GroupJoin  
    (Of T2,TResult)( _  
        ByVal source As System.Collections.Generic.IEnumerable(Of T2), _  
        ByVal keySelector As System.Func(Of T2,TKey), _  
        ByVal resultSelector As System.Func(Of T,IEnumerable(Of T2),TResult) _  
    ) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
System.Collections.Generic.IEnumerable<TResult> GroupJoin<T2,TResult>(  
    System.Collections.Generic.IEnumerable<T2> source,  
    System.Func<T2,TKey> keySelector,  
    System.Func<T,IEnumerable<T2>,TResult> resultSelector  
)
```

## Parameters

*source*

The second sequence to join to this collection.

*keySelector*

A function to extract from an item of the second sequence the value to match against this collection's key value.

*resultSelector*

A function to create a result element from an element from this collection and a collection of matching elements from the second sequence.

## Type Parameters

*T2*

The type of the elements of the second sequence.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to group pairs, where each pair consists of an item of this collection and the corresponding enumeration of the items of the second sequence joined to it.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T,TKey> Interface](#)  
[IIndexScanner<T,TKey> Members](#)  
[Overload List](#)

## Join Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) : Join Method

Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.

## Overload List

Overload	Description
<a href="#">Join&lt;T2,TResult&gt;(IEnumerable&lt;T2&gt;,Func&lt;T2,TKey&gt;,Func&lt;T,T2,TResult&gt;,JoinOperator)</a>	Correlates the items of this indexed collection

	with the items of another sequence and returns the combined items with matching keys.
<a href="#">Join&lt;T2,TResult&gt;(IIndexScanner&lt;T2,TKey&gt;,Func&lt;T,T2,TResult&gt;,JoinOperator)</a>	Correlates the items of this indexed collection with the items of another indexed collection and returns the combined items with matching keys.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IIndexScanner<T,TKey> Interface](#)

[IIndexScanner<T,TKey> Members](#)

Join<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<T,T2,TResult>,JoinOperator) Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) > [Join Method](#) :

Join<T2,TResult>(IEnumerable<T2>,Func<T2,TKey>,Func<T,T2,TResult>,JoinOperator) Method

The type of the elements of the second sequence.

The type of the result elements.

The second sequence to join to this collection.

A function to extract from an item of the second sequence the value to match against this collection's key value.

A function to create a result element from two matching elements.

A comparison operator to match elements.

Correlates the items of this indexed collection with the items of another sequence and returns the combined items with matching keys.

## Syntax

Visual Basic (Declaration)

Overloads Function Join

```
(Of T2,TResult)( _  
    ByVal source As System.Collections.Generic.IEnumerable(Of T2), _  
    ByVal keySelector As System.Func(Of T2,TKey), _  
    ByVal resultSelector As System.Func(Of T,T2,TResult), _  
    ByVal op As JoinOperator _  
) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
System.Collections.Generic.IEnumerable<TResult> Join<T2,TResult>(  
    System.Collections.Generic.IEnumerable<T2> source,  
    System.Func<T2,TKey> keySelector,  
    System.Func<T,T2,TResult> resultSelector,  
    JoinOperator op  
)
```

### Parameters

*source*

The second sequence to join to this collection.

*keySelector*



A function to extract from an item of the second sequence the value to match against this collection's key value.

*resultSelector*

A function to create a result element from two matching elements.

*op*

A comparison operator to match elements.

## Type Parameters

*T2*

The type of the elements of the second sequence.

*TResult*

The type of the result elements.

## Return Value

Enumeration of objects obtained by applying the result selector to pairs of joined elements of the two collections.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[IIndexScanner<T,TKey> Interface](#)  
[IIndexScanner<T,TKey> Members](#)  
[Overload List](#)

[Join<T2,TResult>\(IIndexScanner<T2,TKey>,Func<T,T2,TResult>,JoinOperator\) Method](#)

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) > [Join Method](#) :

[Join<T2,TResult>\(IIndexScanner<T2,TKey>,Func<T,T2,TResult>,JoinOperator\) Method](#)

The type of the elements of the second collection.

The type of the result elements.

The second indexed collection to join to this collection.

A function to create a result element from two matching elements.

A comparison operator to match elements.

Correlates the items of this indexed collection with the items of another indexed collection and returns the combined items with matching keys.

## Syntax

Visual Basic (Declaration)

Overloads Function Join

```
(Of T2,TResult)( _  
    ByVal source As IIndexScanner(Of T2,TKey), _  
    ByVal resultSelector As System.Func(Of T,T2,TResult), _  
    ByVal op As JoinOperator _  
) As System.Collections.Generic.IEnumerable(Of TResult)
```

C#

```
System.Collections.Generic.IEnumerable<TResult> Join<T2,TResult>(  
    IIndexScanner<T2,TKey> source,  
    System.Func<T,T2,TResult> resultSelector,  
    JoinOperator op  
)
```

## Parameters

*source*

The second indexed collection to join to this collection.

*resultSelector*

A function to create a result element from two matching elements.

*op*

A comparison operator to match elements.

## Type Parameters

*T2*

The type of the elements of the second collection.

*TResult*

The type of the result elements.

Return Value

Enumeration of objects obtained by applying the result selector to pairs of joined elements of the two collections.

Remarks

Matching of two elements is performed by matching their keys.

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[IIndexScanner<T,TKey> Interface](#)  
[IIndexScanner<T,TKey> Members](#)  
[Overload List](#)

Keys Method

[C1.LiveLinq.Indexing.Search Namespace](#) > [IIndexScanner<T,TKey> Interface](#) : Keys Method

Specifies the order of the key values to sort the result.

Gets distinct key values in all items of this collection.

Syntax

Visual Basic (Declaration)	
<pre>Function Keys( _     ByVal order As Order _ ) As System.Collections.Generic.IEnumerable(Of TKey)</pre>	
C#	
<pre>System.Collections.Generic.IEnumerable&lt;TKey&gt; Keys(     Order order )</pre>	

Parameters

*order*

Specifies the order of the key values to sort the result.

## Return Value

All distinct key values contained in the items of the collection in the specified order.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference


[IIndexScanner<T,TKey> Interface](#)

[IIndexScanner<T,TKey> Members](#)

# C1.LiveLinq.Listeners Namespace

## Overview

## Classes

	Class	Description
	<a href="#">PropertyChangeListener&lt;T&gt;</a>	Represents a listener object used by LiveLinq to receive notifications of changes to property values in a collection element object. Represents a listener object used by LiveLinq to receive notifications of changes to property values in a collection element object.

## See Also

## Reference

[C1.LiveLinq.4 Assembly](#)

## Classes

## PropertyChangeListener<T>

[C1.LiveLinq.Listeners Namespace](#) : [PropertyChangeListener<T>](#) Class

The type of the elements in the collection.

Represents a listener object used by LiveLinq to receive notifications of changes to property values in a collection element object. Represents a listener object used by LiveLinq to receive notifications of changes to property values in a collection element object.

## Object Model

PropertyChangeListener<T>

## Syntax

Visual Basic (Declaration)	
Public MustInherit Class PropertyChangeListener(Of T)	
C#	
public abstract class PropertyChangeListener<T>	

## Type Parameters

T

The type of the elements in the collection.

## Remarks

In most cases, the default listener mechanism performs its function and does not need user intervention. As an advanced functionality, LiveLinq allows the user to customize default listeners. It may be needed in those rare cases when you use [C1.LiveLinq.Collections.IndexedCollection<T>](#) with element class **T** that does not provide property change notifications, and you can't add such notifications, nor by deriving the class from [C1.LiveLinq.Collections.IndexableObject](#) nor by other means, but you have some other way of knowing when property changes occur (maybe you can listen to some events, for example).

Then you can define your own class derived from the base class **PropertyChangeListener<T>** and pass an object of that class to the [C1.LiveLinq.Collections.IndexedCollection<T>](#) constructor.

## Inheritance Hierarchy

System.Object  
**C1.LiveLinq.Listeners.PropertyChangeListener<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyChangeListener<T> Members](#)  
[C1.LiveLinq.Listeners Namespace](#)

### Overview

[C1.LiveLinq.Listeners Namespace](#) : [PropertyChangeListener<T> Class](#)

The type of the elements in the collection.

Represents a listener object used by LiveLinq to receive notifications of changes to property values in a collection element object. Represents a listener object used by LiveLinq to receive notifications of changes to property values in a collection element object.

## Object Model

[PropertyChangeListener<T>](#)

## Syntax

Visual Basic (Declaration)	
<code>Public MustInherit Class PropertyChangeListener(Of T)</code>	
C#	
<code>public abstract class PropertyChangeListener&lt;T&gt;</code>	

## Type Parameters

*T*

The type of the elements in the collection.

## Remarks

In most cases, the default listener mechanism performs its function and does not need user intervention. As an advanced functionality, LiveLinq allows the user to customize default listeners. It may be needed in those rare cases when you use [C1.LiveLinq.Collections.IndexedCollection<T>](#) with element class **T** that does not provide property change notifications, and you can't add such notifications, nor by deriving the

class from [C1.LiveLinq.Collections.IndexableObject](#) nor by other means, but you have some other way of knowing when property changes occur (maybe you can listen to some events, for example).

Then you can define your own class derived from the base class **PropertyChangeListener<T>** and pass an object of that class to the [C1.LiveLinq.Collections.IndexedCollection<T>](#) constructor.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.Listeners.PropertyChangeListener<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyChangeListener<T> Members](#)  
[C1.LiveLinq.Listeners Namespace](#)




## Members



[Methods](#) [Events](#)

[C1.LiveLinq.Listeners Namespace](#) : [PropertyChangeListener<T>](#) Class

The following tables list the members exposed by [PropertyChangeListener<T>](#).


## Public Methods

	Name	Description
	<a href="#">Clear</a>	Stop listening to all, don't listen to any objects.
	<a href="#">CreateDefault</a>	Creates the default listener used by LiveLinq to listen to property change notifications in objects of type <b>T</b> .
	<a href="#">GetListeningProperties</a>	Gets the list of property names for which change notifications are supported.

	<a href="#">StartListening</a>	Start listening to property changes in a particular object.
	<a href="#">StopListening</a>	Stop listening to property changes in a particular object.

[Top](#)

## Public Events

	Name	Description
	<a href="#">PropertyChanged</a>	Occurs after a property has changed its value.

[Top](#)

## See Also

### Reference

[PropertyChangedListener<T> Class](#)







[C1.LiveLinq.Listeners Namespace](#)

## Methods

[C1.LiveLinq.Listeners Namespace](#) : [PropertyChangedListener<T> Class](#)

For a list of all members of this type, see [PropertyChangedListener<T> members](#).

## Public Methods

	Name	Description
	<a href="#">Clear</a>	Stop listening to all, don't listen to any objects.
 	<a href="#">CreateDefault</a>	Creates the default listener used by LiveLinq to listen to property change notifications in objects of type <b>T</b> .
	<a href="#">GetListeningProperties</a>	Gets the list of property names for which change notifications are supported.
	<a href="#">StartListening</a>	Start listening to property changes in a particular object.
	<a href="#">StopListening</a>	Stop listening to property changes in a particular object.

[Top](#)



## See Also

### Reference

[PropertyChangeListener<T> Class](#)

[C1.LiveLinq.Listeners Namespace](#)

### Clear Method

[C1.LiveLinq.Listeners Namespace](#) > [PropertyChangeListener<T> Class](#) : Clear Method

Stop listening to all, don't listen to any objects.

## Syntax

Visual Basic (Declaration)	
<pre>Public MustOverride Sub Clear()</pre>	
C#	
<pre>public abstract void Clear()</pre>	

## Remarks

LiveLinq calls this method when a collection is cleared, all objects are removed from it, so it no longer needs to be notified of changes in any of those objects.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyChangeListener<T> Class](#)

[PropertyChangeListener<T> Members](#)

### CreateDefault Method

[C1.LiveLinq.Listeners Namespace](#) > [PropertyChangeListener<T> Class](#) : CreateDefault() Method

Creates the default listener used by LiveLinq to listen to property change notifications in objects of type **T**.

## Syntax

Visual Basic (Declaration)	
<code>Public Shared Function CreateDefault() As PropertyChangedListener(Of T)</code>	
C#	
<code>public static PropertyChangedListener&lt;T&gt; CreateDefault()</code>	

## Return Value

The default listener handling property change notifications.

## Remarks

For ADO.NET and XML objects (**DataRow**, **DataRowView**, **XContainer**), their corresponding property change notifications are used.

For objects implementing **INotifyPropertyChanged** interface, that interface is used.

If the class is neither of the above, the default listener tries to use two change notification patterns if they are present in the class:

- events named *propertyNameChanged*
- dependency properties of WPF classes

These patterns don't have to exist in the class, it's just a last ditch attempt of the default listener to find a notification mechanism if the standard one is not found.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyChangedListener<T> Class](#)

[PropertyChangedListener<T> Members](#)

### GetListeningProperties Method

[C1.LiveLinq.Listeners Namespace](#) > [PropertyChangedListener<T> Class](#) : GetListeningProperties Method

Gets the list of property names for which change notifications are supported.

## Syntax

Visual Basic (Declaration)	
<pre>Public MustOverride Function GetListeningProperties() As System.Collections.Generic.IEnumerable(Of String)</pre>	
C#	
<pre>public abstract System.Collections.Generic.IEnumerable&lt;string&gt; GetListeningProperties()</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyChangeListener<T> Class](#)

[PropertyChangeListener<T> Members](#)

### StartListening Method

[C1.LiveInq.Listeners Namespace](#) > [PropertyChangeListener<T> Class](#) : StartListening Method

The object to listen to.

Start listening to property changes in a particular object.

## Syntax

Visual Basic (Declaration)	
<pre>Public MustOverride Sub StartListening( _     ByVal item As T _ )</pre>	
C#	
<pre>public abstract void StartListening(     T item )</pre>	

### Parameters

*item*

The object to listen to.

## Remarks

LiveLinq calls this method when a new object is added to a collection, so LiveLinq needs to be notified of changes to property values in that object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyChangeListener<T> Class](#)

[PropertyChangeListener<T> Members](#)

### StopListening Method

[C1.LiveLinq.Listeners Namespace](#) > [PropertyChangeListener<T> Class](#) : StopListening Method

The object to stop listening to.

Stop listening to property changes in a particular object.

## Syntax

Visual Basic (Declaration)

```
Public MustOverride Sub StopListening( _  
    ByVal item As T _  
)
```

C#

```
public abstract void StopListening(  
    T item  
)
```

### Parameters

*item*

The object to stop listening to.

## Remarks

LiveLinq calls this method when an object is removed from a collection, so LiveLinq no longer needs to be notified of changes to property values in that object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyChangeListener<T> Class](#)


[PropertyChangeListener<T> Members](#)

### Events

[C1.LiveLinq.Listeners Namespace](#) : [PropertyChangeListener<T> Class](#)

For a list of all members of this type, see [PropertyChangeListener<T> members](#).

## Public Events

	Name	Description
	<a href="#">PropertyChanged</a>	Occurs after a property has changed its value.

[Top](#)

## See Also

### Reference

[PropertyChangeListener<T> Class](#)

[C1.LiveLinq.Listeners Namespace](#)

### PropertyChanged Event

[C1.LiveLinq.Listeners Namespace](#) > [PropertyChangeListener<T> Class](#) : PropertyChanged Event

Occurs after a property has changed its value.

## Syntax

Visual Basic (Declaration)

**Public Event** PropertyChanged **As**

System.ComponentModel.PropertyChangedEventHandler	
C#	
<b>public event</b> System.ComponentModel.PropertyChangedEventHandler PropertyChanged	

## Event Data

The event handler receives an argument of type `System.ComponentModel.PropertyChangedEventArgs` containing data related to this event. The following **PropertyChangedEventArgs** properties provide information specific to this event.

Property	Description
<b>PropertyName</b>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



### Reference








[PropertyChangeListener<T> Class](#)  
[PropertyChangeListener<T> Members](#)

# C1.LiveLinq.LiveViews Namespace




## Overview


### Classes

	Class	Description
	<a href="#">AggregationView&lt;TSource,TResult&gt;</a>	Represents a view having a single element calculated by aggregating a source view.
	<a href="#">GroupView&lt;TKey,TElement&gt;</a>	A group in a grouping view.

	<a href="#">OrderedView&lt;T&gt;</a>	Represents a sorted view.
	<a href="#">PropertyIsNotVirtualException</a>	Represents an exception that indicates that a property used in a result selector of a live view is not virtual.
	<a href="#">View</a>	Base class for the <a href="#">View&lt;T&gt;</a> class. Contains members that don't depend on the element type <i>T</i> .
	<a href="#">View&lt;T&gt;</a>	Represents a <i>live view</i> : a LINQ query result that supports two-way data binding and is kept up-to-date with base data.
	<a href="#">ViewRow</a>	Represents a view element (item) for the purposes of dynamic, programmatic access to its properties and data binding.
	<a href="#">ViewRowAddingEventArgs</a>	Provides data for the <a href="#">ViewRowCollection.ViewRowAdding</a> event.
	<a href="#">ViewRowCollection</a>	Represents a collection of <a href="#">ViewRow</a> objects used for programmatic access to view elements (items) and for data binding.
	<a href="#">ViewRowPropertyInfo</a>	Allows to control certain behavior of a property of the element type of a <a href="#">View</a> .

## Enumerations

	Enumeration	Description
	<a href="#">DataBindingMode</a>	Enumeration of the possible data binding modes. It is used by the <a href="#">View.DataBindingMode</a> property.
	<a href="#">ViewMaintenanceMode</a>	Specifies how a view is synchronized with changes in its base data.
	<a href="#">ViewOrder</a>	Specifies whether and how a view must preserve item order if it exists in the source.

	<b>ViewRowState</b>	The state of a view row with regard to edit, add and delete operations if they are performed directly on the view.
---	---------------------	--

## See Also

### Reference

[C1.LiveLinq.4 Assembly](#)

## Classes

### AggregationView<TSource,TResult>

[C1.LiveLinq.LiveViews Namespace](#) : AggregationView<TSource,TResult> Class

The type of the elements of the source view.

The type of the single element of the aggregation view.

Represents a view having a single element calculated by aggregating a source view.

## Object Model

AggregationView<TSource,TResult>

## Syntax

Visual Basic (Declaration)	
<pre>Public Class AggregationView     (Of TSource,TResult)     Inherits View(Of TResult)     Implements C1.LiveLinq.Indexing.IIndexedSource(Of TResult), C1.LiveLinq.IObservableSource(Of TResult)</pre>	
C#	
<pre>public class AggregationView&lt;TSource,TResult&gt; : View&lt;TResult&gt;, C1.LiveLinq.Indexing.IIndexedSource&lt;TResult&gt;, C1.LiveLinq.IObservableSource&lt;TResult&gt;</pre>	

## Type Parameters

*TSource*

The type of the elements of the source view.



*TResult*

The type of the single element of the aggregation view.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

**C1.LiveLinq.LiveViews.AggregationView<TSource,TResult>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AggregationView<TSource,TResult> Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

### Overview

[C1.LiveLinq.LiveViews Namespace](#) : [AggregationView<TSource,TResult>](#) Class

The type of the elements of the source view.

The type of the single element of the aggregation view.

Represents a view having a single element calculated by aggregating a source view.

## Object Model

**AggregationView<TSource,TResult>**

## Syntax

Visual Basic (Declaration)

```
Public Class AggregationView
    (Of TSource,TResult)
    Inherits View(Of TResult)
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of TResult),
    C1.LiveLinq.IObservableSource(Of TResult)
```

C#

```
public class AggregationView<TSource,TResult> : View<TResult>,  
C1.LiveLinq.Indexing.IIndexedSource<TResult>,  
C1.LiveLinq.IObservableSource<TResult>
```

## Type Parameters

*TSource*

The type of the elements of the source view.

*TResult*

The type of the single element of the aggregation view.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

**C1.LiveLinq.LiveViews.AggregationView<TSource,TResult>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AggregationView<TSource,TResult> Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

## Members












[Properties](#) [Methods](#) [Events](#)


[C1.LiveLinq.LiveViews Namespace](#) : [AggregationView<TSource,TResult>](#) Class

The following tables list the members exposed by [AggregationView<TSource,TResult>](#).

## Public Properties









Name	Description
------	-------------

	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited

		from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<b>Value</b>	Gets the value of the single element of the aggregation view.

[Top](#)

## Public Methods

	Name	Description
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">Concat</a>	Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from

		<a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">GroupBy</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">Join&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">SelectMany</a>	Overloaded. (Inherited from

		<a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">View.Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">Union</a>	Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )

[Top](#)

## Public Events

	Name	Description
⚡	<a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )

[Top](#)

## See Also

### Reference

[AggregationView<TSource,TResult> Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)











## Properties


[C1.LiveLinq.LiveViews Namespace](#) : [AggregationView<TSource,TResult> Class](#)

For a list of all members of this type, see [AggregationView<TSource,TResult> members](#).

## Public Properties

	Name	Description
📊	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from

		<a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited

		from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Value</a>	Gets the value of the single element of the aggregation view.

[Top](#)

## See Also

### Reference

[AggregationView<TSource,TResult> Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### Value Property

[C1.LiveLinq.LiveViews Namespace](#) > [AggregationView<TSource,TResult> Class](#) : Value Property

Gets the value of the single element of the aggregation view.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Value As TResult</code>	
C#	
<code>public TResult Value {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AggregationView<TSource,TResult> Class](#)

[AggregationView<TSource,TResult> Members](#)

## GroupView<TKey,TElement>

[C1.LiveLinq.LiveViews Namespace](#) : [GroupView<TKey,TElement> Class](#)

The type of the key used for grouping.

The type of the elements in the group view.



A group in a grouping view.

## Object Model

GridView<TKey,TElement>

## Syntax

Visual Basic (Declaration)

```
<System.Diagnostics.DebuggerDisplayAttribute(Value="\{ Count= {Count}, Key= {Key} \}",  
    Name="",  
    Type="",  
    Target=,  
    TargetTypeName="")>  
<System.Diagnostics.DebuggerTypeProxyAttribute(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.LiveLinq.4, Version=4.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",  
    Target=,  
    TargetTypeName="")>  
Public NotInheritable Class GridView  
    (Of TKey,TElement)  
    Inherits View(Of TElement)  
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of TElement),  
    C1.LiveLinq.IObservableSource(Of TElement)
```

C#

```
[System.Diagnostics.DebuggerDisplay(Value="\{ Count= {Count}, Key= {Key} \}",  
    Name="",  
    Type="",  
    Target=,  
    TargetTypeName="")]  
[System.Diagnostics.DebuggerTypeProxy(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.LiveLinq.4, Version=4.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",  
    Target=,  
    TargetTypeName="")]  
public sealed class GridView<TKey,TElement> : View<TElement>,  
    C1.LiveLinq.Indexing.IIndexedSource<TElement>,  
    C1.LiveLinq.IObservableSource<TElement>
```

## Type Parameters

*TKey*

The type of the key used for grouping.

*TElement*

The type of the elements in the group view.

## Remarks

A grouping view is a result of a **GroupBy** operation on a live view. It is a collection of groups. Each group contains elements with the same key. That collection is a live view, and every group in itself is a live view, an object of type **GroupView<TKey, TElement>**.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

**C1.LiveLinq.LiveViews.GroupView<TKey,TElement>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

### Overview

[C1.LiveLinq.LiveViews Namespace](#) : GroupView<TKey,TElement> Class

The type of the key used for grouping.

The type of the elements in the group view.

A group in a grouping view.

## Object Model

GroupView<TKey,TElement>

## Syntax

### Visual Basic (Declaration)

```
<System.Diagnostics.DebuggerDisplayAttribute(Value="\{ Count= {Count}, Key=
{Key} \}",
    Name="",
    Type="",
    Target=,
    TargetTypeName="")>
<System.Diagnostics.DebuggerTypeProxyAttribute(ProxyTypeName="C1.LiveLinq.Seq
uenceDebugView, C1.LiveLinq.4, Version=4.0.20151.455, Culture=neutral,
PublicKeyToken=2aa4ec5576d6c3ce",
    Target=,
    TargetTypeName="")>
Public NotInheritable Class GroupView
    (Of TKey,TElement)
    Inherits View(Of TElement)
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of TElement),
C1.LiveLinq.IObservableSource(Of TElement)
```

### C#

```
[System.Diagnostics.DebuggerDisplay(Value="\{ Count= {Count}, Key= {Key} \}",
    Name="",
    Type="",
    Target=,
    TargetTypeName="")]
[System.Diagnostics.DebuggerTypeProxy(ProxyTypeName="C1.LiveLinq.SequenceDebu
gView, C1.LiveLinq.4, Version=4.0.20151.455, Culture=neutral,
PublicKeyToken=2aa4ec5576d6c3ce",
    Target=,
    TargetTypeName="")]
public sealed class GroupView<TKey,TElement> : View<TElement>,
C1.LiveLinq.Indexing.IIndexedSource<TElement>,
C1.LiveLinq.IObservableSource<TElement>
```

## Type Parameters

*TKey*

The type of the key used for grouping.

*TElement*

The type of the elements in the group view.

## Remarks

A grouping view is a result of a **GroupBy** operation on a live view. It is a collection of groups. Each group contains elements with the same key. That collection is a live view, and every group in itself is a live view, an object of type **GroupView<TKey, TElement>**.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

**C1.LiveLinq.LiveViews.GroupView<TKey,TElement>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Members](#)

[C1.LiveLinq.LiveViews Namespace](#)



## Members












[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.LiveViews Namespace](#) : [GroupView<TKey,TElement>](#) Class

The following tables list the members exposed by [GroupView<TKey,TElement>](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Overridden. This property overrides <a href="#">View.DeferredMaintenance</a> .
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
	<a href="#">Key</a>	Gets the key value of the group.
	<a href="#">MaintenanceMode</a>	Overridden. This property overrides <a href="#">View.MaintenanceMode</a> .
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Parent</a>	Gets the grouping view (the result of a <b>GroupBy</b> operation) to which this group belongs.
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## Public Methods

	Name	Description
≡	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">Concat</a>	Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">GroupBy</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )

⇒	<a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒	<a href="#">Join&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒	<a href="#">Maintain</a>	Overridden. This method overrides <a href="#">View.Maintain</a> .
⇒	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒	<a href="#">PurgeEmptyGroups</a>	Overridden. This method overrides <a href="#">View.PurgeEmptyGroups</a> .
⇒	<a href="#">Rebuild</a>	Overridden. This method overrides <a href="#">View.Rebuild</a> .
⇒	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒	<a href="#">SelectMany</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">View.Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">ToString</a>	Returns a string that represents this instance of <a href="#">GroupView&lt;TKey,TElement&gt;</a>

⇒💎	<a href="#">Union</a>	Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒💎	<a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )

[Top](#)

## Public Events

	Name	Description
⚡	<a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )

[Top](#)

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)






## Methods

>

Name	Description
⇒💎 <a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒💎 <a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒💎 <a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒💎 <a href="#">AttachView</a>	Overloaded. (Inherited from



	<a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒ <a href="#">Concat</a>	Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒ <a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒ <a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒ <a href="#">GroupBy</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒ <a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒ <a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒ <a href="#">Join&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒ <a href="#">Maintain</a>	Overridden. This method overrides <a href="#">View.Maintain</a> .
⇒ <a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒ <a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒ <a href="#">PurgeEmptyGroups</a>	Overridden. This method overrides <a href="#">View.PurgeEmptyGroups</a> .
⇒ <a href="#">Rebuild</a>	Overridden. This method overrides <a href="#">View.Rebuild</a> .
⇒ <a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )

 <a href="#">SelectMany</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
 <a href="#">SetTransaction</a>	Sets the value of the <a href="#">View.Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">ToString</a>	Returns a string that represents this instance of <a href="#">GroupView&lt;TKey,TElement&gt;</a>
 <a href="#">Union</a>	Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
 <a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )

[Top](#)

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

### Maintain Method

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : Maintain Method

This method overrides [View.Maintain](#).

## Syntax

Visual Basic (Declaration)	
<b>Public Overrides NotOverridable Sub</b> Maintain()	
C#	
<b>public override void</b> Maintain()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)  
[GroupView<TKey,TElement> Members](#)

## PurgeEmptyGroups Method

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : PurgeEmptyGroups Method

This method overrides [View.PurgeEmptyGroups](#).

## Syntax

Visual Basic (Declaration)	
<b>Public Overrides NotOverridable Sub</b> PurgeEmptyGroups()	
C#	
<b>public override void</b> PurgeEmptyGroups()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)  
[GroupView<TKey,TElement> Members](#)

## Rebuild Method

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : Rebuild Method

This method overrides [View.Rebuild](#).

## Syntax

Visual Basic (Declaration)	
<b>Public Overrides NotOverridable Sub</b> Rebuild()	
C#	
<b>public override void</b> Rebuild()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GridView<TKey,TElement> Class](#)  
[GridView<TKey,TElement> Members](#)

### ToString Method

[C1.LiveLinq.LiveViews Namespace](#) > [GridView<TKey,TElement> Class](#) : ToString Method

Returns a string that represents this instance of [GridView<TKey,TElement>](#)

## Syntax

Visual Basic (Declaration)	
<b>Public Overrides Function</b> ToString() <b>As</b> System.String	
C#	
<b>public override</b> System.string ToString()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference













[GridView<TKey,TElement> Class](#)  
[GridView<TKey,TElement> Members](#)


### Properties

[C1.LiveLinq.LiveViews Namespace](#) : [GridView<TKey,TElement> Class](#)

For a list of all members of this type, see [GridView<TKey,TElement> members](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Overridden. This property overrides <a href="#">View.DeferredMaintenance</a> .
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
	<a href="#">Key</a>	Gets the key value of the group.
	<a href="#">MaintenanceMode</a>	Overridden. This property overrides <a href="#">View.MaintenanceMode</a> .
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Parent</a>	Gets the grouping view (the result of a <b>GroupBy</b> operation) to which this group belongs.

	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
---	-----------------------------	---

[Top](#)

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### DeferredMaintenance Property

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : DeferredMaintenance Property

This property overrides [View.DeferredMaintenance](#).

## Syntax

Visual Basic (Declaration)	
<b>Public Overrides NotOverridable ReadOnly Property</b> DeferredMaintenance <b>As</b> System.Boolean	
C#	
<b>public override</b> System.bool DeferredMaintenance { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)

[GroupView<TKey,TElement> Members](#)

## Key Property

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : Key Property

Gets the key value of the group.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Key As TKey</code>	
C#	
<code>public TKey Key {get;}</code>	

## Remarks

The key value is common to the elements of this group.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)

[GroupView<TKey,TElement> Members](#)

## MaintenanceMode Property

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : MaintenanceMode Property

This property overrides [View.MaintenanceMode](#).

## Syntax

Visual Basic (Declaration)	
<code>Public Overrides NotOverridable Property MaintenanceMode As ViewMaintenanceMode</code>	
C#	
<code>public override ViewMaintenanceMode MaintenanceMode {get; set;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)

[GroupView<TKey,TElement> Members](#)

### Parent Property

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : Parent Property

Gets the grouping view (the result of a **GroupBy** operation) to which this group belongs.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Parent As View</code>	
C#	
<code>public View Parent {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)

[GroupView<TKey,TElement> Members](#)

## OrderedView<T>

[C1.LiveLinq.LiveViews Namespace](#) : [OrderedView<T> Class](#)

The type of the elements in the view.

Represents a sorted view.



# Object Model

OrderedView<T>

## Syntax

Visual Basic (Declaration)

```
Public Class OrderedView(Of T)
    Inherits View(Of T)
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T),
C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class OrderedView<T> : View<T>,
C1.LiveLinq.Indexing.IIndexedSource<T>, C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

The type of the elements in the view.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

**C1.LiveLinq.LiveViews.OrderedView<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[OrderedView<T> Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

[OrderBy<TKey>\(Expression<Func<T,TKey>>\) Method](#)

[OrderByDescending<TKey> Method](#)

## Overview

[C1.LiveLinq.LiveViews.Namespace](#) : [OrderedView<T>](#) Class

The type of the elements in the view.

Represents a sorted view.

## Object Model

[OrderedView<T>](#)

## Syntax

Visual Basic (Declaration)

```
Public Class OrderedView(Of T)
    Inherits View(Of T)
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T),
C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class OrderedView<T> : View<T>,
C1.LiveLinq.Indexing.IIndexedSource<T>, C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

The type of the elements in the view.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

**C1.LiveLinq.LiveViews.OrderedView<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[OrderedView<T> Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)  
[OrderBy<TKey>\(Expression<Func<T,TKey>>\) Method](#)  
[OrderByDescending<TKey> Method](#)









## Members




[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.LiveViews Namespace](#) : [OrderedView<T>](#) Class

The following tables list the members exposed by [OrderedView<T>](#).





## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Indexes</a>	Gets the collection of indexes for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

	<b>MoveToFirstOnReset</b>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<b>Order</b>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<b>Transaction</b>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## Public Methods

	Name	Description
	<b>AsCollectionViewFactory</b>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<b>AsDynamic</b>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<b>AttachAggregationView</b>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<b>AttachView</b>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )


⇒  <a href="#">Concat</a>	Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒  <a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">Join&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒  <a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

⇒	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">View.Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">ThenBy&lt;TKey&gt;</a>	Performs a subsequent ordering of view elements in ascending order according to a key.
⇒	<a href="#">ThenByDescending&lt;TKey&gt;</a>	Performs a subsequent ordering of view elements in descending order according to a key.
⇒	<a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Union</a>	Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## Public Events

	Name	Description
--	------	-------------

	<b>Changed</b>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
---	----------------	---

[Top](#)

## See Also

### Reference

[OrderedView<T> Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

[OrderBy<TKey>\(Expression<Func<T,TKey>>\) Method](#)






[OrderByDescending<TKey> Method](#)

## Methods

[C1.LiveLinq.LiveViews Namespace](#) : [OrderedView<T> Class](#)

For a list of all members of this type, see [OrderedView<T> members](#).

## Public Methods

	Name	Description
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a <a href="#">View&lt;dynamic&gt;</a> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">Concat</a>	Concatenation of two views. (Inherited from

		<a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Join&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )



⇒ 	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ 	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ 	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">View.Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ 	<a href="#">ThenBy&lt;TKey&gt;</a>	Performs a subsequent ordering of view elements in ascending order according to a key.
⇒ 	<a href="#">ThenByDescending&lt;TKey&gt;</a>	Performs a subsequent ordering of view elements in descending order according to a key.
⇒ 	<a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">Union</a>	Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ 	<a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## See Also

## Reference

[OrderedView<T> Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)  
[OrderBy<TKey>\(Expression<Func<T,TKey>>\) Method](#)  
[OrderByDescending<TKey> Method](#)

## ThenBy<TKey> Method

[C1.LiveLinq.LiveViews Namespace](#) > [OrderedView<T> Class](#) : ThenBy<TKey>(Expression<Func<T,TKey>>) Method

The type of the key returned by *keySelector*.

A function to extract a key from each element.

Performs a subsequent ordering of view elements in ascending order according to a key.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function ThenBy(Of TKey)( _     ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of T,TKey))) _ ) As OrderedView(Of T)</pre>	
C#	
<pre>public OrderedView&lt;T&gt; ThenBy&lt;TKey&gt;(     System.Linq.Expressions.Expression&lt;Func&lt;T,TKey&gt;&gt; keySelector )</pre>	

### Parameters

*keySelector*

A function to extract a key from each element.

### Type Parameters

*TKey*

The type of the key returned by *keySelector*.

### Return Value

A view whose elements are sorted according to a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[OrderedView<T> Class](#)  
[OrderedView<T> Members](#)

### ThenByDescending<TKey> Method

[C1.LiveLinq.LiveViews Namespace](#) > [OrderedView<T> Class](#) : ThenByDescending<TKey> Method

The type of the key returned by *keySelector*.

A function to extract a key from each element.

Performs a subsequent ordering of view elements in descending order according to a key.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function ThenByDescending(Of TKey)( _     ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of T, TKey)) _ ) As OrderedView(Of T)</pre>	
C#	
<pre>public OrderedView&lt;T&gt; ThenByDescending&lt;TKey&gt;(     System.Linq.Expressions.Expression&lt;Func&lt;T, TKey&gt;&gt; keySelector )</pre>	

### Parameters

*keySelector*

A function to extract a key from each element.

### Type Parameters

*TKey*

The type of the key returned by *keySelector*.

### Return Value

A view whose elements are sorted in descending order according to a key.

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[OrderedView<T> Class](#)  
[OrderedView<T> Members](#)

# PropertyIsNotVirtualException

[C1.LiveLinq.LiveViews Namespace](#) : PropertyIsNotVirtualException Class

Represents an exception that indicates that a property used in a result selector of a live view is not virtual.

# Object Model

PropertyIsNotVirtualException

# Syntax

Visual Basic (Declaration)	
<pre>&lt;System.SerializableAttribute(&gt;&gt; Public Class PropertyIsNotVirtualException     Inherits System.Exception</pre>	
C#	
<pre>[System.Serializable()] public class PropertyIsNotVirtualException : System.Exception</pre>	

# Inheritance Hierarchy

System.Object  
    System.Exception  
        **C1.LiveLinq.LiveViews.PropertyIsNotVirtualException**

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyIsNotVirtualException Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Overview

[C1.LiveLinq.LiveViews Namespace](#) : PropertyIsNotVirtualException Class

Represents an exception that indicates that a property used in a result selector of a live view is not virtual.

## Object Model

PropertyIsNotVirtualException

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.SerializableAttribute(&gt; Public Class PropertyIsNotVirtualException     Inherits System.Exception</pre>	
C#	
<pre>[System.Serializable()] public class PropertyIsNotVirtualException : System.Exception</pre>	

## Inheritance Hierarchy

System.Object  
    System.Exception  
        **C1.LiveLinq.LiveViews.PropertyIsNotVirtualException**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[PropertyIsNotVirtualException Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)


## Members

[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.LiveViews Namespace](#) : [PropertyIsNotVirtualException](#) Class








The following tables list the members exposed by [PropertyIsNotVirtualException](#).





## Protected Constructors

	Name	Description
	<a href="#">PropertyIsNotVirtualException Constructor</a>	Initializes a new instance of the <a href="#">PropertyIsNotVirtualException</a> class with serialized data.

[Top](#)





## Public Properties

	Name	Description
	<a href="#">Context</a>	Lambda expression where a type with the non-virtual property is used.
	<a href="#">Data</a>	(Inherited from System.Exception)
	<a href="#">HelpLink</a>	(Inherited from System.Exception)
	<a href="#">HResult</a>	(Inherited from System.Exception)
	<a href="#">InnerException</a>	(Inherited from System.Exception)
	<a href="#">Message</a>	Overridden. Gets a message that describes the current exception.
	<a href="#">Property</a>	The non-virtual property.

	<a href="#">ResultType</a>	The result type of the lambda expression.
	<a href="#">Source</a>	(Inherited from System.Exception)
	<a href="#">StackTrace</a>	(Inherited from System.Exception)
	<a href="#">TargetSite</a>	(Inherited from System.Exception)


[Top](#)

## Public Methods

	Name	Description
	<a href="#">GetBaseException</a>	(Inherited from System.Exception)
	<a href="#">GetObjectData</a>	(Inherited from System.Exception)
	<a href="#">GetType</a>	(Inherited from System.Exception)
	<a href="#">ToString</a>	(Inherited from System.Exception)

[Top](#)

## Protected Events

	Name	Description
	<a href="#">SerializeObjectState</a>	(Inherited from System.Exception)

[Top](#)

## See Also

### Reference

[PropertyIsNotVirtualException Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## PropertyIsNotVirtualException Constructor

[C1.LiveLinq.LiveViews Namespace](#) > [PropertyIsNotVirtualException Class](#) : PropertyIsNotVirtualException Constructor

The **System.Runtime.Serialization.SerializationInfo** that holds the serialized object data about the exception being thrown.

The **System.Runtime.Serialization.StreamingContext** that contains contextual information about the source or destination.

Initializes a new instance of the [PropertyIsNotVirtualException](#) class with serialized data.

## Syntax

Visual Basic (Declaration)	
<pre>Protected Function New( _     ByVal info As System.Runtime.Serialization.SerializationInfo, _     ByVal context As System.Runtime.Serialization.StreamingContext _ )</pre>	
C#	
<pre>protected PropertyIsNotVirtualException(     System.Runtime.Serialization.SerializationInfo info,     System.Runtime.Serialization.StreamingContext context )</pre>	

### Parameters

*info*

The **System.Runtime.Serialization.SerializationInfo** that holds the serialized object data about the exception being thrown.

*context*

The **System.Runtime.Serialization.StreamingContext** that contains contextual information about the source or destination.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference












[PropertyIsNotVirtualException Class](#)

[PropertyIsNotVirtualException Members](#)



# Properties

>

Name	Description
 <a href="#">Context</a>	Lambda expression where a type with the non-virtual property is used.
 <a href="#">Data</a>	(Inherited from System.Exception)
 <a href="#">HelpLink</a>	(Inherited from System.Exception)
 <a href="#">HResult</a>	(Inherited from System.Exception)
 <a href="#">InnerException</a>	(Inherited from System.Exception)
 <a href="#">Message</a>	Overridden. Gets a message that describes the current exception.
 <a href="#">Property</a>	The non-virtual property.
 <a href="#">ResultType</a>	The result type of the lambda expression.
 <a href="#">Source</a>	(Inherited from System.Exception)
 <a href="#">StackTrace</a>	(Inherited from System.Exception)
 <a href="#">TargetSite</a>	(Inherited from System.Exception)

[Top](#)

## See Also

### Reference

[PropertyIsNotVirtualException Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### Context Property

[C1.LiveLinq.LiveViews Namespace](#) > [PropertyIsNotVirtualException Class](#) : Context Property

Lambda expression where a type with the non-virtual property is used.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property Context As System.Linq.Expressions.LambdaExpression
```

C#	
----	--

<code>public System.Linq.Expressions.LambdaExpression Context {get;}</code>	
---	--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyIsNotVirtualException Class](#)

[PropertyIsNotVirtualException Members](#)

### Message Property

[C1.LiveLinq.LiveViews Namespace](#) > [PropertyIsNotVirtualException Class](#) : Message Property

Gets a message that describes the current exception.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<code>Public Overrides ReadOnly Property Message As System.String</code>	
--	--

C#	
----	--

<code>public override System.string Message {get;}</code>	
---	--

### Property Value

The error message that explains the reason for the exception.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyIsNotVirtualException Class](#)

[PropertyIsNotVirtualException Members](#)

## Property Property

[C1.LiveLinq.LiveViews Namespace](#) > [PropertyIsNotVirtualException Class](#) : Property Property

The non-virtual property.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Property As System.Reflection.PropertyInfo</code>	
C#	
<code>public System.Reflection.PropertyInfo Property {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyIsNotVirtualException Class](#)

[PropertyIsNotVirtualException Members](#)

## ResultType Property

[C1.LiveLinq.LiveViews Namespace](#) > [PropertyIsNotVirtualException Class](#) : ResultType Property

The result type of the lambda expression.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property ResultType As System.Type</code>	
C#	
<code>public System.Type ResultType {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

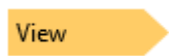
[PropertyIsNotVirtualException Class](#)  
[PropertyIsNotVirtualException Members](#)

# View

[C1.LiveLinq.LiveViews Namespace](#) : View Class

Base class for the [View<T>](#) class. Contains members that don't depend on the element type *T*.

# Object Model



# Syntax

Visual Basic (Declaration)	
<code>Public MustInherit Class View</code>	
C#	
<code>public abstract class View</code>	

# Remarks

Use this class to type variables that can accept views with different element types or a view with anonymous element type.

# Inheritance Hierarchy

System.Object  
    **C1.LiveLinq.LiveViews.View**  
        [C1.LiveLinq.LiveViews.View<T>](#)

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

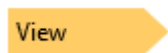
## Reference

## Overview

[C1.LiveLinq.LiveViews Namespace](#) : View Class

Base class for the [View<T>](#) class. Contains members that don't depend on the element type *T*.

## Object Model



## Syntax

Visual Basic (Declaration)

```
Public MustInherit Class View
```

C#

```
public abstract class View
```

## Remarks

Use this class to type variables that can accept views with different element types or a view with anonymous element type.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.View**

[C1.LiveLinq.LiveViews.View<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference











## Members

[Properties](#) [Methods](#)

[C1.LiveLinq.LiveViews Namespace](#) : View Class

The following tables list the members exposed by [View](#).









### Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view.
	<a href="#">CurrentItem</a>	Gets the current item in the view.
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view.
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode.
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable.
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data.
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True.
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source.
	<a href="#">Rows</a>	Gets the collection of <a href="#">ViewRow</a> objects used for programmatic access to view elements (items) and for data binding.
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so

		the programmer does not need to do it manually in code.
--	--	---

[Top](#)

## Public Methods

	Name	Description
≡  <b>S</b>	<a href="#">AllowInResult</a>	Specifies that a type with non-virtual properties can be used in a result selector.
≡ 	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> .
≡ 	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access.
≡ 	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance.
≡ 	<a href="#">Maintain</a>	Brings the view up to date with its source data.
≡ 	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view.
≡ 	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query.
≡ 	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property.

[Top](#)

## See Also

### Reference

[View Class](#)










[C1.LiveLinq.LiveViews Namespace](#)

## Methods

[C1.LiveLinq.LiveViews Namespace](#) : View Class

For a list of all members of this type, see [View members](#).

## Public Methods

	Name	Description
 	<a href="#">AllowInResult</a>	Specifies that a type with non-virtual properties can be used in a result selector.
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> .
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access.
	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance.
	<a href="#">Maintain</a>	Brings the view up to date with its source data.
	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view.
	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query.
	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property.

[Top](#)

## See Also

### Reference

[View Class](#)

[C1.LiveLinq.LiveViews Namespace](#)



## AllowInResult Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : AllowInResult Method

The type that is allowed to be used.

Specifies that a type with non-virtual properties can be used in a result selector.

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared Sub AllowInResult( _     ByVal type As System.Type _ )</pre>	
C#	
<pre>public static void AllowInResult(     System.Type type )</pre>	

## Parameters

*type*

The type that is allowed to be used.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

## AsCollectionViewFactory Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : AsCollectionViewFactory Method

Returns an instance of **System.ComponentModel.ICollectionViewFactory** that can be used as a source of a **System.Windows.Data.CollectionViewSource**.

## Syntax

Visual Basic (Declaration)	
<pre><b>Public Function</b> AsCollectionViewFactory() <b>As</b> System.ComponentModel.ICollectionViewFactory</pre>	
C#	
<pre><b>public</b> System.ComponentModel.ICollectionViewFactory AsCollectionViewFactory()</pre>	

### Return Value

A factory that returns the same View as a **System.ComponentModel.ICollectionView**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

### AsDynamic Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : AsDynamic Method

Used for views with anonymous type constructor as the result selector, converts the [View](#) to a View<dynamic> so it can be used for data binding and programmatic access.

## Syntax

Visual Basic (Declaration)	
<pre><b>Public Function</b> AsDynamic() <b>As</b> View(Of Object)</pre>	
C#	
<pre><b>public</b> View&lt;object&gt; AsDynamic()</pre>	

### Return Value

A dynamic view.

## Remarks

A view with anonymous type constructor as the result selector cannot be used for data binding or programmatic access without applying [AsDynamic](#) to it. An attempt to do so results in an exception. After applying [AsDynamic](#), such view can be used for data binding and programmatic access without limitations.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

### DeferMaintenance Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : DeferMaintenance Method

Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance.

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> DeferMaintenance() <b>As</b> System.IDisposable	
C#	
<b>public</b> System.IDisposable DeferMaintenance()	

### Return Value

An **System.IDisposable** object that you can use to dispose of the calling object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

## Maintain Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : Maintain Method

Brings the view up to date with its source data.

## Syntax

Visual Basic (Declaration)	
<code>Public Overridable Sub Maintain()</code>	
C#	
<code>public virtual void Maintain()</code>	

## Remarks

If source data have not changed since the last time the view was maintained (updated), this method does nothing. It also does nothing if the view's [MaintenanceMode](#) is **Immediate**, because in that case the view is guaranteed to be in synch with its base data at all times. If the view is in deferred mode (its [MaintenanceMode](#) property returns **true**), the programmer can use the **Maintain** method to force updating the view.

Note that it is not necessary to call **Maintain** to make sure you get updated data from the view. The view is automatically updated every time you request data from it, if base data changed since the last request, regardless of the view's [MaintenanceMode](#).

LiveLinq maintains views using optimized incremental algorithms, not simply re-populates them from scratch. It calculating the delta in the view from the delta in the base data. In most cases, it allows to propagate the change from base data to the view very fast.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

## PurgeEmptyGroups Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : PurgeEmptyGroups Method

Remove empty groups from a grouping view.

### Syntax

Visual Basic (Declaration)

```
Public Overridable Sub PurgeEmptyGroups()
```

C#

```
public virtual void PurgeEmptyGroups()
```

### Remarks

This method is used only for **GroupBy** (grouping) views, does nothing for views of other kinds.

When a grouping view is populated, it does not contain empty groups. But later, as a result of maintaining the grouping view, keeping it in synch with changes in its base data, some of the groups can become empty. If it is undesirable to have empty groups, you can call this method to delete them.

### Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

### See Also

#### Reference

[View Class](#)

[View Members](#)

#### Rebuild Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : Rebuild Method

Re-populates the view by re-executing the view's query.

### Syntax

Visual Basic (Declaration)

```
Public Overridable Sub Rebuild()
```

C#

```
public virtual void Rebuild()
```

## Remarks

This method is rarely needed, because normally automatic incremental [maintenance](#) is faster than re-executing the query over the entire base data collection. However, if for some reason you need to re-populate it from scratch, that can be done with the **Rebuild** method.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

### SetTransaction Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : SetTransaction Method

The new value for the the [Transaction](#) property.

Set this parameter to True to prevent exception if you have writable property paths and want to ignore them (but be aware that updates through property paths are not tracked by the transaction). The default is False.

Sets the value of the [Transaction](#) property.

## Syntax

Visual Basic (Declaration)

```
Public Sub SetTransaction( _  
    ByVal transaction As ITransaction, _  
    Optional ByVal allowPropertyPaths As System.Boolean _  
)
```

C#

```
public void SetTransaction(  
    ITransaction transaction,
```

```
System.bool allowPropertyPaths
)
```

## Parameters

*transaction*

The new value for the the [Transaction](#) property.

*allowPropertyPaths*

Set this parameter to True to prevent exception if you have writable property paths and want to ignore them (but be aware that updates through property paths are not tracked by the transaction). The default is False.

## Remarks

If a writable property path exists in the view element type (for example, `Order.Customer.City`), then an exception is thrown unless suppressed by setting [allowPropertyPaths](#) to True, because modifying properties using such paths cannot be supported by transactions. To prevent the exception, set the [allowPropertyPaths](#) parameter to True, but make sure you do not use such property paths in your code and don't have a two-way data binding to such property path in your controls. If you modify a property using such path in your code or through data binding, that modification happens outside the transaction scope.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)











## Properties

[C1.LiveLinq.LiveViews Namespace](#) : View Class

For a list of all members of this type, see [View members](#).

## Public Properties

Name	Description
------	-------------

	<a href="#">Count</a>	Gets the total number of elements in the view.
	<a href="#">CurrentItem</a>	Gets the current item in the view.
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view.
	<a href="#">DeferredMaintenance</a>	Gets the effective value of <a href="#">MaintenanceMode</a> .
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable.
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data.
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True.
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source.
	<a href="#">Rows</a>	Gets the collection of <a href="#">ViewRow</a> objects used for programmatic access to view elements (items) and for data binding.
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code.

[Top](#)

## See Also

### Reference

[View Class](#)

[C1.LiveLinq.LiveViews Namespace](#)



## Count Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : Count Property

Gets the total number of elements in the view.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Count As System.Integer</code>	
C#	
<code>public System.int Count {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

## CurrentItem Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : CurrentItem Property

Gets the current item in the view.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property CurrentItem As System.Object</code>	
C#	
<code>public System.object CurrentItem {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

### DataBindingMode Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : DataBindingMode Property

Gets or sets the data binding mode for this view.

## Syntax

Visual Basic (Declaration)

```
Public Property DataBindingMode As DataBindingMode
```

C#

```
public DataBindingMode DataBindingMode {get; set;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

[DataBindingMode enumeration.](#)

### DeferredMaintenance Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : DeferredMaintenance Property

Gets the effective value of MaintenanceMode.

## Syntax

Visual Basic (Declaration)

```
Public Overridable ReadOnly Property DeferredMaintenance As System.Boolean
```

C#	
<code>public virtual System.bool DeferredMaintenance {get;}</code>	

## Property Value

**true** if [MaintenanceMode](#) is set to **Deferred**, or if it is set to **Default** and no listeners are registered with this view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

### IsReadOnly Property

[C1.LiveInq.LiveViews Namespace](#) > [View Class](#) : IsReadOnly Property

Gets a value indicating whether this view is read-only, not updatable.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property IsReadOnly As System.Boolean</code>	
C#	
<code>public System.bool IsReadOnly {get;}</code>	

## Remarks

Properties exposed by a view can be updatable or read-only. Updatable properties of a view can be modified directly in the view.

All properties of a read-only (not updatable) view are read-only. In an updatable view, properties directly corresponding to base data (source) properties are updatable, calculated properties are read-only. For example, in a view `from x in X select new { x.P, Q = x.Q + 1 }` P is updatable and Q is read-only.

Read-only properties of a view cannot be modified directly in the view, but they still reflect up-to-date values of the source, so the difference is often not critical, you can always modify corresponding property in the source, that will automatically change the property in the view.

Also, a read-only view cannot be cleared or its rows deleted or new rows added directly in the view (but all these actions can be performed on the source data collection and they will normally result in the corresponding changes of the view).

A **Join** view is read-only by default, but one of its two parts can be made updatable using the [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#) method.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

- [View Class](#)
- [View Members](#)
- [AsUpdatable<T> Method](#)
- [ViewRow Class](#)
- [ViewRowState Enumeration](#)

### MaintenanceMode Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : MaintenanceMode Property

Gets or sets a value controlling how the view is synchronized with changes in its base data.

## Syntax

Visual Basic (Declaration)	
<code>Public Overridable Property MaintenanceMode As ViewMaintenanceMode</code>	
C#	
<code>public virtual ViewMaintenanceMode MaintenanceMode {get; set;}</code>	

## Remarks

A view in **Default** mode (which is the default value for this property) is effectively in **Immediate** mode if it has a listener (for example, if a GUI control is bound to it); otherwise it is in **Deferred** mode. To find out whether the view is effectively in **Deferred** or in **Immediate** mode, you can use the [DeferredMaintenance](#) property.

If you set this property to **Deferred**, no listeners are allowed to register with this view. An attempt to register a listener will result in an exception.

**See Also:** View Maintenance Mode.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

[DeferredMaintenance Property](#)

### MoveToFirstOnReset Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : MoveToFirstOnReset Property

Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any [C1.LiveLinq.SourceChangeType.Reset](#) notification) if current item was not set by other means. The default is True.

## Syntax

Visual Basic (Declaration)	
<code>Public Property MoveToFirstOnReset As System.Boolean</code>	
C#	
<code>public System.bool MoveToFirstOnReset {get; set;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

## Order Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : Order Property

Gets a value indicating whether and how this view preserves item order if it exists in its base data source.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property Order As ViewOrder
```

C#

```
public ViewOrder Order {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

## Rows Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : Rows Property

Gets the collection of [ViewRow](#) objects used for programmatic access to view elements (items) and for data binding.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property Rows As ViewRowCollection
```

C#

```
public ViewRowCollection Rows {get;}
```

## Remarks

There can be a view with elements of any type *T*, as represented by the generic class [View<T>](#). That type is not always known beforehand, so it is not always possible to access properties of view elements with strong-typed (early binding) code. Dynamic (untyped, late binding) access to view elements is provided by the [ViewRowCollection](#) owned by the view.

The collection of *view rows* ([ViewRow](#) objects) is always synchronized with the collection of view elements. [ViewRow](#) objects provide programmatic access to view elements and their properties.

Also, in WinForms, the **Rows** collection serves as the data source for data binding, when you bind a control or another client to a view, because a view returns its **Rows** collection in its implementation of the **System.ComponentModel.IListSource** interface.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

### Transaction Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : Transaction Property

Gets an instance of [C1.LiveLinq.ITransaction](#) associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Transaction As ITransaction</code>	
C#	
<code>public ITransaction Transaction {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[View Class](#)  
[View Members](#)

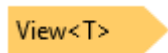
# View<T>

[C1.LiveLinq.LiveViews Namespace](#) : View<T> Class

The type of the elements in the view.

Represents a *live view*: a LINQ query result that supports two-way data binding and is kept up-to-date with base data.

# Object Model



# Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Diagnostics.DebuggerDisplayAttribute(Value="Count={Count}",     Name="",     Type="",     Target=,     TargetTypeName="")&gt; &lt;System.Reflection.DefaultMemberAttribute("Item")&gt; &lt;System.Diagnostics.DebuggerTypeProxyAttribute(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.LiveLinq.4, Version=4.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",     Target=,     TargetTypeName="")&gt; Public Class View(Of T)     Inherits View     Implements C1.LiveLinq.Indexing.IIndexedSource(Of T), C1.LiveLinq.IObservableSource(Of T)</pre>	
C#	
<pre>[System.Diagnostics.DebuggerDisplay(Value="Count={Count}",     Name="",     Type="",</pre>	



```

        Target=,
        TargetTypeName="")]
[System.Reflection.DefaultMember("Item")]
[System.Diagnostics.DebuggerTypeProxy(ProxyTypeName="C1.LiveLinq.SequenceDebu
gView, C1.LiveLinq.4, Version=4.0.20151.455, Culture=neutral,
PublicKeyToken=2aa4ec5576d6c3ce",
        Target=,
        TargetTypeName="")]
```

```

public class View<T> : View, C1.LiveLinq.Indexing.IIndexedSource<T>,
C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

The type of the elements in the view.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

**C1.LiveLinq.LiveViews.View<T>**

[C1.Data.ClientView<T>](#)

[C1.LiveLinq.LiveViews.AggregationView<TSource,TResult>](#)

[C1.LiveLinq.LiveViews.GroupView<TKey,TElement>](#)

[C1.LiveLinq.LiveViews.OrderedView<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

## Overview

[C1.LiveLinq.LiveViews Namespace](#) : View<T> Class

The type of the elements in the view.

Represents a *live view*: a LINQ query result that supports two-way data binding and is kept up-to-date with base data.

# Object Model

View<T>

## Syntax

Visual Basic (Declaration)

```
<System.Diagnostics.DebuggerDisplayAttribute(Value="Count={Count}",  
    Name="",  
    Type="",  
    Target=,  
    TargetTypeName="")>  
<System.Reflection.DefaultMemberAttribute("Item")>  
<System.Diagnostics.DebuggerTypeProxyAttribute(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.LiveLinq.4, Version=4.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",  
    Target=,  
    TargetTypeName="")>  
Public Class View(Of T)  
    Inherits View  
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of T),  
    C1.LiveLinq.IObservableSource(Of T)
```

C#

```
[System.Diagnostics.DebuggerDisplay(Value="Count={Count}",  
    Name="",  
    Type="",  
    Target=,  
    TargetTypeName="")]  
[System.Reflection.DefaultMember("Item")]  
[System.Diagnostics.DebuggerTypeProxy(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.LiveLinq.4, Version=4.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",  
    Target=,  
    TargetTypeName="")]  
public class View<T> : View, C1.LiveLinq.Indexing.IIndexedSource<T>,  
    C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

T

The type of the elements in the view.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

**C1.LiveLinq.LiveViews.View<T>**

[C1.Data.ClientView<T>](#)

[C1.LiveLinq.LiveViews.AggregationView<TSource,TResult>](#)

[C1.LiveLinq.LiveViews.GroupView<TKey,TElement>](#)

[C1.LiveLinq.LiveViews.OrderedView<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Members](#)

[C1.LiveLinq.LiveViews Namespace](#)




## Members









[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.LiveViews Namespace](#) : [View<T>](#) Class

The following tables list the members exposed by [View<T>](#).


## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Indexes</a>	Gets the collection of indexes for this view.
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position.
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## Public Methods

	Name	Description
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> .

		(Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a <code>View&lt;dynamic&gt;</code> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">AttachAggregationView</a>	Overloaded.
≡	<a href="#">AttachView</a>	Overloaded.
≡	<a href="#">Concat</a>	Concatenation of two views.
≡	<a href="#">Contains</a>	Determines whether the view contains a specified item.
≡	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items.
≡	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function.
≡	<a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on equality of keys and groups the results.
≡	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence.
≡	<a href="#">Join&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on matching keys.
≡	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

⇒	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order.
⇒	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order.
⇒	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form.
⇒	<a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.
⇒	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">ToString</a>	Returns a string representing this view.
⇒	<a href="#">Union</a>	Set union of two views.
⇒	<a href="#">Where</a>	Filters the source view based on a predicate.

[Top](#)

## Public Events

	Name	Description
⚡	<a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed.

[Top](#)

## See Also

### Reference

[View<T> Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

## Methods

[C1.LiveLinq.LiveViews Namespace](#) : [View<T> Class](#)



For a list of all members of this type, see [View<T> members](#).

## Public Methods

	Name	Description
⇒	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a <a href="#">View&lt;dynamic&gt;</a> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">AttachAggregationView</a>	Overloaded.
⇒	<a href="#">AttachView</a>	Overloaded.
⇒	<a href="#">Concat</a>	Concatenation of two views.
⇒	<a href="#">Contains</a>	Determines whether the view contains a specified item.
⇒	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items.
⇒	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to

		a specified key selector function.
⇒	<a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on equality of keys and groups the results.
⇒	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence.
⇒	<a href="#">Join&lt;TInner,TKey,TResult&gt;</a>	Correlates the elements of two views based on matching keys.
⇒	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order.
⇒	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order.
⇒	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form.
⇒	<a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.
⇒	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">ToString</a>	Returns a string representing this view.



	<a href="#">Union</a>	Set union of two views.
	<a href="#">Where</a>	Filters the source view based on a predicate.

[Top](#)

## See Also

### Reference

[View<T> Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### AttachAggregationView Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : AttachAggregationView Method

## Overload List

Overload	Description
<a href="#">AttachAggregationView&lt;TResult&gt;(Object,Func&lt;View,AggregationView&lt;T,TResult&gt;&gt;)</a>	
<a href="#">AttachAggregationView&lt;TResult&gt;(Func&lt;View,AggregationView&lt;T,TResult&gt;&gt;)</a>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

[AttachAggregationView<TResult>\(Object,Func<View,AggregationView<T,TResult>>\) Method](#)

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [AttachAggregationView Method](#) :

[AttachAggregationView<TResult>\(Object,Func<View,AggregationView<T,TResult>>\) Method](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function AttachAggregationView(Of TResult)( _     ByVal subqueryId As System.Object, _     ByVal selector As System.Func(Of View(Of T),AggregationView(Of T,TResult))     _ ) As AggregationView(Of T,TResult)</pre>	
C#	
<pre>public AggregationView&lt;T,TResult&gt; AttachAggregationView&lt;TResult&gt;(     System.object subqueryId,     System.Func&lt;View&lt;T&gt;,AggregationView&lt;T,TResult&gt;&gt; selector )</pre>	

## Parameters

*subqueryId*

*selector*

## Type Parameters

*TResult*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

AttachAggregationView<TResult>(Func<View,AggregationView<T,TResult>>) Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [AttachAggregationView Method](#) :

AttachAggregationView<TResult>(Func<View,AggregationView<T,TResult>>) Method

## Syntax

Visual Basic (Declaration)	
----------------------------	--

```
Public Overloads Function AttachAggregationView(Of TResult)( _
    ByVal selector As System.Func(Of View(Of T),AggregationView(Of T,TResult))
-
) As AggregationView(Of T,TResult)

C#

public AggregationView<T,TResult> AttachAggregationView<TResult>(
    System.Func<View<T>,AggregationView<T,TResult>> selector
)
```

Parameters

*selector*

Type Parameters

*TResult*

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

- [View<T> Class](#)
- [View<T> Members](#)
- [Overload List](#)

AttachView Method

[C1.LiveInq.LiveViews Namespace](#) > [View<T> Class](#) : AttachView Method

Overload List

Overload	Description
<a href="#">AttachView&lt;TResult&gt;(Func&lt;View,View&lt;TResult&gt;&gt;&gt;)</a>	
<a href="#">AttachView&lt;TResult&gt;(Object,Func&lt;View,View&lt;TResult&gt;&gt;&gt;)</a>	

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[View<T> Class](#)  
[View<T> Members](#)

[AttachView<TResult>\(Func<View,View<TResult>>\) Method](#)  
[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [AttachView Method](#) :  
[AttachView<TResult>\(Func<View,View<TResult>>\) Method](#)

Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function AttachView(Of TResult)( _     ByVal selector As System.Func(Of View(Of T),View(Of TResult)) _ ) As View(Of TResult)</pre>	
C#	
<pre>public View&lt;TResult&gt; AttachView&lt;TResult&gt;(      System.Func&lt;View&lt;T&gt;,View&lt;TResult&gt;&gt; selector )</pre>	

Parameters

*selector*

Type Parameters

*TResult*

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[View<T> Class](#)  
[View<T> Members](#)  
[Overload List](#)

[AttachView<TResult>\(Object,Func<View,View<TResult>>\) Method](#)

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [AttachView Method](#) :

[AttachView<TResult>\(Object,Func<View,View<TResult>>\) Method](#)

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function AttachView(Of TResult)( _  
    ByVal subqueryId As System.Object, _  
    ByVal selector As System.Func(Of View(Of T),View(Of TResult)) _  
) As View(Of TResult)
```

C#

```
public View<TResult> AttachView<TResult>(  
    System.object subqueryId,  
    System.Func<View<T>,View<TResult>> selector  
)
```

### Parameters

*subqueryId*

*selector*

### Type Parameters

*TResult*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)  
[Overload List](#)

## Concat Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Concat Method

A collection (usually, a view) to concatenate to this view's collection of elements.

Concatenation of two views.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function Concat( _     ByVal second As IObservableSource(Of T) _ ) As View(Of T)</pre>	
C#	
<pre>public View&lt;T&gt; Concat(     IObservableSource&lt;T&gt; second )</pre>	

### Parameters

*second*

A collection (usually, a view) to concatenate to this view's collection of elements.

### Return Value

The view that contains first the elements of this view and then the elements of the parameter collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

## Contains Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Contains Method

The item to locate in the view.

Determines whether the view contains a specified item.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function Contains( _     ByVal item As T _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool Contains(     T item )</pre>	

### Parameters

*item*

The item to locate in the view.

### Return Value

**true** if the view contains the specified item; otherwise, **false**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

### GetEnumerator Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : GetEnumerator Method

Returns an enumerator that iterates through the view items.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<b>Public Function</b> GetEnumerator() <b>As</b> System.Collections.Generic.IEnumerator(Of T)	
C#	
<b>public</b> System.Collections.Generic.IEnumerator<T> GetEnumerator()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)

### GroupBy Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : GroupBy Method

Groups the elements of a view according to a specified key selector function.

## Overload List

Overload	Description
<a href="#">GroupBy&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;)</a>	Group s the elements of a view according to a specified key selector



	function.
<code>GroupBy&lt;TKey,TElement&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Expression&lt;Func&lt;T,TElement&gt;&gt; )</code>	Groups the elements of a view according to a specified key selector or function and projects the elements for each group by using a specified function.
<code>GroupBy&lt;TKey,TElement,TResult&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Expression&lt;Func&lt;T,TElement&gt;&gt;,Expression&lt;Func&lt;TKey,IEnumerable&lt;TElement&gt;,TResult&gt;&gt;)</code>	Groups the elements

	<p>nts of a view according to a specified key select or function and creates a result value from each group and its key. The elements of each group are projected by using a specified function</p>
--	---

	on.
--	-----

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)

[GroupBy<TKey>\(Expression<Func<T,TKey>>\) Method](#)  
[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [GroupBy Method](#) :  
[GroupBy<TKey>\(Expression<Func<T,TKey>>\) Method](#)

The type of the key returned by *keySelector*.

A function to extract the key for each element.

Groups the elements of a view according to a specified key selector function.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function GroupBy(Of TKey)( _     ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of T, TKey)) _ ) As View(Of GroupView(Of TKey, T))</pre>	
C#	
<pre>public View&lt;GroupView&lt;TKey, T&gt;&gt; GroupBy&lt;TKey&gt;(     System.Linq.Expressions.Expression&lt;Func&lt;T, TKey&gt;&gt; keySelector )</pre>	

### Parameters

*keySelector*

A function to extract the key for each element.

### Type Parameters

*TKey*

The type of the key returned by *keySelector*.

## Return Value

A view containing elements of type [GroupView<TKey,TElement>](#) each containing a key value and a view of the elements having that key value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

[GroupBy<TKey,TElement>\(Expression<Func<T,TKey>>,Expression<Func<T,TElement>>\)](#)  
Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [GroupBy Method](#) :

[GroupBy<TKey,TElement>\(Expression<Func<T,TKey>>,Expression<Func<T,TElement>>\)](#) Method

The type of the key returned by *keySelector*.

The type of the element to which elements of each group are projected.

A function to extract the key for each element.

A function to map each source element to a *TElement*.

Groups the elements of a view according to a specified key selector function and projects the elements for each group by using a specified function.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function GroupBy
    (Of TKey,TElement)( _
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of
T,TKey)), _
    ByVal elementSelector As System.Linq.Expressions.Expression(Of Func(Of
```

```
T,TElement)) _  
) As View(Of GroupView(Of TKey,TElement))
```

C#

```
public View<GroupView<TKey,TElement>> GroupBy<TKey,TElement>(  
    System.Linq.Expressions.Expression<Func<T,TKey>> keySelector,  
    System.Linq.Expressions.Expression<Func<T,TElement>> elementSelector  
)
```

## Parameters

*keySelector*

A function to extract the key for each element.

*elementSelector*

A function to map each source element to a *TElement*.

## Type Parameters

*TKey*

The type of the key returned by *keySelector*.

*TElement*

The type of the element to which elements of each group are projected.

## Return Value

A view containing elements of type [GroupView<TKey,TElement>](#) each containing a key value and a view of the elements projected (mapped) from the elements having that key value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

GroupBy<TKey,TElement,TResult>(Expression<Func<T,TKey>>,Expression<Func<T,TElement>>,Expression<Func<TKey,IEnumerable<TElement>,TResult>>) Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [GroupBy Method](#) :

GroupBy<TKey,TElement,TResult>(Expression<Func<T,TKey>>,Expression<Func<T,TElement>>,Expression<Func<TKey,IEnumerable<TElement>,TResult>>) Method

The type of the key returned by *keySelector*.

The type of the elements in groups.

The type of the result value returned by *resultSelector*.

A function to extract the key for each element.

A function to map each source element to an element in the **System.Linq.IGrouping`2**.

A function to create a result value from each group.

Groups the elements of a view according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function GroupBy
    (Of TKey,TElement,TResult)( _
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of
T,TKey)), _
    ByVal elementSelector As System.Linq.Expressions.Expression(Of Func(Of
T,TElement)), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
TKey,IEnumerable(Of TElement),TResult)) _
) As View(Of TResult)
```

C#

```
public View<TResult> GroupBy<TKey,TElement,TResult>(
    System.Linq.Expressions.Expression<Func<T,TKey>> keySelector,
    System.Linq.Expressions.Expression<Func<T,TElement>> elementSelector,

System.Linq.Expressions.Expression<Func<TKey,IEnumerable<TElement>,TResult>>
resultSelector
)
```

## Parameters

*keySelector*

A function to extract the key for each element.

*elementSelector*

A function to map each source element to an element in the **System.Linq.IGrouping`2**.

*resultSelector*

A function to create a result value from each group.

## Type Parameters

*TKey*

The type of the key returned by *keySelector*.

*TElement*

The type of the elements in groups.

*TResult*

The type of the result value returned by *resultSelector*.

## Return Value

A view of elements of type *TResult* where each element represents a projection over a group and its key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

## GroupJoin<TInner,TKey,TResult> Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : [GroupJoin<TInner,TKey,TResult> Method](#)

The type of the elements of the view to join with this view.

The type of the keys returned by the key selector functions.

The type of the result elements.

The collection (usually, a view) to join to this view.

A function to extract the join key from each element of this view.

A function to extract the join key from each element of the second view.

A function to create a result view from an element from this view and a collection of matching elements from the second view.

Correlates the elements of two views based on equality of keys and groups the results.

## Syntax

Visual Basic (Declaration)

```
Public Function GroupJoin  
    (Of TInner,TKey,TResult)( _  
    ByVal inner As IObservableSource(Of TInner), _  
    ByVal outerKeySelector As System.Linq.Expressions.Expression(Of Func(Of  
T,TKey)), _  
    ByVal innerKeySelector As System.Linq.Expressions.Expression(Of Func(Of  
TInner,TKey)), _  
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of  
T,GroupView(Of TKey,TInner),TResult)) _  
    ) As View(Of TResult)
```

C#

```
public View<TResult> GroupJoin<TInner,TKey,TResult>(  
    IObservableSource<TInner> inner,  
    System.Linq.Expressions.Expression<Func<T,TKey>> outerKeySelector,  
    System.Linq.Expressions.Expression<Func<TInner,TKey>> innerKeySelector,  
    System.Linq.Expressions.Expression<Func<T,GroupView<TKey,TInner>,TResult>>  
resultSelector  
)
```

## Parameters

*inner*

The collection (usually, a view) to join to this view.

*outerKeySelector*

A function to extract the join key from each element of this view.



*innerKeySelector*

A function to extract the join key from each element of the second view.

*resultSelector*

A function to create a result view from an element from this view and a collection of matching elements from the second view.

## Type Parameters

*TInner*

The type of the elements of the view to join with this view.

*TKey*

The type of the keys returned by the key selector functions.

*TResult*

The type of the result elements.

## Return Value

A view containing elements of type *TResult* that are obtained by performing a grouped join on two views.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

## IndexOf Method

[C1.LiveInq.LiveViews Namespace](#) > [View<T> Class](#) : IndexOf Method

The object to locate in the view.

Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence.

## Syntax

Visual Basic (Declaration)

```
Public Function IndexOf( _  
    ByVal item As T _  
) As System.Integer
```

C#

```
public System.int IndexOf(  
    T item  
)
```

### Parameters

*item*

The object to locate in the view.

### Return Value

The zero-based ordinal position of the first occurrence of *item* in the view, if found; otherwise, -1.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

### Join<TInner,TKey,TResult> Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Join<TInner,TKey,TResult> Method

The type of the elements of the view to join with this view.

The type of the keys returned by the key selector functions.

The type of the result elements.

The collection (usually, a view) to join to this view.

A function to extract the join key from each element of this view.

A function to extract the join key from each element of the second view.

A function to create a result element from two matching elements.

Correlates the elements of two views based on matching keys.

## Syntax

Visual Basic (Declaration)

```
Public Function Join
    (Of TInner, TKey, TResult)( _
    ByVal inner As IObservableSource(Of TInner), _
    ByVal outerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
T, TKey)), _
    ByVal innerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
TInner, TKey)), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
T, TInner, TResult))) _
) As View(Of TResult)
```

C#

```
public View<TResult> Join<TInner, TKey, TResult>(
    IObservableSource<TInner> inner,
    System.Linq.Expressions.Expression<Func<T, TKey>> outerKeySelector,
    System.Linq.Expressions.Expression<Func<TInner, TKey>> innerKeySelector,
    System.Linq.Expressions.Expression<Func<T, TInner, TResult>> resultSelector
)
```

## Parameters

*inner*

The collection (usually, a view) to join to this view.

*outerKeySelector*

A function to extract the join key from each element of this view.

*innerKeySelector*

A function to extract the join key from each element of the second view.

*resultSelector*

A function to create a result element from two matching elements.

## Type Parameters

*TInner*

The type of the elements of the view to join with this view.

*TKey*

The type of the keys returned by the key selector functions.

*TResult*

The type of the result elements.

## Return Value

A view containing elements of type *TResult* that are obtained by performing an inner join on two views.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

### OrderBy<TKey>(Expression<Func<T,TKey>>) Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : OrderBy<TKey>(Expression<Func<T,TKey>>)  
Method

The type of the key returned by *keySelector*.

A function to extract a key from an element.

Sorts the elements of a view in ascending order.

## Syntax

Visual Basic (Declaration)

```
Public Function OrderBy(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of
```

```
T, TKey)) _  
) As OrderedView(Of T)
```

C#

```
public OrderedView<T> OrderBy<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector  
)
```

## Parameters

*keySelector*

A function to extract a key from an element.

## Type Parameters

*TKey*

The type of the key returned by *keySelector*.

## Return Value

A view whose elements are sorted according to a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

## OrderByDescending<TKey> Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : [OrderByDescending<TKey> Method](#)

The type of the key returned by *keySelector*.

A function to extract a key from an element.

Sorts the elements of a view in descending order.

## Syntax

## Visual Basic (Declaration)

```
Public Function OrderByDescending(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
) As OrderedView(Of T)
```

## C#

```
public OrderedView<T> OrderByDescending<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector  
)
```

## Parameters

*keySelector*

A function to extract a key from an element.

## Type Parameters

*TKey*

The type of the key returned by *keySelector*.

## Return Value

A view whose elements are sorted in descending order according to a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

## Select<TResult> Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Select<TResult> Method

The type of the value returned by *selector*.

A transform function to apply to each element.

Projects each element of a view into a new form.

## Syntax

Visual Basic (Declaration)

```
Public Function Select(Of TResult)( _  
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of  
T, TResult))) _  
) As View(Of TResult)
```

C#

```
public View<TResult> Select<TResult>(  
    System.Linq.Expressions.Expression<Func<T, TResult>> selector  
)
```

### Parameters

*selector*

A transform function to apply to each element.

### Type Parameters

*TResult*

The type of the value returned by *selector*.

### Return Value

A view whose elements are the result of invoking the transform function on each element of this view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

### SelectMany Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : SelectMany Method

Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.

## Overload List

Overload	Description
<a href="#">SelectMany&lt;TCollection,TResult&gt;(Expression&lt;Func&lt;T,IEnumerableSource&lt;TCollection&gt;&gt;&gt;,Expression&lt;Func&lt;T,TCollection,TResult&gt;&gt;)</a>	Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each



	<p>element therein.</p>
<p><a href="#">SelectMany&lt;TResult&gt;(Expression&lt;Func&lt;T,IEnumerableSource&lt;TResult&gt;&gt;&gt;)</a></p>	<p>Projects each element of this view to a collection of <i>TResult</i> and flattens the resulting collections into one view.</p>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

SelectMany<TCollection,TResult>(Expression<Func<T,IEnumerableSource<TCollection>>>,Expression<Func<T,TCollection,TResult>>) Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [SelectMany Method](#) :

SelectMany<TCollection,TResult>(Expression<Func<T,IEnumerableSource<TCollection>>>,Expression<Func<T,TCollection,TResult>>) Method

The type of the intermediate elements collected by *collectionSelector*.

The type of the elements of the resulting view.

A transform function to apply to each element of this view.

A transform function to apply to each element of the intermediate collection.

Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function SelectMany     (Of TCollection,TResult)( _     ByVal collectionSelector As System.Linq.Expressions.Expression(Of Func(Of T,IEnumerableSource(Of TCollection))), _     ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of T,TCollection,TResult)) _ ) As View(Of TResult)</pre>	
C#	
<pre>public View&lt;TResult&gt; SelectMany&lt;TCollection,TResult&gt;(     System.Linq.Expressions.Expression&lt;Func&lt;T,IEnumerableSource&lt;TCollection&gt;&gt;&gt; collectionSelector,     System.Linq.Expressions.Expression&lt;Func&lt;T,TCollection,TResult&gt;&gt; resultSelector )</pre>	

### Parameters

*collectionSelector*

A transform function to apply to each element of this view.

*resultSelector*

A transform function to apply to each element of the intermediate collection.

### Type Parameters

*TCollection*

The type of the intermediate elements collected by *collectionSelector*.

*TResult*

The type of the elements of the resulting view.

## Return Value

A view whose elements are the result of invoking the one-to-many transform function *collectionSelector* on each element of this view and then mapping each of those collection elements and their corresponding source element to a result element.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

SelectMany<TResult>(Expression<Func<T,IEnumerableSource<TResult>>>) Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [SelectMany Method](#) :

SelectMany<TResult>(Expression<Func<T,IEnumerableSource<TResult>>>) Method

The type of the elements of the resulting view.

A transform function to apply to each element.

Projects each element of this view to a collection of *TResult* and flattens the resulting collections into one view.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function SelectMany(Of TResult)( _  
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of  
T,IEnumerableSource(Of TResult))) _
```

```
) As View(Of TResult)
```

C#

```
public View<TResult> SelectMany<TResult>(
    System.Linq.Expressions.Expression<Func<T, IObservableSource<TResult>>>
    selector
)
```

## Parameters

*selector*

A transform function to apply to each element.

## Type Parameters

*TResult*

The type of the elements of the resulting view.

## Return Value

A view whose elements are the result of invoking the one-to-many transform function on each element of this view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

## ToString Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : ToString Method

Returns a string representing this view.

## Syntax

Visual Basic (Declaration)

```
Public Overrides Function ToString() As System.String
```

```
C#
```

```
public override System.string ToString()
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

### Union Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Union Method

A collection (usually, a view) whose distinct elements form the second set for the union.

Set union of two views.

## Syntax

Visual Basic (Declaration)

```
Public Function Union( _  
    ByVal second As IObservableSource(Of T) _  
) As View(Of T)
```

```
C#
```

```
public View<T> Union(  
    IObservableSource<T> second  
)
```

### Parameters

*second*

A collection (usually, a view) whose distinct elements form the second set for the union.

### Return Value

The view that contains the elements from both input views, excluding duplicates.

## Remarks

This method excludes duplicates from the result set. This is different behavior to the [Concat](#) method, which returns all the elements in the input sequences including duplicates.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

### Where Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Where Method

A function to test each element for a condition.

Filters the source view based on a predicate.

## Syntax

Visual Basic (Declaration)

```
Public Function Where( _  
    ByVal predicate As System.Linq.Expressions.Expression(Of Func(Of  
T, Boolean)) _  
) As View(Of T)
```

C#

```
public View<T> Where(  
    System.Linq.Expressions.Expression<Func<T, bool>> predicate  
)
```

### Parameters

*predicate*

A function to test each element for a condition.

### Return Value

A view that contains elements of this view that satisfy the condition.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)








[View<T> Members](#)





## Properties

[C1.LiveLinq.LiveViews Namespace](#) : [View<T> Class](#)

For a list of all members of this type, see [View<T> members](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DataBindingMode</a>	Gets or sets the data binding mode for this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Indexes</a>	Gets the collection of indexes for this view.
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position.

	<b>MaintenanceMode</b>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<b>MoveToFirstOnReset</b>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<b>Order</b>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<b>Transaction</b>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## See Also

### Reference

[View<T> Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### Indexes Property

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Indexes Property

Gets the collection of indexes for this view.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> Indexes <b>As</b> <a href="#">IndexCollection(Of T)</a>	
C#	
<b>public</b> <a href="#">IndexCollection&lt;T&gt;</a> Indexes { <b>get</b> ;}	



## Remarks

Live views can be indexed, just like other LiveLinq data sources, to optimize search operations over their data. For an example of an index over a view, see [Live Views How To: Create Views Based on Other Views and Create Indexes on Views](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

[Indexes Property](#)

### Item Property

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Item Property

The zero-based ordinal position of the view item.

Gets the view item (element) at the specified ordinal position.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Default Property Item( _  
    ByVal ordinal As System.Integer _  
) As T
```

C#

```
public T this[  
    System.int ordinal  
]; {get;}
```

### Parameters

*ordinal*

The zero-based ordinal position of the view item.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference


[View<T> Class](#)  
[View<T> Members](#)

### Events

[C1.LiveLinq.LiveViews Namespace](#) : [View<T> Class](#)

For a list of all members of this type, see [View<T> members](#).

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed.

[Top](#)

## See Also

### Reference

[View<T> Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

### Changed Event

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Changed Event

Occurs after an item of the view or the entire view has changed.

## Syntax

Visual Basic (Declaration)	
<code>Public Event Changed As System.EventHandler(Of SourceChangeEventArgs(Of T))</code>	
C#	
<code>public event System.EventHandler&lt;SourceChangeEventArgs&lt;T&gt;&gt; Changed</code>	

## Event Data

The event handler receives an argument of type [SourceChangeEventArgs<T>](#) containing data related to this event. The following **SourceChangeEventArgs<T>** properties provide information specific to this event.

Property	Description
<a href="#">ChangeType</a>	Gets the type of change.
<a href="#">Item</a>	Gets the object that is being changed.
<a href="#">Ordinal</a>	Gets the ordinal position of the collection item that is being changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)

## ViewRow

[C1.LiveLinq.LiveViews Namespace](#) : ViewRow Class

Represents a view element (item) for the purposes of dynamic, programmatic access to its properties and data binding.

## Object Model

ViewRow

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Reflection.DefaultMemberAttribute("Item")&gt; Public MustInherit Class ViewRow     Inherits C1.LiveLinq.Collections.IndexableObject</pre>	
C#	

```
[System.Reflection.DefaultMember("Item")]  
public abstract class ViewRow : C1.LiveLinq.Collections.IndexableObject
```

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.Collections.IndexableObject](#)

**C1.LiveLinq.LiveViews.ViewRow**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

[Rows Property](#)

## Overview

[C1.LiveLinq.LiveViews Namespace](#) : ViewRow Class

Represents a view element (item) for the purposes of dynamic, programmatic access to its properties and data binding.

## Object Model

ViewRow

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>  
Public MustInherit Class ViewRow  
    Inherits C1.LiveLinq.Collections.IndexableObject
```

C#

```
[System.Reflection.DefaultMember("Item")]  
public abstract class ViewRow : C1.LiveLinq.Collections.IndexableObject
```

# Inheritance Hierarchy

System.Object

[C1.LiveLinq.Collections.IndexableObject](#)

**C1.LiveLinq.LiveViews.ViewRow**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

[Rows Property](#)






## Members

[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.LiveViews Namespace](#) : ViewRow Class






The following tables list the members exposed by [ViewRow](#).

## Public Properties

	Name	Description
	<a href="#">Item</a>	Overloaded. Gets or sets a value of the specified view property.
	<a href="#">RowState</a>	Gets the state of a view row with regard to edit, add and delete operations if they are performed directly on the view.
	<a href="#">Tag</a>	Gets or sets user-supplied data associated with the view item.
	<a href="#">Value</a>	Gets the view element (item) represented by this <a href="#">ViewRow</a> object.
	<a href="#">View</a>	Gets the view to which the <a href="#">ViewRow</a> belongs.


[Top](#)

## Public Methods

	Name	Description
	<a href="#">BeginEdit</a>	Puts the <a href="#">ViewRow</a> into edit mode.
	<a href="#">CancelEdit</a>	Cancels the edit occurring on the row.
	<a href="#">Delete</a>	Deletes a view item.
	<a href="#">EndEdit</a>	Ends the edit occurring on the row.
	<a href="#">ToString</a>	Gets the string representing this view row.


[Top](#)

## Protected Methods

	Name	Description
	<a href="#">OnPropertyChanged</a>	Overridden. Raises the <b>System.ComponentModel.INotifyPropertyChanged.PropertyChanged</b> event.

[Top](#)

## Public Events

	Name	Description
	<a href="#">PropertyChanged</a>	Occurs when a property value changes, after it has been changed.

[Top](#)

## See Also

### Reference

[ViewRow Class](#)

[C1.LiveInq.LiveViews Namespace](#)






[Rows Property](#)

## Methods

[C1.LiveLinq.LiveViews Namespace](#) : ViewRow Class


For a list of all members of this type, see [ViewRow members](#).

## Public Methods

	Name	Description
	<a href="#">BeginEdit</a>	Puts the <a href="#">ViewRow</a> into edit mode.
	<a href="#">CancelEdit</a>	Cancels the edit occurring on the row.
	<a href="#">Delete</a>	Deletes a view item.
	<a href="#">EndEdit</a>	Ends the edit occurring on the row.
	<a href="#">ToString</a>	Gets the string representing this view row.

[Top](#)

## Protected Methods

	Name	Description
	<a href="#">OnPropertyChanged</a>	Overridden. Raises the <b>System.ComponentModel.INotifyPropertyChanged.PropertyChanged</b> event.

[Top](#)

## See Also

### Reference

[ViewRow Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

[Rows Property](#)

### BeginEdit Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : [BeginEdit\(\)](#) Method

Puts the [ViewRow](#) into edit mode.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> BeginEdit()	
C#	
<b>public void</b> BeginEdit()	

## Remarks

In edit mode, events and notifications are temporarily suspended, letting the user make changes to more than one property without triggering validation rules.

Edit mode is a standard feature of .NET data binding mechanism, supported by every data source implementing the **System.ComponentModel.IEditableObject** interface. For detailed explanation, see, for example, **System.Data.DataRowView** in .NET Framework documentation.

While a view item is in edit mode, changes made to its updatable properties directly in the view are not propagated to the corresponding base data properties until the edit operation is completed by a call to [EndEdit](#) (see [View.IsReadOnly](#) about updatability of view element properties directly in the view).

If you change base data (source) properties, those changes are propagated to this view and other views depending on that base data according to the normal view maintenance process, regardless of whether the view row is in edit mode or not.

Many-to-one relations between view properties are maintained automatically on changes made to updatable properties directly in the view, regardless of whether the view row is in edit mode or not. For example, if you set a CustomerID directly in the view, CustomerName will change accordingly, even if you do it in edit mode.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)



## CancelEdit Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : CancelEdit Method

Cancels the edit occurring on the row.

## Syntax

Visual Basic (Declaration)	
<b>Public Overridable Sub</b> CancelEdit()	
C#	
<b>public virtual void</b> CancelEdit()	

## Remarks

If the user set some of the updatable properties of the row while it was in edit mode, the changes to those properties are rolled back and not propagated to the corresponding base data properties.

See the [BeginEdit](#) method for more information.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

## Delete Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : Delete Method

Deletes a view item.

## Syntax

Visual Basic (Declaration)	
<b>Public Overridable Sub</b> Delete()	

C#	
----	--

<code>public virtual void Delete()</code>	
---	--

## Remarks

Deleting a view item is an update operation on the view. As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)), and it results in updating (in this case, deleting an item from) one of the view's base data collections.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

### EndEdit Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : EndEdit Method

Ends the edit occurring on the row.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<code>Public Overridable Sub EndEdit()</code>	
---	--

C#	
----	--

<code>public virtual void EndEdit()</code>	
--	--

## Remarks

If the user set some of the updatable properties of the row while it was in edit mode, the changes to those properties are propagated to the corresponding base data properties at this point.

See the [BeginEdit](#) method for more information.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

### OnPropertyChanged Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : OnPropertyChanged Method

The name of the property that is changed.

Raises the **System.ComponentModel.INotifyPropertyChanged.PropertyChanged** event.

## Syntax

Visual Basic (Declaration)

```
Protected Overrides Sub OnPropertyChanged( _  
    ByVal propertyName As System.String _  
)
```

C#

```
protected override void OnPropertyChanged(  
    System.string propertyName  
)
```

### Parameters

*propertyName*

The name of the property that is changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ViewRow Class](#)

[ViewRow Members](#)

## ToString Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : ToString Method

Gets the string representing this view row.

## Syntax

Visual Basic (Declaration)

```
Public Overrides Function ToString() As System.String
```

C#

```
public override System.string ToString()
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)



[ViewRow Members](#)




## Properties

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRow Class](#)

For a list of all members of this type, see [ViewRow members](#).

## Public Properties

	Name	Description
	<a href="#">Item</a>	Overloaded. Gets or sets a value of the specified view property.
	<a href="#">RowState</a>	Gets the state of a view row with regard to edit, add and delete

		operations if they are performed directly on the view.
	<a href="#">Tag</a>	Gets or sets user-supplied data associated with the view item.
	<a href="#">Value</a>	Gets the view element (item) represented by this <a href="#">ViewRow</a> object.
	<a href="#">View</a>	Gets the view to which the <a href="#">ViewRow</a> belongs.

[Top](#)

## See Also

### Reference

[ViewRow Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

[Rows Property](#)

### Item Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : Item Property

Gets or sets a value of the specified view property.

## Overload List

Overload	Description
<a href="#">Item(Int32)</a>	Gets or sets a value of the specified view property.
<a href="#">Item(String)</a>	Gets or sets a value of the specified view property.

## Remarks

Setting a view property is an update operation on the view. As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)), and it results in updating one of the view's base data collections.

Only updatable properties can be set. An attempt to set a read-only property results in an exception. See [View.IsReadOnly](#) for more details.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

#### Item(Int32) Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) > [Item Property](#) : Item(Int32) Property

The ordinal position of the property in the collection of public properties of the type of the view element.

Gets or sets a value of the specified view property.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Property Item( _  
    ByVal propertyOrdinal As System.Integer _  
) As System.Object
```

C#

```
public System.object Item(  
    System.int propertyOrdinal  
) {get; set;}
```

### Parameters

*propertyOrdinal*

The ordinal position of the property in the collection of public properties of the type of the view element.

### Property Value

The value of the property.

## Remarks

Setting a view property is an update operation on the view. As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if

the view is not read-only (see [View.IsReadOnly](#)), and it results in updating one of the view's base data collections.

Only updatable properties can be set. An attempt to set a read-only property results in an exception. See [View.IsReadOnly](#) for more details.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

[Overload List](#)

Item(String) Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) > [Item Property](#) : Item(String) Property

The name of the property.

Gets or sets a value of the specified view property.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Property Item( _  
    ByVal propertyName As System.String _  
) As System.Object
```

C#

```
public System.object Item(  
    System.string propertyName  
) {get; set;}
```

### Parameters

*propertyName*

The name of the property.

### Property Value

The value of the property.

## Remarks

Setting a view property is an update operation on the view. As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)), and it results in updating one of the view's base data collections.

Only updatable properties can be set. An attempt to set a read-only property results in an exception. See [View.IsReadOnly](#) for more details.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)  
[ViewRow Members](#)  
[Overload List](#)

### RowState Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : RowState Property

Gets the state of a view row with regard to edit, add and delete operations if they are performed directly on the view.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> RowState <b>As</b> ViewRowState	
C#	
<b>public</b> ViewRowState RowState { <b>get</b> ;}	

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)  
[ViewRow Members](#)

### Tag Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : Tag Property

Gets or sets user-supplied data associated with the view item.

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> Tag <b>As</b> System.Object	
C#	
<b>public</b> System.object Tag { <b>get</b> ; <b>set</b> ;}	

## Remarks

Use this property to store any object you want to associate in your code with the view item that you need to access quickly.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)  
[ViewRow Members](#)

### Value Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : Value Property

Gets the view element (item) represented by this [ViewRow](#) object.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Value As System.Object</code>	
C#	
<code>public System.object Value {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

### View Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : View Property

Gets the view to which the [ViewRow](#) belongs.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property View As View</code>	
C#	
<code>public View View {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference


[ViewRow Class](#)  
[ViewRow Members](#)

## Events

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRow Class](#)

For a list of all members of this type, see [ViewRow members](#).

## Public Events

	Name	Description
	<a href="#">PropertyChanged</a>	Occurs when a property value changes, after it has been changed.

[Top](#)

## See Also

### Reference

[ViewRow Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)  
[Rows Property](#)

### PropertyChanged Event

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : PropertyChanged Event

Occurs when a property value changes, after it has been changed.

## Syntax

Visual Basic (Declaration)	
<b>Public Shadows Event</b> PropertyChanged <b>As</b> System.ComponentModel.PropertyChangedEventArgs	
C#	
<b>public new event</b> System.ComponentModel.PropertyChangedEventArgs PropertyChanged	

## Event Data

The event handler receives an argument of type `System.ComponentModel.PropertyChangedEventArgs` containing data related to this event. The following **PropertyChangedEventArgs** properties provide information specific to this event.

Property	Description
<b>PropertyName</b>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

## ViewRowAddingEventArgs

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowAddingEventArgs Class

Provides data for the [ViewRowCollection.ViewRowAdding](#) event.

## Object Model

ViewRowAddingEventArgs

## Syntax

Visual Basic (Declaration)	
<pre><b>Public Class</b> ViewRowAddingEventArgs     <b>Inherits</b> System.EventArgs</pre>	
C#	
<pre><b>public class</b> ViewRowAddingEventArgs : System.EventArgs</pre>	

## Inheritance Hierarchy

System.Object

System.EventArgs

**C1.LiveLinq.LiveViews.ViewRowAddingEventArgs**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[ViewRowAddingEventArgs Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)

Overview

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowAddingEventArgs Class

Provides data for the [ViewRowCollection.ViewRowAdding](#) event.

Object Model

ViewRowAddingEventArgs

Syntax

Visual Basic (Declaration)	
<code>Public Class ViewRowAddingEventArgs</code> <code>    Inherits System.EventArgs</code>	
C#	
<code>public class ViewRowAddingEventArgs : System.EventArgs</code>	

Inheritance Hierarchy

System.Object  
    System.EventArgs  
        **C1.LiveLinq.LiveViews.ViewRowAddingEventArgs**

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference


## Members

[Properties](#)

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRowAddingEventArgs](#) Class

The following tables list the members exposed by [ViewRowAddingEventArgs](#).

### Public Properties

	Name	Description
	<a href="#">Row</a>	Gets the new view row that has just been added to <a href="#">ViewRowCollection</a> .

[Top](#)

## See Also

### Reference


[ViewRowAddingEventArgs Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Properties

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRowAddingEventArgs](#) Class

For a list of all members of this type, see [ViewRowAddingEventArgs members](#).

### Public Properties

	Name	Description
	<a href="#">Row</a>	Gets the new view row that has just been added to <a href="#">ViewRowCollection</a> .

[Top](#)

## See Also

### Reference

[ViewRowAddingEventArgs Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Row Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowAddingEventArgs Class](#) : Row Property

Gets the new view row that has just been added to [ViewRowCollection](#).

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property Row As ViewRow
```

C#

```
public ViewRow Row {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowAddingEventArgs Class](#)

[ViewRowAddingEventArgs Members](#)

## ViewRowCollection

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowCollection Class

Represents a collection of [ViewRow](#) objects used for programmatic access to view elements (items) and for data binding.

## Object Model

**ViewRowCollection**

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>  
<System.Diagnostics.DebuggerDisplayAttribute(Value="Count = {Count}",  
    Name="",
```

```

        Type="",
        Target=,
        TargetTypeName="")>
<System.Diagnostics.DebuggerTypeProxyAttribute(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.LiveLinq.4, Version=4.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",
        Target=,
        TargetTypeName="")>
Public MustInherit Class ViewRowCollection
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of ViewRow),
C1.LiveLinq.IObservableSource(Of ViewRow)

```

C#

```

[System.Reflection.DefaultMember("Item")]
[System.Diagnostics.DebuggerDisplay(Value="Count = {Count}",
    Name="",
    Type="",
    Target=,
    TargetTypeName="")]
[System.Diagnostics.DebuggerTypeProxy(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.LiveLinq.4, Version=4.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",
    Target=,
    TargetTypeName="")]
public abstract class ViewRowCollection :
C1.LiveLinq.Indexing.IIndexedSource<ViewRow>,
C1.LiveLinq.IObservableSource<ViewRow>

```

## Remarks

A **ViewRowCollection** is owned by a view, see [View.Rows](#).

The collection of *view rows* ([ViewRow](#) objects) is always synchronized with the collection of view elements.

[ViewRow](#) objects provide programmatic access to view elements and their properties. Also, the [View.Rows](#) collection serves as the data source for data binding, when you bind a control or another client to a view.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.ViewRowCollection**



## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

## Overview

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowCollection Class

Represents a collection of [ViewRow](#) objects used for programmatic access to view elements (items) and for data binding.

## Object Model

ViewRowCollection

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>
<System.Diagnostics.DebuggerDisplayAttribute(Value="Count = {Count}",
    Name="",
    Type="",
    Target=,
    TargetTypeName="")>
<System.Diagnostics.DebuggerTypeProxyAttribute(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.LiveLinq.4, Version=4.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",
    Target=,
    TargetTypeName="")>
Public MustInherit Class ViewRowCollection
    Implements C1.LiveLinq.Indexing.IIndexedSource(Of ViewRow),
    C1.LiveLinq.IObservableSource(Of ViewRow)
```

C#

```
[System.Reflection.DefaultMember("Item")]
[System.Diagnostics.DebuggerDisplay(Value="Count = {Count}",
    Name="",
    Type="",
    Target=,
    TargetTypeName="")]
[System.Diagnostics.DebuggerTypeProxy(ProxyTypeName="C1.LiveLinq.SequenceDebu
gView, C1.LiveLinq.4, Version=4.0.20151.455, Culture=neutral,
PublicKeyToken=2aa4ec5576d6c3ce",
    Target=,
    TargetTypeName="")]
public abstract class ViewRowCollection :
C1.LiveLinq.Indexing.IIndexedSource<ViewRow>,
C1.LiveLinq.IObservableSource<ViewRow>
```

## Remarks

A **ViewRowCollection** is owned by a view, see [View.Rows](#).

The collection of *view rows* ([ViewRow](#) objects) is always synchronized with the collection of view elements.

[ViewRow](#) objects provide programmatic access to view elements and their properties. Also, the [View.Rows](#) collection serves as the data source for data binding, when you bind a control or another client to a view.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.ViewRowCollection**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Members](#)












[C1.LiveLinq.LiveViews Namespace](#)

## Members

[Properties](#) [Methods](#) [Events](#)










The following tables list the members exposed by [ViewRowCollection](#).

## Public Properties

	Name	Description
	<a href="#">AllowClear</a>	Gets a value indicating whether the <a href="#">Clear</a> operation is allowed on the view directly.
	<a href="#">AllowEdit</a>	Gets a value indicating whether modifying property values is allowed on the view directly.
	<a href="#">AllowNew</a>	Gets a value indicating whether the <a href="#">CreateRow</a> operation is allowed on the view directly.
	<a href="#">AllowRemove</a>	Gets a value indicating whether deleting rows is allowed on the view directly.
	<a href="#">Count</a>	Gets the number of elements in the view.
	<a href="#">GroupDescriptions</a>	Gets a collection of <b>System.ComponentModel.GroupDescription</b> objects that describe how the items in the collection are grouped.
	<a href="#">Groups</a>	Gets the top-level groups.
	<a href="#">Indexes</a>	Gets the collection of indexes for this view allowing to search for <a href="#">ViewRow</a> objects.
	<a href="#">Item</a>	Gets the view row at the specified ordinal position.
	<a href="#">Properties</a>	Returns the collection of properties available in the view element type to programmatic access through <a href="#">ViewRow</a> and to data binding.
	<a href="#">SortDescriptions</a>	Gets a collection of <b>System.ComponentModel.SortDescription</b> objects that describe how the items in the collection are sorted.



[Top](#)

## Public Methods

	Name	Description
≡	 <a href="#">Clear</a>	Deletes all view elements.
≡	 <a href="#">Contains</a>	Determines whether the <a href="#">ViewRowCollection</a> contains a specific view row.
≡	 <a href="#">CreateRow</a>	Creates a new item directly in the view, and a view row associated with it, and adds it to the <a href="#">ViewRowCollection</a> .
≡	 <a href="#">DeferRefresh</a>	Enters a defer cycle that you can use to merge changes to the view and delay automatic refresh.
≡	 <a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the collection.
≡	 <a href="#">GetItemProperties</a>	Returns the collection of properties available in the view element type to programmatic access through <a href="#">ViewRow</a> and to data binding.
≡	 <a href="#">IndexOf</a>	Determines the ordinal position of a specific view row in the <a href="#">ViewRowCollection</a> .
≡	 <a href="#">Remove</a>	Deletes the specified view item.
≡	 <a href="#">RemoveAt</a>	Deletes the view row at a specified ordinal position in <a href="#">ViewRowCollection</a> .

[Top](#)

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after a view row has changed.
	<a href="#">ViewRowAdding</a>	Occurs after a new view row is created so it can be populated with default values.

[Top](#)

## See Also

### Reference

[ViewRowCollection Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### Methods

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRowCollection Class](#)

For a list of all members of this type, see [ViewRowCollection members](#).

### Public Methods

	Name	Description
⇒	<a href="#">Clear</a>	Deletes all view elements.
⇒	<a href="#">Contains</a>	Determines whether the <a href="#">ViewRowCollection</a> contains a specific view row.
⇒	<a href="#">CreateRow</a>	Creates a new item directly in the view, and a view row associated with it, and adds it to the <a href="#">ViewRowCollection</a> .
⇒	<a href="#">DeferRefresh</a>	Enters a defer cycle that you can use to merge changes to the view and delay automatic refresh.
⇒	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the collection.
⇒	<a href="#">GetItemProperties</a>	Returns the collection of properties available in the view element type to programmatic access through <a href="#">ViewRow</a> and to data binding.
⇒	<a href="#">IndexOf</a>	Determines the ordinal position of a specific view row in the <a href="#">ViewRowCollection</a> .
⇒	<a href="#">Remove</a>	Deletes the specified view item.
⇒	<a href="#">RemoveAt</a>	Deletes the view row at a specified ordinal position in <a href="#">ViewRowCollection</a> .

[Top](#)

# See Also

## Reference

[ViewRowCollection Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Clear Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Clear Method

Deletes all view elements.

# Syntax

Visual Basic (Declaration)	
<code>Public Sub Clear()</code>	
C#	
<code>public void Clear()</code>	

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ViewRowCollection Class](#)  
[ViewRowCollection Members](#)

## Contains Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Contains Method

The view row to locate in the collection.

Determines whether the [ViewRowCollection](#) contains a specific view row.

# Syntax

Visual Basic (Declaration)	
<code>Public Function Contains( _</code>	

<pre> ByVal row As ViewRow _ ) As System.Boolean </pre>	
C#	
<pre> public System.bool Contains(     ViewRow row ) </pre>	

## Parameters

*row*

The view row to locate in the collection.

## Return Value

**true** if the view row is found in the collection; otherwise, **false**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

## CreateRow Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : CreateRow Method

Creates a new item directly in the view, and a view row associated with it, and adds it to the [ViewRowCollection](#).

## Syntax

Visual Basic (Declaration)	
<pre> Public Function CreateRow() As ViewRow </pre>	
C#	
<pre> public ViewRow CreateRow() </pre>	

## Return Value

The newly created view row.

## Remarks

Creating a new row is an update operation on the view. As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)), and it results in updating (in this case, adding a new item to) one of the view's base data collections.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)  
[ViewRowCollection Members](#)

### DeferRefresh Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : DeferRefresh Method

Enters a defer cycle that you can use to merge changes to the view and delay automatic refresh.

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> DeferRefresh() <b>As</b> System.IDisposable	
C#	
<b>public</b> System.IDisposable DeferRefresh()	

### Return Value

An **System.IDisposable** object that you can use to dispose of the calling object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2



## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### GetEnumerator Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : GetEnumerator Method

Returns an enumerator that iterates through the collection.

## Syntax

Visual Basic (Declaration)

```
Public Function GetEnumerator() As System.Collections.Generic.IEnumerator(Of ViewRow)
```

C#

```
public System.Collections.Generic.IEnumerator<ViewRow> GetEnumerator()
```

### Return Value

A **System.Collections.Generic.IEnumerator`1** that can be used to iterate through the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### GetItemProperties Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : GetItemProperties(PropertyDescriptor[]) Method

An array of **System.ComponentModel.PropertyDescriptor** objects to find in the collection as bindable. Can be null.

Returns the collection of properties available in the view element type to programmatic access through [ViewRow](#) and to data binding.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function GetItemProperties( _     ByVal listAccessors() As System.ComponentModel.PropertyDescriptor _ ) As System.ComponentModel.PropertyDescriptorCollection</pre>	
C#	
<pre>public System.ComponentModel.PropertyDescriptorCollection GetItemProperties(     System.ComponentModel.PropertyDescriptor[] listAccessors )</pre>	

### Parameters

*listAccessors*

An array of **System.ComponentModel.PropertyDescriptor** objects to find in the collection as bindable. Can be null.

## Remarks

If *listAccessors* is null, this method returns the collection of properties of the view element type.

Non-null *listAccessors* is used for obtaining property collection for hierarchical binding: If *listAccessors* contains a single element, then it is used to find an object-valued property in the element type, and the collection of properties of the type of that object-valued property is returned. If *listAccessors* contains two elements, then its second element is used to find an object-valued property in the collection of properties on the previous level, and so on, see **ITypedList.GetItemProperties** for more information.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

## IndexOf Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : [IndexOf\(ViewRow\) Method](#)

The view row to locate in the collection.

Determines the ordinal position of a specific view row in the [ViewRowCollection](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Function IndexOf( _     ByVal row As ViewRow _ ) As System.Integer</pre>	
C#	
<pre>public System.int IndexOf(     ViewRow row )</pre>	

## Parameters

*row*

The view row to locate in the collection.

## Return Value

The ordinal position of the view row in the collection if it is found; otherwise, -1.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

## Remove Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : [Remove Method](#)

The view row representing the item to delete.

Deletes the specified view item.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function Remove( _     ByVal row As ViewRow _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool Remove(     ViewRow row )</pre>	

### Parameters

*row*

The view row representing the item to delete.

### Return Value

**true**, if the item was deleted as a result of this operation; otherwise, **false**.

## Remarks

This is an update operation on the view equivalent to calling [ViewRow.Delete](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### RemoveAt Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : RemoveAt Method

The zero-based ordinal position of the item to remove.

Deletes the view row at a specified ordinal position in [ViewRowCollection](#).

# Syntax

Visual Basic (Declaration)	
<pre>Public Sub RemoveAt( _     ByVal ordinal As System.Integer _ )</pre>	
C#	
<pre>public void RemoveAt(     System.int ordinal )</pre>	

## Parameters

*ordinal*

The zero-based ordinal position of the item to remove.

## Remarks

This is an update operation on the view equivalent to calling [ViewRow.Delete](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

- [ViewRowCollection Class](#)
- [ViewRowCollection Members](#)












## Properties

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowCollection Class

For a list of all members of this type, see [ViewRowCollection members](#).

## Public Properties

	Name	Description
--	------	-------------

	<a href="#">AllowClear</a>	Gets a value indicating whether the <a href="#">Clear</a> operation is allowed on the view directly.
	<a href="#">AllowEdit</a>	Gets a value indicating whether modifying property values is allowed on the view directly.
	<a href="#">AllowNew</a>	Gets a value indicating whether the <a href="#">CreateRow</a> operation is allowed on the view directly.
	<a href="#">AllowRemove</a>	Gets a value indicating whether deleting rows is allowed on the view directly.
	<a href="#">Count</a>	Gets the number of elements in the view.
	<a href="#">GroupDescriptions</a>	Gets a collection of <b>System.ComponentModel.GroupDescription</b> objects that describe how the items in the collection are grouped.
	<a href="#">Groups</a>	Gets the top-level groups.
	<a href="#">Indexes</a>	Gets the collection of indexes for this view allowing to search for <a href="#">ViewRow</a> objects.
	<a href="#">Item</a>	Gets the view row at the specified ordinal position.
	<a href="#">Properties</a>	Returns the collection of properties available in the view element type to programmatic access through <a href="#">ViewRow</a> and to data binding.
	<a href="#">SortDescriptions</a>	Gets a collection of <b>System.ComponentModel.SortDescription</b> objects that describe how the items in the collection are sorted.

[Top](#)

## See Also

### Reference

[ViewRowCollection Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

## AllowClear Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : AllowClear Property

Gets a value indicating whether the [Clear](#) operation is allowed on the view directly.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> AllowClear <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool AllowClear { <b>get</b> ;}	

## Remarks

As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)). Note that the same operation is often allowed on the base data collection, and it will normally result in the corresponding change of the view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

## AllowEdit Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : AllowEdit Property

Gets a value indicating whether modifying property values is allowed on the view directly.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> AllowEdit <b>As</b> System.Boolean	

C#	
<code>public System.bool AllowEdit {get;}</code>	

## Remarks

As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)).

Although read-only properties of a view cannot be modified directly in the view, they still reflect up-to-date values of the source, so the difference is often not critical, you can always modify corresponding property in the source, that will automatically change the property in the view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### AllowNew Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : AllowNew Property

Gets a value indicating whether the [CreateRow](#) operation is allowed on the view directly.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property AllowNew As System.Boolean</code>	
C#	
<code>public System.bool AllowNew {get;}</code>	

## Remarks

As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)). Note that the same operation



is usually allowed on the base data collection, and it will normally result in the corresponding change of the view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)  
[ViewRowCollection Members](#)

### AllowRemove Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : AllowRemove Property

Gets a value indicating whether deleting rows is allowed on the view directly.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> AllowRemove <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool AllowRemove { <b>get</b> ;}	

## Remarks

As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)). Note that the same operation is often allowed on the base data collection, and it will normally result in the corresponding change of the view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### Count Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Count Property

Gets the number of elements in the view.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Count As System.Integer</code>	
C#	
<code>public System.int Count {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### GroupDescriptions Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : GroupDescriptions Property

Gets a collection of **System.ComponentModel.GroupDescription** objects that describe how the items in the collection are grouped.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property GroupDescriptions As System.Collections.ObjectModel.ObservableCollection(Of GroupDescription)</code>	
C#	

```
public System.Collections.ObjectModel.ObservableCollection<GroupDescription>  
GroupDescriptions {get;}
```

### Property Value

A collection of **System.ComponentModel.GroupDescription** objects that describe how the items in the collection are grouped.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### Groups Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Groups Property

Gets the top-level groups.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property Groups As  
System.Collections.ObjectModel.ReadOnlyObservableCollection(Of Object)
```

C#

```
public System.Collections.ObjectModel.ReadOnlyObservableCollection<object>  
Groups {get;}
```

### Property Value

A read-only collection of the top-level groups.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ViewRowCollection Class](#)  
[ViewRowCollection Members](#)

## Indexes Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Indexes Property

Gets the collection of indexes for this view allowing to search for [ViewRow](#) objects.

# Syntax

Visual Basic (Declaration)	
<code>Public MustOverride ReadOnly Property Indexes As IndexCollection(Of ViewRow)</code>	
C#	
<code>public abstract IndexCollection&lt;ViewRow&gt; Indexes {get;}</code>	

# Remarks

[ViewRowCollection](#) can be indexed, just like other LiveLinq data sources, to optimize searches for [ViewRow](#) objects. It implements the [C1.LiveLinq.Indexing.IIndexedSource<T>](#) interface.

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ViewRowCollection Class](#)  
[ViewRowCollection Members](#)  
[Indexes Property](#)

## Item Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Item Property

The zero-based ordinal position of the view row.

Gets the view row at the specified ordinal position.

## Syntax

Visual Basic (Declaration)	
<pre>Public ReadOnly Default Property Item( _     ByVal ordinal As System.Integer _ ) As ViewRow</pre>	
C#	
<pre>public ViewRow this[     System.int ordinal ]; {get;}</pre>	

### Parameters

*ordinal*

The zero-based ordinal position of the view row.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### Properties Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Properties Property

Returns the collection of properties available in the view element type to programmatic access through [ViewRow](#) and to data binding.

## Syntax

Visual Basic (Declaration)	
<pre>Public ReadOnly Property Properties As ViewRowPropertyInfo()</pre>	
C#	

```
public ViewRowPropertyInfo[] Properties {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### SortDescriptions Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : SortDescriptions Property

Gets a collection of **System.ComponentModel.SortDescription** objects that describe how the items in the collection are sorted.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property SortDescriptions As  
System.ComponentModel.SortDescriptionCollection
```

C#

```
public System.ComponentModel.SortDescriptionCollection SortDescriptions  
{get;}
```

### Property Value

A collection of **System.ComponentModel.SortDescription** objects that describe how the items in the collection are sorted.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



### Reference

## Events

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRowCollection Class](#)

For a list of all members of this type, see [ViewRowCollection members](#).

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after a view row has changed.
	<a href="#">ViewRowAdding</a>	Occurs after a new view row is created so it can be populated with default values.

[Top](#)

## See Also

### Reference

[ViewRowCollection Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

### Changed Event

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Changed Event

Occurs after a view row has changed.

## Syntax

Visual Basic (Declaration)	
<pre>Public Event Changed As System.EventHandler(Of SourceChangeEventArgs(Of ViewRow))</pre>	
C#	
<pre>public event System.EventHandler&lt;SourceChangeEventArgs&lt;ViewRow&gt;&gt; Changed</pre>	

## Event Data

The event handler receives an argument of type [SourceChangeEventArgs<T>](#) containing data related to this event. The following **SourceChangeEventArgs<T>** properties provide information specific to this event.

Property	Description
<a href="#">ChangeType</a>	Gets the type of change.
<a href="#">Item</a>	Gets the object that is being changed.
<a href="#">Ordinal</a>	Gets the ordinal position of the collection item that is being changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)  
[ViewRowCollection Members](#)

### ViewRowAdding Event

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : ViewRowAdding Event

Occurs after a new view row is created so it can be populated with default values.

## Syntax

Visual Basic (Declaration)	
<code>Public Event ViewRowAdding As System.EventHandler(Of ViewRowAddingEventArgs)</code>	
C#	
<code>public event System.EventHandler&lt;ViewRowAddingEventArgs&gt; ViewRowAdding</code>	

## Event Data

The event handler receives an argument of type [ViewRowAddingEventArgs](#) containing data related to this event. The following **ViewRowAddingEventArgs** properties provide information specific to this event.



Property	Description
<a href="#">Row</a>	Gets the new view row that has just been added to <a href="#">ViewRowCollection</a> .

## Remarks

This event occurs only if the new row is created directly in the view, as a result of a view update operation, that is, with [CreateRow](#) or via data binding. It does not occur when new rows appear in the view as a result of view maintenance on changes made to the view's source data collections.

This event occurs immediately on creating the new row, before the method creating it returns, before the row enters edit mode (see [ViewRowState](#) about edit mode).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

## ViewRowPropertyInfo

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowPropertyInfo Class

Allows to control certain behavior of a property of the element type of a [View](#).

## Object Model

ViewRowPropertyInfo

## Syntax

Visual Basic (Declaration)	
<b>Public Class</b> ViewRowPropertyInfo	
C#	
<b>public class</b> ViewRowPropertyInfo	

# Inheritance Hierarchy

System.Object  
C1.LiveLinq.LiveViews.ViewRowPropertyInfo

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowPropertyInfo Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Overview

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowPropertyInfo Class

Allows to control certain behavior of a property of the element type of a [View](#).

## Object Model

ViewRowPropertyInfo

## Syntax

Visual Basic (Declaration)	
Public Class ViewRowPropertyInfo	
C#	
public class ViewRowPropertyInfo	

## Inheritance Hierarchy

System.Object  
C1.LiveLinq.LiveViews.ViewRowPropertyInfo

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowPropertyInfo Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)



## Members

[Properties](#)

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRowPropertyInfo](#) Class

The following tables list the members exposed by [ViewRowPropertyInfo](#).

## Public Properties

	Name	Description
	<a href="#">ImmediateUpdate</a>	Gets or sets a boolean value indicating whether changes made to this property through data binding must be immediately sent to the corresponding view item even if the view is in editing mode.
	<a href="#">PropertyName</a>	Gets the name of the property.

[Top](#)


## See Also


### Reference

[ViewRowPropertyInfo Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Properties

>

Name	Description
 <a href="#">ImmediateUpdate</a>	Gets or sets a boolean value indicating whether changes made to this property through data binding must be immediately sent to the corresponding view item even if the view is in editing mode.

 **PropertyName** Gets the name of the property.

[Top](#)

## See Also

### Reference

[ViewRowPropertyInfo Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### ImmediateUpdate Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowPropertyInfo Class](#) : ImmediateUpdate Property

Gets or sets a boolean value indicating whether changes made to this property through data binding must be immediately sent to the corresponding view item even if the view is in editing mode.

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> ImmediateUpdate <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool ImmediateUpdate { <b>get</b> ; <b>set</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowPropertyInfo Class](#)

[ViewRowPropertyInfo Members](#)

[BeginEdit Method](#)

### PropertyName Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowPropertyInfo Class](#) : PropertyName Property

Gets the name of the property.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property PropertyName As System.String</code>	
C#	
<code>public System.string PropertyName {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowPropertyInfo Class](#)

[ViewRowPropertyInfo Members](#)

## Enumerations

### DataBindingMode

[C1.LiveLinq.LiveViews Namespace](#) : DataBindingMode Enumeration

Enumeration of the possible data binding modes. It is used by the [View.DataBindingMode](#) property.

## Syntax

Visual Basic (Declaration)	
<code>Public Enum DataBindingMode</code> <code>    Inherits System.Enum</code>	
C#	
<code>public enum DataBindingMode : System.Enum</code>	

## Members

Member	Description
<b>Default</b>	The default mode, which is <a href="#">WPF mode</a> in WPF applications and <a href="#">WinForms</a>

	<a href="#">mode</a> in WinForms applications.
<b>WinForms</b>	In data binding to a <a href="#">View</a> , view rows (elements of the <a href="#">View.Rows</a> collection) are used for data binding.
<b>WPF</b>	In data binding to a <a href="#">View</a> , view items (elements of the <a href="#">View</a> itself as a collection) are used for data binding.

## Inheritance Hierarchy

System.Object  
     System.ValueType  
         System.Enum  
             **C1.LiveLinq.LiveViews.DataBindingMode**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.LiveLinq.LiveViews Namespace](#)

## ViewMaintenanceMode

[C1.LiveLinq.LiveViews Namespace](#) : ViewMaintenanceMode Enumeration

Specifies how a view is synchronized with changes in its base data.

## Syntax

Visual Basic (Declaration)	
<pre><b>Public Enum</b> ViewMaintenanceMode     <b>Inherits</b> System.Enum</pre>	
C#	
<pre><b>public enum</b> ViewMaintenanceMode : System.Enum</pre>	

## Members

Member	Description
<b>Default</b>	<p>By default, the view is in a "smart mode": It is in <b>Deferred</b> mode if nobody is interested in its data, and it is synchronized and goes to <b>Immediate</b> mode if there is a client interested in receiving notifications of changes in that view's data.</p> <p>In other words, a view in <b>Default</b> mode is effectively in <b>Deferred</b> mode if no listeners registered to be notified of its changes, and it is effectively in <b>Immediate</b> mode if there are such listeners (for example, if there is GUI control bound to it).</p>
<b>Deferred</b>	<p>When a change in base data occur, the view is not synchronized with it immediately. It is allowed to go stale, out of sync with its base data as long as there are no requests for this view's data. The view is synchronized on demand, that is, it is synchronized when any request for its data arrives.</p>
<b>Immediate</b>	<p>The view is synchronized with its base data automatically and immediately after any change in its base data occurs. The view is kept synchronized with its base data at all times.</p>

## Remarks

**See Also:** View Maintenance Mode.

## Inheritance Hierarchy

System.Object

System.ValueType

System.Enum

**C1.LiveLinq.LiveViews.ViewMaintenanceMode**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.LiveLinq.LiveViews Namespace](#)  
[MaintenanceMode Property](#)  
[DeferredMaintenance Property](#)

## ViewOrder

[C1.LiveLinq.LiveViews Namespace](#) : ViewOrder Enumeration

Specifies whether and how a view must preserve item order if it exists in the source.

### Syntax

Visual Basic (Declaration)	
<pre>Public Enum ViewOrder     Inherits System.Enum</pre>	
C#	
<pre>public enum ViewOrder : System.Enum</pre>	

### Members

Member	Description
<b>NotPreserved</b>	Source order is not preserved. Preserving source order is not guaranteed even at view creation.
<b>PartiallyPreserved</b>	Source order is partially preserved. When the view is created, it preserves source order, but later, when changes occur in the source, view items added or modified to reflect those changes aren't guaranteed to appear at the same order position in the view as in the source.
<b>Preserved</b>	Source order is preserved completely. Order of items in the view is always the same as in the source (if source has a particular order), even after the view is maintained to reflect changes that occurred in the source.

### Inheritance Hierarchy

System.Object  
  System.ValueType  
    System.Enum  
      **C1.LiveLinq.LiveViews.ViewOrder**



## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.LiveLinq.LiveViews Namespace](#)

## ViewState

[C1.LiveLinq.LiveViews Namespace](#) : ViewState Enumeration

The state of a view row with regard to edit, add and delete operations if they are performed directly on the view.

## Syntax

Visual Basic (Declaration)	
<pre>Public Enum ViewState     Inherits System.Enum</pre>	
C#	
<pre>public enum ViewState : System.Enum</pre>	

## Members

Member	Description
<b>Detached</b>	The row was deleted, or it is a new row after exiting edit mode.
<b>Modified</b>	The row is in edit mode ( <a href="#">ViewRow.BeginEdit</a> was called, neither <a href="#">ViewRow.EndEdit</a> nor <a href="#">ViewRow.CancelEdit</a> was called yet), and the row is not new (was not created by <a href="#">ViewRowCollection.CreateRow</a> ).
<b>New</b>	The row was added to the view directly (not by adding to a basic data collection) by calling <a href="#">ViewRowCollection.CreateRow</a> or through data binding (such new row enters edit mode once it is created) and still in edit mode (neither <a href="#">ViewRow.EndEdit</a> nor <a href="#">ViewRow.CancelEdit</a> was called yet).

<b>Unmodified</b>	The row is a regular view row, not in edit mode and not deleted.
-------------------	--

## Remarks

This state concerns edit, add and delete operations performed directly on the view (programmatically via [ViewRow](#) objects or through data binding). It does not concern modifications made to the view's base (source) data collections. Modifications made to source data can also change view items, as a result of the normal view maintenance process, see [View.MaintenanceMode](#), but in that case the state of thus added or modified rows remains **Unmodified**.

### Note on adding rows directly to the view:

If a row is added directly to the view (as opposed to adding it to one of its base data collections), the following happens:

When a new row is created with [ViewRowCollection.CreateRow](#) or through data binding, it enters edit mode.

When it is committed with [ViewRow.EndEdit](#), a new row is added to the view's base data collection, and, usually, a corresponding row appears in the view, that has **Unmodified** state, although in some cases it may be more than one row or none, depending on the view query. The original view row no longer corresponds to a view item after the [ViewRow.EndEdit](#) or [ViewRow.CancelEdit](#) call, its state becomes **Detached**. Accessing it after that would throw an exception.

## Inheritance Hierarchy

```

System.Object
  System.ValueType
    System.Enum
      C1.LiveLinq.LiveViews.ViewRowState

```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



### Reference

[C1.LiveLinq.LiveViews Namespace](#)

# C1.LiveLinq.LiveViews.Xml Namespace

## Overview

### Classes

	Class	Description
	XHint	The class used for the return type in the <a href="#">XmlExtensions.IndexedElement</a> and <a href="#">XmlExtensions.IndexedAttribute</a> extension methods.
	XmlExtensions	Provides a set of static (extension) methods for LiveLinq to XML.

### See Also

#### Reference

[C1.LiveLinq.4 Assembly](#)

## Classes

### XHint

[C1.LiveLinq.LiveViews.Xml Namespace](#) : XHint Class

The class used for the return type in the [XmlExtensions.IndexedElement](#) and [XmlExtensions.IndexedAttribute](#) extension methods.

## Object Model

XHint

### Syntax

Visual Basic (Declaration)	
<pre>&lt;HintAttribute(C1.LiveLinq.LiveViews.Xml.XmlHintConverter)&gt; Public MustInherit Class XHint</pre>	
C#	
<pre>[Hint(C1.LiveLinq.LiveViews.Xml.XmlHintConverter)] public abstract class XHint</pre>	

## Remarks

This class is not used in execution, because hints like [XmlExtensions.IndexedElement](#) and [XmlExtensions.IndexedAttribute](#) are never executed, they are used merely to convey information ('hint') to LiveLinq query optimizer, they are discarded before the query is executed. The only purpose of this class is to preserve syntactical correctness of the query.

## Inheritance Hierarchy

System.Object  
    **C1.LiveLinq.LiveViews.Xml.XHint**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XHint Members](#)  
[C1.LiveLinq.LiveViews.Xml Namespace](#)

## Overview

[C1.LiveLinq.LiveViews.Xml Namespace](#) : XHint Class

The class used for the return type in the [XmlExtensions.IndexedElement](#) and [XmlExtensions.IndexedAttribute](#) extension methods.

## Object Model

XHint

## Syntax

Visual Basic (Declaration)	
<pre>&lt;HintAttribute(C1.LiveLinq.LiveViews.Xml.XmlHintConverter)&gt; Public MustInherit Class XHint</pre>	
C#	
<pre>[Hint(C1.LiveLinq.LiveViews.Xml.XmlHintConverter)]</pre>	

```
public abstract class XHint
```

## Remarks

This class is not used in execution, because hints like [XmlExtensions.IndexedElement](#) and [XmlExtensions.IndexedAttribute](#) are never executed, they are used merely to convey information ('hint') to LiveLinq query optimizer, they are discarded before the query is executed. The only purpose of this class is to preserve syntactical correctness of the query.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.Xml.XHint**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XHint Members](#)


[C1.LiveLinq.LiveViews.Xml Namespace](#)

## Members

[C1.LiveLinq.LiveViews.Xml Namespace](#) : XHint Class

The following tables list the members exposed by [XHint](#).

## Public Operators

 <a href="#">Explicit Type Conversion</a>	Overloaded.
--	-------------

[Top](#)

## See Also

### Reference

[XHint Class](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#)

## Operators

### Explicit Type Conversion Operator

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XHint Class](#) : Explicit Type Conversion Operator

### Overload List

Overload	Description
<a href="#">Explicit Type Conversion(String,XHint)</a>	
<a href="#">Explicit Type Conversion(Boolean,XHint)</a>	
<a href="#">Explicit Type Conversion(Int32,XHint)</a>	
<a href="#">Explicit Type Conversion(UInt32,XHint)</a>	
<a href="#">Explicit Type Conversion(Int64,XHint)</a>	
<a href="#">Explicit Type Conversion(UInt64,XHint)</a>	
<a href="#">Explicit Type Conversion(Single,XHint)</a>	
<a href="#">Explicit Type Conversion(Double,XHint)</a>	
<a href="#">Explicit Type Conversion(Decimal,XHint)</a>	
<a href="#">Explicit Type Conversion(DateTime,XHint)</a>	
<a href="#">Explicit Type Conversion(DateTimeOffset,XHint)</a>	
<a href="#">Explicit Type Conversion(TimeSpan,XHint)</a>	
<a href="#">Explicit Type Conversion(Guid,XHint)</a>	
<a href="#">Explicit Type Conversion(Nullable&lt;Boolean&gt;,XHint)</a>	
<a href="#">Explicit Type Conversion(Nullable&lt;Int32&gt;,XHint)</a>	

Explicit Type Conversion(Nullable<UInt32>,XHint)	
Explicit Type Conversion(Nullable<Int64>,XHint)	
Explicit Type Conversion(Nullable<UInt64>,XHint)	
Explicit Type Conversion(Nullable<Single>,XHint)	
Explicit Type Conversion(Nullable<Double>,XHint)	
Explicit Type Conversion(Nullable<Decimal>,XHint)	
Explicit Type Conversion(Nullable<DateTime>,XHint)	
Explicit Type Conversion(Nullable<DateTimeOffset>,XHint)	
Explicit Type Conversion(Nullable<TimeSpan>,XHint)	
Explicit Type Conversion(Nullable<Guid>,XHint)	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XHint Class](#)

[XHint Members](#)

## XmlExtensions

[C1.LiveLinq.LiveViews.Xml Namespace](#) : XmlExtensions Class

Provides a set of static (extension) methods for LiveLinq to XML.

## Object Model

XmlExtensions

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;&gt; &lt;HintAttribute(C1.LiveLinq.LiveViews.Xml.XmlHintConverter)&gt; Public MustInherit NotInheritable Class XmlExtensions</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] [Hint(C1.LiveLinq.LiveViews.Xml.XmlHintConverter)] public static class XmlExtensions</pre>	

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.Xml.XmlExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Members](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#)

## Overview

[C1.LiveLinq.LiveViews.Xml Namespace](#) : XmlExtensions Class

Provides a set of static (extension) methods for LiveLinq to XML.

## Object Model

XmlExtensions

## Syntax

Visual Basic (Declaration)	
----------------------------	--



```
<System.Runtime.CompilerServices.ExtensionAttribute(>>
<HintAttribute(C1.LiveLinq.LiveViews.Xml.XmlHintConverter)>
Public MustInherit NotInheritable Class XmlExtensions
```

C#

```
[System.Runtime.CompilerServices.Extension()]
[Hint(C1.LiveLinq.LiveViews.Xml.XmlHintConverter)]
public static class XmlExtensions
```

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.Xml.XmlExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Members](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#)





## Members












[Methods](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#) : XmlExtensions Class

The following tables list the members exposed by [XmlExtensions](#).

## Public Methods

	Name	Description
 	<a href="#">AsLive</a>	Overloaded. Creates a view based on the specified XML document.
 	<a href="#">Attributes</a>	Overloaded. Returns a view representing the collection of the attributes of every element in the source view.

≡  S	<a href="#">BeginUpdate</a>	Suspends notifications while massive changes are being made to an XML node and its descendants.
≡  S	<a href="#">DescendantNodes&lt;T&gt;</a>	Returns a view representing the collection of the descendent nodes of every element and document in the source view.
≡  S	<a href="#">DescendantNodesAndSelf</a>	Returns a view representing the collection of nodes that contains every element in the source view, and the descendent nodes of every element in the source view.
≡  S	<a href="#">Descendants</a>	Overloaded. Returns a view representing the collection of elements that contains the descendent elements of every element and document in the source view.
≡  S	<a href="#">DescendantsAndSelf</a>	Overloaded. Returns a view representing a collection of elements that contains every element in the source view, and the descendent elements of every element in the source view.
≡  S	<a href="#">Elements</a>	Overloaded. Returns a view representing the collection of child elements of every element and document in the source view..
≡  S	<a href="#">EndUpdate</a>	Ends notification suspension started with <a href="#">BeginUpdate</a> .
≡  S	<a href="#">IndexedAttribute</a>	Overloaded. A hint to create and use an index on the specified XML attribute. The hint has default action.
≡  S	<a href="#">IndexedElement</a>	Overloaded. A hint to create and use an index on the specified XML element. The hint has default action.
≡  S	<a href="#">Nodes&lt;T&gt;</a>	Returns a view representing the collection of child nodes of every document and element in the source view.
≡  S	<a href="#">Root</a>	Gets the view for the root element of the XML tree for this document.

[Top](#)

## See Also












### Reference

[XmlExtensions Class](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#)

### Methods

>

Name	Description
 <a href="#">S AsLive</a>	Overloaded. Creates a view based on the specified XML document.
 <a href="#">S Attributes</a>	Overloaded. Returns a view representing the collection of the attributes of every element in the source view.
 <a href="#">S BeginUpdate</a>	Suspends notifications while massive changes are being made to an XML node and its descendants.
 <a href="#">S DescendantNodes&lt;T&gt;</a>	Returns a view representing the collection of the descendent nodes of every element and document in the source view.
 <a href="#">S DescendantNodesAndSelf</a>	Returns a view representing the collection of nodes that contains every element in the source view, and the descendent nodes of every element in the source view.
 <a href="#">S Descendants</a>	Overloaded. Returns a view representing the collection of elements that contains the descendent elements of every element and document in the source view.
 <a href="#">S DescendantsAndSelf</a>	Overloaded. Returns a view representing a collection of elements that contains every element in the source view, and the descendent elements of every element in the source view.
 <a href="#">S Elements</a>	Overloaded. Returns a view representing the collection of child elements of every element and document in the source view..
 <a href="#">S EndUpdate</a>	Ends notification suspension started with <a href="#">BeginUpdate</a> .
 <a href="#">S IndexedAttribute</a>	Overloaded. A hint to create and use an index on the specified XML attribute. The hint has default action.
 <a href="#">S IndexedElement</a>	Overloaded. A hint to create and use an index on the specified

XML element. The hint has default action.

 [Nodes<T>](#)

Returns a view representing the collection of child nodes of every document and element in the source view.

 [Root](#)

Gets the view for the root element of the XML tree for this document.

[Top](#)

## See Also

### Reference

[XmlExtensions Class](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#)

### AsLive Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : AsLive Method

Creates a view based on the specified XML document.

## Overload List

Overload	Description
<a href="#">AsLive(XDocument)</a>	Creates a view based on the specified XML document.
<a href="#">AsLive(XElement)</a>	Creates a view based on the specified XML element.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

### AsLive(XDocument) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [AsLive Method](#) : AsLive(XDocument) Method

The XML document to expose as a view.

Creates a view based on the specified XML document.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function AsLive( _     ByVal document As System.Xml.Linq.XDocument _ ) As View(Of XDocument)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XDocument&gt; AsLive(     System.Xml.Linq.XDocument document )</pre>	

### Parameters

*document*

The XML document to expose as a view.

### Return Value

A view representing the specified XML document.

## Remarks

Since there can be only one root element in a document, the view based on a document (`document.AsLive()`) is similar to the view based on its root element (`document.Root.AsLive()`). The difference is that the document view contains the document as its only item, and the root view contains the root as its only item. This difference is essential only when the root of the document is replaced with a different element (and so the whole XML tree changes), then the document view remains valid and shows the changed document contents, whereas the root view becomes disconnected from the document.

**Note:** This view is owned by the **System.Xml.Linq.XDocument** object (it is stored as one of its annotations), so, if you create a view for the same document several times, it will be the same object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

#### AsLive(XElement) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [AsLive Method](#) : AsLive(XElement) Method

The XML element to expose as a view.

Creates a view based on the specified XML element.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function AsLive( _  
    ByVal element As System.Xml.Linq.XElement _  
) As View(Of XElement)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static View<XElement> AsLive(  
    System.Xml.Linq.XElement element  
)
```

### Parameters

*element*

The XML element to expose as a view.

### Return Value

A view representing the specified XML element.

## Remarks

This view represents a single XML node. Therefore, as a collection of items, this view's **Count** is always 1. This view is usually used as a starting point to construct a view containing elements or attributes from this node's descendants by using a query with operators from [XmlExtensions](#) such as **Elements**, **Descendants** and others, in combination with standard LINQ query operators **where**, **join** and others.

**Note:** This view is owned by the **System.Xml.Linq.XElement** object (it is stored as one of its annotations), so, if you create a view for the same element several times, it will be the same object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

### Attributes Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : Attributes Method

Returns a view representing the collection of the attributes of every element in the source view.

## Overload List

Overload	Description
<a href="#">Attributes(View&lt;XElement&gt;)</a>	Returns a view representing the collection of the attributes of every element in the source view.
<a href="#">Attributes(View&lt;XElement&gt;,XName)</a>	Returns a view representing a filtered collection of the attributes of every element in the source view. Only attributes that have a matching <b>System.Xml.Linq.XName</b> are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

Attributes(View<XElement>) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [Attributes Method](#) :

Attributes(View<XElement>) Method

The source view.

Returns a view representing the collection of the attributes of every element in the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function Attributes( _     ByVal view As View(Of XElement) _ ) As View(Of XAttribute)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XAttribute&gt; Attributes(     View&lt;XElement&gt; view )</pre>	

### Parameters

*view*

The source view.

### Return Value

A view containing the attributes of every element in the source view.

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

Attributes(View<XElement>,XName) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [Attributes Method](#) :

Attributes(View<XElement>,XName) Method

The source view.

The **System.Xml.Linq.XName** to match.

Returns a view representing a filtered collection of the attributes of every element in the source view. Only attributes that have a matching **System.Xml.Linq.XName** are included.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function Attributes( _     ByVal view As View(Of XElement), _     ByVal name As System.Xml.Linq.XName _ ) As View(Of XAttribute)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XAttribute&gt; Attributes(     View&lt;XElement&gt; view,     System.Xml.Linq.XName name )</pre>	

### Parameters

*view*

The source view.

*name*

The **System.Xml.Linq.XName** to match.

## Return Value

A view that contains a filtered collection of the attributes of every element in the source view. Only attributes that have a matching **System.Xml.Linq.XName** are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

## BeginUpdate Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : BeginUpdate Method

The node that is the root of a tree where massive changes are made in code.

Suspends notifications while massive changes are being made to an XML node and its descendants.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Sub BeginUpdate( _     ByVal node As System.Xml.Linq.XContainer _ )</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static void BeginUpdate(     System.Xml.Linq.XContainer node )</pre>	

## Parameters

*node*

The node that is the root of a tree where massive changes are made in code.

## Remarks

This method must be followed by [EndUpdate](#).

Use this method when you already have indexes over this XML or live views based on it, and you need to perform massive changes on the contents of this node and its descendants. Without this method, every single change you make causes LiveLinq to perform necessary operations for maintaining your indexes and live views dependent on this node and its descendants. In case of massive changes, this can be slower than to wait until the massive changes are done and rebuild the indexes and live views.

Between **BeginUpdate** and [EndUpdate](#) calls, indexes, live views, bound controls and other change notification listeners are not updated, they don't receive change notifications. When [EndUpdate](#) is called, a [SourceChangeType.Reset](#) notification is sent, meaning all indexes, live views and other collections dependent on this node and its descendants must be rebuilt from scratch.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

### DescendantNodes<T> Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : DescendantNodes<T> Method

The type of the objects in the source view, constrained to **System.Xml.Linq.XContainer**.

The source view.

Returns a view representing the collection of the descendent nodes of every element and document in the source view.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Shared Function DescendantNodes(Of T As System.Xml.Linq.XContainer)( _
    ByVal view As View(Of T) _
) As View(Of XNode)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<XNode> DescendantNodes<T>(
    View<T> view
)
where T: System.Xml.Linq.XContainer
```

## Parameters

*view*

The source view.

## Type Parameters

*T*

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

## Return Value

A view containing every descendent node of every document and element in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

## DescendantNodesAndSelf Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : DescendantNodesAndSelf Method

The source view.

Returns a view representing the collection of nodes that contains every element in the source view, and the descendent nodes of every element in the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Function DescendantNodesAndSelf( _     ByVal view As View(Of XElement) _ ) As View(Of XNode)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XNode&gt; DescendantNodesAndSelf(     View&lt;XElement&gt; view )</pre>	

### Parameters

*view*

The source view.

### Return Value

A view containing every element in the source view, and the descendent nodes of every element in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

### Descendants Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : Descendants Method

Returns a view representing the collection of elements that contains the descendent elements of every element and document in the source view.

## Overload List

Overload	Description
<a href="#">Descendants&lt;T&gt;(View&lt;T&gt;)</a>	Returns a view representing the collection of elements that contains the descendent elements of every element and document in the source view.
<a href="#">Descendants&lt;T&gt;(View&lt;T&gt;,XName)</a>	Returns a view representing a filtered collection of elements that contains the descendent elements of every element and document in the source view. Only elements that have a matching <b>System.Xml.Linq.XName</b> are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

[Descendants<T>\(View<T>\) Method](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [Descendants Method](#) :

[Descendants<T>\(View<T>\) Method](#)

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

The source view.

Returns a view representing the collection of elements that contains the descendent elements of every element and document in the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; <a href="#">Public Overloads Shared Function</a> Descendants(<a href="#">Of T As</a></pre>	

```
System.Xml.Linq.XContainer)( _
    ByVal view As View(Of T) _
) As View(Of XElement)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<XElement> Descendants<T>(
    View<T> view
)
where T: System.Xml.Linq.XContainer
```

## Parameters

*view*

The source view.

## Type Parameters

*T*

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

## Return Value

A view containing every descendent element of every element and document in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

[Overload List](#)

Descendants<T>(View<T>,XName) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [Descendants Method](#) :

Descendants<T>(View<T>,XName) Method

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

The source view.

The **System.Xml.Linq.XName** to match.

Returns a view representing a filtered collection of elements that contains the descendent elements of every element and document in the source view. Only elements that have a matching **System.Xml.Linq.XName** are included.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function Descendants(Of T As System.Xml.Linq.XContainer)( _     ByVal view As View(Of T), _     ByVal name As System.Xml.Linq.XName _ ) As View(Of XElement)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XElement&gt; Descendants&lt;T&gt;(     View&lt;T&gt; view,     System.Xml.Linq.XName name ) where T: System.Xml.Linq.XContainer</pre>	

### Parameters

*view*

The source view.

*name*

The **System.Xml.Linq.XName** to match.

### Type Parameters

*T*

The type of the objects in the source view, constrained to **System.Xml.Linq.XContainer**.

### Return Value

A view containing descendants of elements and documents in the source view. Only elements that have a matching **System.Xml.Linq.XName** are included.



## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

### DescendantsAndSelf Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : DescendantsAndSelf Method

Returns a view representing a collection of elements that contains every element in the source view, and the descendent elements of every element in the source view.

## Overload List

Overload	Description
<a href="#">DescendantsAndSelf(View&lt;XElement&gt;)</a>	Returns a view representing a collection of elements that contains every element in the source view, and the descendent elements of every element in the source view.
<a href="#">DescendantsAndSelf(View&lt;XElement&gt;,XName)</a>	Returns a view representing a filtered collection of elements that contains every element in the source view, and the descendants of every element in the source view. Only elements that have a matching <b>System.Xml.Linq.XName</b> are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

DescendantsAndSelf(View<XElement>) Method

[C1.LiveInq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [DescendantsAndSelf Method](#) :

DescendantsAndSelf(View<XElement>) Method

The source view.

Returns a view representing a collection of elements that contains every element in the source view, and the descendent elements of every element in the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function DescendantsAndSelf( _     ByVal view As View(Of XElement) _ ) As View(Of XElement)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XElement&gt; DescendantsAndSelf(     View&lt;XElement&gt; view )</pre>	

### Parameters

*view*

The source view.

### Return Value

A view containing every element in the source view, and the descendent elements of every element in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

[Overload List](#)

DescendantsAndSelf(View<XElement>,XName) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [DescendantsAndSelf Method](#) :

DescendantsAndSelf(View<XElement>,XName) Method

The source view.

The **System.Xml.Linq.XName** to match.

Returns a view representing a filtered collection of elements that contains every element in the source view, and the descendants of every element in the source view. Only elements that have a matching **System.Xml.Linq.XName** are included.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function DescendantsAndSelf( _     ByVal view As View(Of XElement), _     ByVal name As System.Xml.Linq.XName _ ) As View(Of XElement)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XElement&gt; DescendantsAndSelf(     View&lt;XElement&gt; view,     System.Xml.Linq.XName name )</pre>	

### Parameters

*view*

The source view.

*name*

The **System.Xml.Linq.XName** to match.

### Return Value

A view containing elements in the source view, and the descendants of elements in the source view. Only elements that have a matching **System.Xml.Linq.XName** are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

### Elements Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : Elements Method

Returns a view representing the collection of child elements of every element and document in the source view..

## Overload List

Overload	Description
<a href="#">Elements&lt;T&gt;(View&lt;T&gt;)</a>	Returns a view representing the collection of child elements of every element and document in the source view..
<a href="#">Elements&lt;T&gt;(View&lt;T&gt;,XName)</a>	Returns a view representing filtered collection of child elements of every element and document in the source view. Only elements that have a matching <b>System.Xml.Linq.XName</b> are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

Elements<T>(View<T>) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [Elements Method](#) :

Elements<T>(View<T>) Method

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

The source view.

Returns a view representing the collection of child elements of every element and document in the source view..

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function Elements(Of T As System.Xml.Linq.XContainer)( _     ByVal view As View(Of T) _ ) As View(Of XElement)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XElement&gt; Elements&lt;T&gt;(     View&lt;T&gt; view ) where T: System.Xml.Linq.XContainer</pre>	

### Parameters

*view*

The source view.

### Type Parameters

*T*

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

### Return Value

A view containing the child elements of every element and document in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

Elements<T>(View<T>,XName) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [Elements Method](#) :

Elements<T>(View<T>,XName) Method

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

The source view.

The **System.Xml.Linq.XName** to match.

Returns a view representing filtered collection of child elements of every element and document in the source view. Only elements that have a matching **System.Xml.Linq.XName** are included.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function Elements(Of T As  
System.Xml.Linq.XContainer)( _  
    ByVal view As View(Of T), _  
    ByVal name As System.Xml.Linq.XName _  
) As View(Of XElement)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static View<XElement> Elements<T>(  
    View<T> view,  
    System.Xml.Linq.XName name  
)
```

where T: System.Xml.Linq.XContainer

## Parameters

*view*

The source view.

*name*

The **System.Xml.Linq.XName** to match.

## Type Parameters

*T*

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

## Return Value

A view that contains a filtered collection of child elements of every element and document in the source view. Only elements that have a matching **System.Xml.Linq.XName** are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

## EndUpdate Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : EndUpdate Method

The node that is the root of a tree where massive changes have been made since the [BeginUpdate](#) call.

Ends notification suspension started with [BeginUpdate](#).

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Sub EndUpdate( _     ByVal node As System.Xml.Linq.XContainer _ )</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static void EndUpdate(     System.Xml.Linq.XContainer node )</pre>	

## Parameters

*node*

The node that is the root of a tree where massive changes have been made since the [BeginUpdate](#) call.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)

## IndexedAttribute Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : IndexedAttribute Method

A hint to create and use an index on the specified XML attribute. The hint has default action.

## Overload List

Overload	Description
<a href="#">IndexedAttribute(XElement,XName)</a>	A hint to create and use an index on the specified XML attribute. The hint



	has default action.
<a href="#">IndexedAttribute(XAttribute)</a>	A hint to create and use an index on the specified XML attribute. The hint has default action.
<a href="#">IndexedAttribute(XAttribute,IndexingHintAction)</a>	A hint to create and use an index on the specified XML attribute. The hint has specified action.
<a href="#">IndexedAttribute(XElement,XName,IndexingHintAction)</a>	A hint to create and use an index on the specified XML attribute. The hint has specified action.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

[IndexedAttribute\(XElement,XName\) Method](#)

[Example](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [IndexedAttribute Method](#) :

[IndexedAttribute\(XElement,XName\) Method](#)

The element containing the attribute.

The name of the attribute to get.

A hint to create and use an index on the specified XML attribute. The hint has default action.

## Syntax

Visual Basic (Declaration)	
<System.Runtime.CompilerServices.ExtensionAttribute(>	

```
Public Overloads Shared Function IndexedAttribute( _
    ByVal element As System.Xml.Linq.XElement, _
    ByVal name As System.Xml.Linq.XName _
) As XHint
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static XHint IndexedAttribute(
    System.Xml.Linq.XElement element,
    System.Xml.Linq.XName name
)
```

## Parameters

*element*

The element containing the attribute.

*name*

The name of the attribute to get.

## Return Value

The attribute that has the specified name.

## Remarks

Hints are used declaratively. They tell LiveLINQ query optimizer to create and use an index on this attribute, if possible. When the query is executed, the hint method **IndexedAttribute** is replaced with the standard LINQ to XML method **System.Xml.Linq.XElement.Attribute(System.Xml.Linq.XName)**. See [C1.LiveLINQ.Hints](#) for more details.

## Example

- C#

```
var query =
    from c in customers
    where (string)c.IndexedAttribute("CustomerID") == "ALFKI"
    select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

[IndexedAttribute\(XAttribute\) Method](#)

[Example](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [IndexedAttribute Method](#) :

[IndexedAttribute\(XAttribute\) Method](#)

The attribute to apply the hint to.

A hint to create and use an index on the specified XML attribute. The hint has default action.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function IndexedAttribute( _     ByVal attribute As System.Xml.Linq.XAttribute _ ) As XHint</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static XHint IndexedAttribute(     System.Xml.Linq.XAttribute attribute )</pre>	

### Parameters

*attribute*

The attribute to apply the hint to.

### Return Value

Formally, the hint returns the same value that it receives in the parameter. In fact, it is never executed, its role is purely declarative.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on this attribute, if possible. After it is used for query optimization, before the query is executed, this hint is removed from the expression. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```
var query =
    from c in customers
    where (string)c.Attribute("CustomerID").IndexedAttribute() ==
"ALFKI"
    select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

IndexedAttribute(XAttribute,IndexingHintAction) Method

[Example](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [IndexedAttribute Method](#) :  
IndexedAttribute(XAttribute,IndexingHintAction) Method

The attribute to apply the hint to.

The action specified by the hint.

A hint to create and use an index on the specified XML attribute. The hint has specified action.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function IndexedAttribute( _     ByVal attribute As System.Xml.Linq.XAttribute, _     ByVal action As IndexingHintAction _ ) As XHint</pre>	

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static XHint IndexedAttribute(  
    System.Xml.Linq.XAttribute attribute,  
    IndexingHintAction action  
)
```

## Parameters

*attribute*

The attribute to apply the hint to.

*action*

The action specified by the hint.

## Return Value

Formally, the hint returns the same value that it receives in the parameter. In fact, it is never executed, its role is purely declarative.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on this attribute, if possible. After it is used for query optimization, before the query is executed, this hint is removed from the expression. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```
var query =  
    from c in customers  
    where  
        (string)c.Attribute("CustomerID").IndexedAttribute(IndexingHintAction.Mandatory) == "ALFKI"  
    select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

## IndexedAttribute(XElement,XName,IndexingHintAction) Method

### [Example](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [IndexedAttribute Method](#) :

IndexedAttribute(XElement,XName,IndexingHintAction) Method

The element containing the attribute.

The name of the attribute to get.

The action specified by the hint.

A hint to create and use an index on the specified XML attribute. The hint has specified action.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function IndexedAttribute( _     ByVal element As System.Xml.Linq.XElement, _     ByVal name As System.Xml.Linq.XName, _     ByVal action As IndexingHintAction _ ) As XHint</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static XHint IndexedAttribute(     System.Xml.Linq.XElement element,     System.Xml.Linq.XName name,     IndexingHintAction action )</pre>	

## Parameters

*element*

The element containing the attribute.

*name*

The name of the attribute to get.

*action*

The action specified by the hint.

## Return Value

The attribute that has the specified name.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on this attribute, if possible. When the query is executed, the hint method **IndexedAttribute** is replaced with the standard LINQ to XML method **System.Xml.Linq.XElement.Attribute(System.Xml.Linq.XName)**. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```
var query =  
    from c in customers  
    where (string)c.IndexedAttribute("CustomerID",  
IndexingHintAction.Mandatory) == "ALFKI"  
    select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

## IndexedElement Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : IndexedElement Method

A hint to create and use an index on the specified XML element. The hint has default action.

## Overload List

Overload	Description
<a href="#">IndexedElement(XContainer,XName)</a>	A hint to create and use an index on

	the specified XML element. The hint has default action.
<a href="#">IndexedElement(XElement)</a>	A hint to create and use an index on the specified XML element. The hint has default action.
<a href="#">IndexedElement(XElement,IndexingHintAction)</a>	A hint to create and use an index on the specified XML element. The hint has specified action.
<a href="#">IndexedElement(XContainer,XName,IndexingHintAction)</a>	A hint to create and use an index on the specified XML element. The hint has specified action.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

[IndexedElement\(XContainer,XName\) Method](#)

[Example](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [IndexedElement Method](#) :

[IndexedElement\(XContainer,XName\) Method](#)

The element or document containing the element.

The name of the element to get.

A hint to create and use an index on the specified XML element. The hint has default action.

## Syntax

Visual Basic (Declaration)



```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function IndexedElement( _
    ByVal container As System.Xml.Linq.XContainer, _
    ByVal name As System.Xml.Linq.XName _
) As XHint
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static XHint IndexedElement(
    System.Xml.Linq.XContainer container,
    System.Xml.Linq.XName name
)
```

## Parameters

*container*

The element or document containing the element.

*name*

The name of the element to get.

## Return Value

The element that has the specified name.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on this element, if possible. When the query is executed, the hint method **IndexedElement** is replaced with the standard LINQ to XML method **System.Xml.Linq.XContainer.Element(System.Xml.Linq.XName)**. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```
var query =
    from c in customers
    where (string)c.IndexedElement("CustomerID") == "ALFKI"
    select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

IndexedElement(XElement) Method

[Example](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [IndexedElement Method](#) :

IndexedElement(XElement) Method

The element to apply the hint to.

A hint to create and use an index on the specified XML element. The hint has default action.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function IndexedElement( _  
    ByVal element As System.Xml.Linq.XElement _  
) As XHint
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static XHint IndexedElement(  
    System.Xml.Linq.XElement element  
)
```

### Parameters

*element*

The element to apply the hint to.

### Return Value

Formally, the hint returns the same value that it receives in the parameter. In fact, it is never executed, its role is purely declarative.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on this element, if possible. After it is used for query optimization, before the query is executed, this hint is removed from the expression. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```
var query =  
    from c in customers  
    where (string)c.Element("CustomerID").IndexedElement() == "ALFKI"  
    select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

[Overload List](#)

IndexedElement(XElement,IndexingHintAction) Method

[Example](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [IndexedElement Method](#) :

IndexedElement(XElement,IndexingHintAction) Method

The element to apply the hint to.

The action specified by the hint.

A hint to create and use an index on the specified XML element. The hint has specified action.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function IndexedElement( _  
    ByVal element As System.Xml.Linq.XElement, _
```

```

    ByVal action As IndexingHintAction _
) As XHint

```

C#

```

[System.Runtime.CompilerServices.Extension()]
public static XHint IndexedElement(
    System.Xml.Linq.XElement element,
    IndexingHintAction action
)

```

## Parameters

*element*

The element to apply the hint to.

*action*

The action specified by the hint.

## Return Value

Formally, the hint returns the same value that it receives in the parameter. In fact, it is never executed, its role is purely declarative.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on this element, if possible. After it is used for query optimization, before the query is executed, this hint is removed from the expression. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```

var query =
    from c in customers
    where
        (string)c.Element("CustomerID").IndexedElement(IndexingHintAction.Mandatory) == "ALFKI"
    select c;

```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

[IndexedElement\(XContainer,XName,IndexingHintAction\) Method](#)

[Example](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [IndexedElement Method](#) :

[IndexedElement\(XContainer,XName,IndexingHintAction\) Method](#)

The element or document containing the element.

The name of the element to get.

The action specified by the hint.

A hint to create and use an index on the specified XML element. The hint has specified action.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function IndexedElement( _     ByVal container As System.Xml.Linq.XContainer, _     ByVal name As System.Xml.Linq.XName, _     ByVal action As IndexingHintAction _ ) As XHint</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static XHint IndexedElement(     System.Xml.Linq.XContainer container,     System.Xml.Linq.XName name,     IndexingHintAction action )</pre>	

### Parameters

*container*

The element or document containing the element.

*name*

The name of the element to get.

*action*

The action specified by the hint.

## Return Value

The element that has the specified name.

## Remarks

Hints are used declaratively. They tell LiveLinq query optimizer to create and use an index on this element, if possible. When the query is executed, the hint method **IndexedElement** is replaced with the standard LINQ to XML method **System.Xml.Linq.XContainer.Element(System.Xml.Linq.XName)**. See [C1.LiveLinq.Hints](#) for more details.

## Example

- [C#](#)

```
var query =  
    from c in customers  
    where (string)c.IndexedElement("CustomerID",  
        IndexingHintAction.Mandatory) == "ALFKI"  
    select c;
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

## Nodes<T> Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : Nodes<T> Method

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

The source *view*.

Returns a view representing the collection of child nodes of every document and element in the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Function Nodes(Of T As System.Xml.Linq.XContainer)( _     ByVal view As View(Of T) _ ) As View(Of XElement)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XElement&gt; Nodes&lt;T&gt;(     View&lt;T&gt; view ) where T: System.Xml.Linq.XContainer</pre>	

### Parameters

*view*

The source view.

### Type Parameters

*T*

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

### Return Value

A view containing every child node of every document and element in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

## Root Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : Root Method

The view representing this XML document.

Gets the view for the root element of the XML tree for this document.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Function Root( _     ByVal documentView As View(Of XDocument) _ ) As View(Of XElement)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XElement&gt; Root(     View&lt;XDocument&gt; documentView )</pre>	

## Parameters

*documentView*

The view representing this XML document.

## Return Value

A view containing a single element, the root of the XML tree.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference



# C1.LiveLinq.Metadata Namespace

## Overview

[Inheritance Hierarchy](#)

## See Also


### Reference

[C1.LiveLinq.4 Assembly](#)

# C1.WPF.LiveLinq Namespace

## Overview

## Classes

	Class	Description
	<a href="#">WpfExtensions</a>	Provides a set of static (extension) methods for <b>System.Collections.Specialized.INotifyCollectionChanged</b> and <b>System.Collections.ObjectModel.ObservableCollection`1</b> types (specific to WPF).

## See Also

### Reference

[C1.LiveLinq.4 Assembly](#)

## Classes

## WpfExtensions

[C1.WPF.LiveLinq Namespace](#) : WpfExtensions Class

Provides a set of static (extension) methods for **System.Collections.Specialized.INotifyCollectionChanged** and **System.Collections.ObjectModel.ObservableCollection`1** types (specific to WPF).

## Object Model

WpfExtensions

# Syntax

Visual Basic (Declaration)	
<code>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public MustInherit NotInheritable Class WpfExtensions</code>	
C#	
<code>[System.Runtime.CompilerServices.Extension() public static class WpfExtensions</code>	

# Inheritance Hierarchy

System.Object  
    **C1.WPF.LiveLinq.WpfExtensions**

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[WpfExtensions Members](#)  
[C1.WPF.LiveLinq Namespace](#)

## Overview

[C1.WPF.LiveLinq Namespace](#) : WpfExtensions Class

Provides a set of static (extension) methods for **System.Collections.Specialized.INotifyCollectionChanged** and **System.Collections.ObjectModel.ObservableCollection`1** types (specific to WPF).

# Object Model

WpfExtensions

# Syntax

Visual Basic (Declaration)	
----------------------------	--

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public MustInherit NotInheritable Class WpfExtensions
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static class WpfExtensions
```

## Inheritance Hierarchy

System.Object

**C1.WPF.LiveLinq.WpfExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[WpfExtensions Members](#)

[C1.WPF.LiveLinq Namespace](#)



## Members

[Methods](#)

[C1.WPF.LiveLinq Namespace](#) : WpfExtensions Class

The following tables list the members exposed by [WpfExtensions](#).

## Public Methods

	Name	Description
	<a href="#">AsLive</a>	Overloaded. Creates a view based on the specified <b>System.Collections.Specialized.INotifyCollectionChanged</b> data source.
	<a href="#">ToIndexed</a>	Overloaded. Creates an <a href="#">C1.LiveLinq.Collections.IndexedCollection&lt;T&gt;</a> based on the specified <b>System.Collections.Specialized.INotifyCollectionChanged</b> data

		source.
--	--	---------

[Top](#)

## See Also

### Reference

[WpfExtensions Class](#)



[C1.WPF.LiveLinq Namespace](#)

## Methods

[C1.WPF.LiveLinq Namespace](#) : [WpfExtensions Class](#)

For a list of all members of this type, see [WpfExtensions members](#).

## Public Methods

	Name	Description
	<a href="#">AsLive</a>	Overloaded. Creates a view based on the specified <b>System.Collections.Specialized.INotifyCollectionChanged</b> data source.
	<a href="#">ToIndexed</a>	Overloaded. Creates an <a href="#">C1.LiveLinq.Collections.IndexedCollection&lt;T&gt;</a> based on the specified <b>System.Collections.Specialized.INotifyCollectionChanged</b> data source.

[Top](#)

## See Also

### Reference

[WpfExtensions Class](#)

[C1.WPF.LiveLinq Namespace](#)

### AsLive Method

[C1.WPF.LiveLinq Namespace](#) > [WpfExtensions Class](#) : [AsLive Method](#)

Creates a view based on the specified

**System.Collections.Specialized.INotifyCollectionChanged** data source.

## Overload List

Overload	Description
<a href="#">AsLive&lt;T&gt;(INotifyCollectionChanged)</a>	Creates a view based on the specified <b>System.Collections.Specialized.INotifyCollectionChanged</b> data source.
<a href="#">AsLive&lt;T&gt;(INotifyCollectionChanged,ViewOrder)</a>	Creates a view based on the specified <b>System.Collections.Specialized.INotifyCollectionChanged</b> data source.
<a href="#">AsLive&lt;T&gt;(ObservableCollection&lt;T&gt;)</a>	A typed specialization of the <a href="#">AsLive&lt;T&gt;(INotifyCollectionChanged)</a> method.
<a href="#">AsLive&lt;T&gt;(ObservableCollection&lt;T&gt;,ViewOrder)</a>	A typed specialization of the <a href="#">AsLive&lt;T&gt;(INotifyCollectionChanged,ViewOrder)</a> method.
<a href="#">AsLive&lt;T&gt;(ReadOnlyObservableCollection&lt;T&gt;)</a>	A typed specialization of the <a href="#">AsLive&lt;T&gt;(INotifyCollectionChanged)</a> method.
<a href="#">AsLive&lt;T&gt;(ReadOnlyObservableCollection&lt;T&gt;,ViewOrder)</a>	A typed specialization of the <a href="#">AsLive&lt;T&gt;(INotifyCollectionChanged,ViewOrder)</a> method.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[WpfExtensions Class](#)

[WpfExtensions Members](#)

[AsLive<T>\(INotifyCollectionChanged\) Method](#)

[C1.WPF.LiveLinq Namespace](#) > [WpfExtensions Class](#) > [AsLive Method](#) :

[AsLive<T>\(INotifyCollectionChanged\) Method](#)

The type of the elements in the view.

The **System.Collections.Specialized.INotifyCollectionChanged** data source to expose as a view.

Creates a view based on the specified

**System.Collections.Specialized.INotifyCollectionChanged** data source.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function AsLive(Of T)( _
    ByVal source As System.Collections.Specialized.INotifyCollectionChanged _
) As View(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<T> AsLive<T>(
    System.Collections.Specialized.INotifyCollectionChanged source
)
```

### Parameters

*source*

The **System.Collections.Specialized.INotifyCollectionChanged** data source to expose as a view.

### Type Parameters

*T*

The type of the elements in the view.

### Return Value

A view that contains the same elements as the **System.Collections.Specialized.INotifyCollectionChanged** data source.

## Remarks

Use this method to build views from existing data sources implementing **System.Collections.Specialized.INotifyCollectionChanged**. The element type of this data source must implement

**System.ComponentModel.INotifyPropertyChanged**, see Using the built-in collection class `IndexedCollection(T)` (LiveLinq to

Objects)|tag=Using\_the\_built\_in\_collection\_class\_IndexedCollectionT\_LiveLinq\_to\_Objects.

The resulting view may have its elements ordered differently than they are ordered in the *source*. Correspondingly, views built on this resulting view (for example, if you filter it with **Where**) will not preserve the source order either. If you need to preserve the source order, consider using the other **AsLive** overload where you can specify to what extent you need the order to be preserved.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[WpfExtensions Class](#)  
[WpfExtensions Members](#)  
[Overload List](#)

**AsLive<T>(INotifyCollectionChanged,ViewOrder) Method**

[C1.WPF.LiveLinq Namespace](#) > [WpfExtensions Class](#) > [AsLive Method](#) :

**AsLive<T>(INotifyCollectionChanged,ViewOrder) Method**

The type of the elements in the view.

The **System.Collections.Specialized.INotifyCollectionChanged** data source to expose as a view.

Specifies whether to preserve source item order.

Creates a view based on the specified

**System.Collections.Specialized.INotifyCollectionChanged** data source.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function AsLive(Of T)( _
    ByVal source As System.Collections.Specialized.INotifyCollectionChanged, _
    ByVal order As ViewOrder _
) As View(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static View<T> AsLive<T>(  
    System.Collections.Specialized.INotifyCollectionChanged source,  
    ViewOrder order  
)
```

## Parameters

*source*

The **System.Collections.Specialized.INotifyCollectionChanged** data source to expose as a view.

*order*

Specifies whether to preserve source item order.

## Type Parameters

*T*

The type of the elements in the view.

## Return Value

A view that contains the same elements as the **System.Collections.Specialized.INotifyCollectionChanged** data source.

## Remarks

Use this method to build views from existing data sources implementing **System.Collections.Specialized.INotifyCollectionChanged**. The element type of this data source must implement **System.ComponentModel.INotifyPropertyChanged**, see Using the built-in collection class `IndexedCollection(T)` (LiveLinq to Objects)|tag=Using\_the\_built\_in\_collection\_class\_IndexedCollectionT\_LiveLinq\_to\_Objects.

If the *order* parameter specifies preserving item order, the order of items in the source is preserved, at a certain performance cost, in the resulting view and in views based on it (for example, if you filter it with **Where**).

Note that **Join** does not preserve source order. If you need to order a join result, use **OrderBy** after **Join**.

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[WpfExtensions Class](#)  
[WpfExtensions Members](#)  
[Overload List](#)

AsLive<T>(ObservableCollection<T>) Method

[C1.WPF.LiveLinq Namespace](#) > [WpfExtensions Class](#) > [AsLive Method](#) :

AsLive<T>(ObservableCollection<T>) Method

The type of the elements in the view.

The **System.Collections.ObjectModel.ObservableCollection`1** to expose as a view.

A typed specialization of the [AsLive<T>\(INotifyCollectionChanged\)](#) method.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function AsLive(Of T)( _     ByVal source As System.Collections.ObjectModel.ObservableCollection(Of T) - ) As View(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsLive&lt;T&gt;(     System.Collections.ObjectModel.ObservableCollection&lt;T&gt; source )</pre>	

### Parameters

*source*

The **System.Collections.ObjectModel.ObservableCollection`1** to expose as a view.

### Type Parameters

*T*

The type of the elements in the view.

## Return Value

A view that contains the same elements as the **System.Collections.ObjectModel.ObservableCollection`1**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[WpfExtensions Class](#)  
[WpfExtensions Members](#)  
[Overload List](#)

AsLive<T>(ObservableCollection<T>,ViewOrder) Method

[C1.WPF.LiveLinq Namespace](#) > [WpfExtensions Class](#) > [AsLive Method](#) :

AsLive<T>(ObservableCollection<T>,ViewOrder) Method

The type of the elements in the view.

The **System.Collections.ObjectModel.ObservableCollection`1** to expose as a view.

Specifies whether to preserve source item order.

A typed specialization of the [AsLive<T>\(INotifyCollectionChanged,ViewOrder\)](#) method.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute()&gt; Public Overloads Shared Function AsLive(Of T)( _     ByVal source As System.Collections.ObjectModel.ObservableCollection(Of T),     _     ByVal order As ViewOrder _ ) As View(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsLive&lt;T&gt;(</pre>	

```
System.Collections.ObjectModel.ObservableCollection<T> source,  
    ViewOrder order  
)
```

## Parameters

*source*

The **System.Collections.ObjectModel.ObservableCollection`1** to expose as a view.

*order*

Specifies whether to preserve source item order.

## Type Parameters

*T*

The type of the elements in the view.

## Return Value

A view that contains the same elements as the  
**System.Collections.ObjectModel.ObservableCollection`1**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[WpfExtensions Class](#)  
[WpfExtensions Members](#)  
[Overload List](#)

AsLive<T>(ReadOnlyObservableCollection<T>) Method

[C1.WPF.LiveLinq Namespace](#) > [WpfExtensions Class](#) > [AsLive Method](#) :

AsLive<T>(ReadOnlyObservableCollection<T>) Method

The type of the elements in the view.

The **System.Collections.ObjectModel.ReadOnlyObservableCollection`1** to expose as a view.

A typed specialization of the [AsLive<T>\(INotifyCollectionChanged\)](#) method.

## Syntax

#### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function AsLive(Of T)( _  
    ByVal source As  
System.Collections.ObjectModel.ReadOnlyObservableCollection(Of T) _  
) As View(Of T)
```

#### C#

```
[System.Runtime.CompilerServices.Extension()]  
public static View<T> AsLive<T>(  
    System.Collections.ObjectModel.ReadOnlyObservableCollection<T> source  
)
```

### Parameters

*source*

The **System.Collections.ObjectModel.ReadOnlyObservableCollection`1** to expose as a view.

### Type Parameters

*T*

The type of the elements in the view.

### Return Value

A view that contains the same elements as the **System.Collections.ObjectModel.ReadOnlyObservableCollection`1**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[WpfExtensions Class](#)  
[WpfExtensions Members](#)  
[Overload List](#)

AsLive<T>(ReadOnlyObservableCollection<T>,ViewOrder) Method

[C1.WPF.LiveLinq Namespace](#) > [WpfExtensions Class](#) > [AsLive Method](#) :

AsLive<T>(ReadOnlyObservableCollection<T>,ViewOrder) Method

The type of the elements in the view.

The **System.Collections.ObjectModel.ReadOnlyObservableCollection`1** to expose as a view.

Specifies whether to preserve source item order.

A typed specialization of the [AsLive<T>\(INotifyCollectionChanged,ViewOrder\)](#) method.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function AsLive(Of T)( _
    ByVal source As
System.Collections.ObjectModel.ReadOnlyObservableCollection(Of T), _
    ByVal order As ViewOrder _
) As View(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<T> AsLive<T>(
    System.Collections.ObjectModel.ReadOnlyObservableCollection<T> source,
    ViewOrder order
)
```

## Parameters

*source*

The **System.Collections.ObjectModel.ReadOnlyObservableCollection`1** to expose as a view.

*order*

Specifies whether to preserve source item order.

## Type Parameters

*T*

The type of the elements in the view.

## Return Value

A view that contains the same elements as the **System.Collections.ObjectModel.ReadOnlyObservableCollection`1**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[WpfExtensions Class](#)  
[WpfExtensions Members](#)  
[Overload List](#)

### ToIndexed Method

[C1.WPF.LiveLinq Namespace](#) > [WpfExtensions Class](#) : ToIndexed Method

Creates an [C1.LiveLinq.Collections.IndexedCollection<T>](#) based on the specified **System.Collections.Specialized.INotifyCollectionChanged** data source.

## Overload List

Overload	Description
<a href="#">ToIndexed&lt;T&gt;(INotifyCollectionChanged)</a>	Creates an <a href="#">C1.LiveLinq.Collections.IndexedCollection&lt;T&gt;</a> based on the specified <b>System.Collections.Specialized.INotifyCollectionChanged</b> data source.
<a href="#">ToIndexed&lt;T&gt;(ObservableCollection&lt;T&gt;)</a>	A typed specialization of the <a href="#">ToIndexed&lt;T&gt;(INotifyCollectionChanged)</a> method.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[WpfExtensions Class](#)

[WpfExtensions Members](#)

ToIndexed<T>(INotifyCollectionChanged) Method

[C1.WPF.LiveLinq Namespace](#) > [WpfExtensions Class](#) > [ToIndexed Method](#) :

ToIndexed<T>(INotifyCollectionChanged) Method

The type of the elements in the collection.

An **System.Collections.Specialized.INotifyCollectionChanged** data source to represent as an [C1.LiveLinq.Collections.IndexedCollection<T>](#).

Creates an [C1.LiveLinq.Collections.IndexedCollection<T>](#) based on the specified **System.Collections.Specialized.INotifyCollectionChanged** data source.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function ToIndexed(Of T)( _     ByVal source As System.Collections.Specialized.INotifyCollectionChanged _ ) As IndexedCollection(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static IndexedCollection&lt;T&gt; ToIndexed&lt;T&gt;(     System.Collections.Specialized.INotifyCollectionChanged source )</pre>	

## Parameters

*source*

An **System.Collections.Specialized.INotifyCollectionChanged** data source to represent as an [C1.LiveLinq.Collections.IndexedCollection<T>](#).

## Type Parameters

*T*

The type of the elements in the collection.

## Return Value

An [C1.LiveLinq.Collections.IndexedCollection<T>](#) that contains the same elements as the **System.Collections.Specialized.INotifyCollectionChanged** and enables indexing of that data source.

## Remarks

Use this method to index and query your existing data sources implementing **System.Collections.Specialized.INotifyCollectionChanged**. The element type of this data source must implement **System.ComponentModel.INotifyPropertyChanged**, see Using the built-in collection class `IndexedCollection(T)` (LiveLinq to Objects)|tag=Using\_the\_built\_in\_collection\_class\_IndexedCollectionT\_LiveLinq\_to\_Objects.

The collection returned by this method also implements the **System.Collections.Specialized.INotifyCollectionChanged** interface, because this method actually returns objects of an internal derived class that implements that interface.

**Note:** Indexes created on the resulting [C1.LiveLinq.Collections.IndexedCollection<T>](#) are owned by it and not by the original data source. Every **ToIndexed()** call creates a separate object that has its own separate indexes. Avoid calling **ToIndexed()** repeatedly for the same collection because it can increase the cost of maintaining indexes.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[WpfExtensions Class](#)  
[WpfExtensions Members](#)  
[Overload List](#)

ToIndexed<T>(ObservableCollection<T>) Method

[C1.WPF.LiveLinq Namespace](#) > [WpfExtensions Class](#) > [ToIndexed Method](#) :

ToIndexed<T>(ObservableCollection<T>) Method

The type of the elements in the collection.

An **System.Collections.ObjectModel.ObservableCollection`1** represent as an [C1.LiveLinq.Collections.IndexedCollection<T>](#).



A typed specialization of the [ToIndexed<T>\(INotifyCollectionChanged\)](#) method.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function ToIndexed(Of T)( _     ByVal source As System.Collections.ObjectModel.ObservableCollection(Of T) - ) As IndexedCollection(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static IndexedCollection&lt;T&gt; ToIndexed&lt;T&gt;(     System.Collections.ObjectModel.ObservableCollection&lt;T&gt; source )</pre>	

### Parameters

*source*

An **System.Collections.ObjectModel.ObservableCollection`1** represent as an [C1.LiveLinq.Collections.IndexedCollection<T>](#).

### Type Parameters

*T*

The type of the elements in the collection.

### Return Value

An [C1.LiveLinq.Collections.IndexedCollection<T>](#) that contains the same elements as the **System.Collections.ObjectModel.ObservableCollection`1** and enables indexing of that collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

# C1.Silverlight.Data.Entity.5 Assembly


## Namespaces

### C1.Data Namespace

#### Overview

#### Classes

Class	Description
 <a href="#">ClientCacheBase</a>	Studio for Entity Framework におけるデータアクセスの中心的な役割を果たすクライアント側のキャッシュを表します。
 <a href="#">ClientScope</a>	
 <a href="#">ClientView&lt;T&gt;</a>	1つのクライアントビューを表します。これは、 <b>System.ServiceModel.DomainServices.Client.DomainContext</b> などのリモートソースに接続された <a href="#">ライブビュー</a> です。
 <a href="#">ClientViewLoadedEventArgs</a>	<a href="#">ClientView&lt;T&gt;.Loaded</a> イベントのデータを提供します。
 <a href="#">DataExtensions</a>	
 <a href="#">FilteredView&lt;T&gt;</a>	
 <a href="#">PagingView&lt;T&gt;</a>	ページブレーク付きの <a href="#">クライアントビュー</a> を表します。
 <a href="#">ProgressiveView&lt;T&gt;</a>	

 <a href="#">SavedChangesEventArgs</a>	C1DataSource.SavedChanges イベントのデータを提供します。
---	---

## See Also

### Reference

[C1.Silverlight.Data.Entity.5 Assembly](#)

## Classes

### ClientCacheBase

[C1.Data Namespace](#) : ClientCacheBase Class

Studio for Entity Framework  
 におけるデータアクセスの中心的な役割を果たすクライアント側のキャッシュを表します。

## Object Model

**ClientCacheBase**

## Syntax

Visual Basic (Declaration)	
<code>Public MustInherit Class ClientCacheBase</code>	
C#	
<code>public abstract class ClientCacheBase</code>	

## Remarks

通常は、アプリケーションの起動時に、このクラスの 1 つのインスタンスが `ObjectContext/DomainContext` をパラメータとして作成され、アプリケーションの全ライフタイムにわたって存在します。  
 一方、各フォーム、ウィンドウ、ユーザーコントロールは、[CreateScope](#) メソッドを呼び出すことで作成される [ClientScope](#) を使用してデータを操作します。

以下のようなプラットフォーム固有の実装の基本クラスです。  
 C1.Data.Entities.EntityClientCache (Entity Framework)  
 C1.Silverlight.Data.RiaServices.RiaClientCache (RIA サービス)

## Inheritance Hierarchy

System.Object  
**C1.Data.ClientCacheBase**  
[C1.Data.Entities.EntityClientCache](#)  
[C1.Silverlight.Data.RiaServices.RiaClientCache](#)

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[ClientCacheBase Members](#)  
[C1.Data Namespace](#)

Overview

[C1.Data Namespace](#) : ClientCacheBase Class

Studio for Entity Framework  
におけるデータアクセスの中心的な役割を果たすクライアント側のキャッシュを表します。

Object Model

ClientCacheBase

Syntax

Visual Basic (Declaration)	
<b>Public MustInherit Class</b> ClientCacheBase	
C#	
<b>public abstract class</b> ClientCacheBase	

Remarks

通常は、アプリケーションの起動時に、このクラスの 1 つのインスタンスが  
ObjectContext/DomainContext をパラメータとして作成され、  
アプリケーションの全ライフタイムにわたって存在します。  
一方、各フォーム、ウィンドウ、ユーザーコントロールは、[CreateScope](#)  
メソッドを呼び出すことで作成される [ClientScope](#) を使用してデータを操作します。

以下のようなプラットフォーム固有の実装の基本クラスです。

- C1.Data.Entities.EntityClientCache (Entity Framework)
- C1.Silverlight.Data.RiaServices.RiaClientCache (RIA サービス)

## Inheritance Hierarchy

System.Object

- C1.Data.ClientCacheBase**
  - C1.Data.Entities.EntityClientCache
  - C1.Silverlight.Data.RiaServices.RiaClientCache

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Members](#)  
[C1.Data Namespace](#)

## Members

[Methods](#)

[C1.Data Namespace](#) : ClientCacheBase Class

The following tables list the members exposed by [ClientCacheBase](#).

## Public Methods

	Name	Description
⇒	<a href="#">BulkChanges</a>	エンティティに対する大規模な変更をグループ化するため、およびエンティティの状態を手作業で明示的に変更できるようにするために使用されます。
⇒	<a href="#">CleanupCache</a>	未使用のメモリを強制的に解放し、未使用のエンティティをコンテキストから強制的にデタッチします。 通常は自動的に行われるため、プログラマがコード内でこのメソッドを呼び出す必要はほとんどありません。
⇒	<a href="#">Clear</a>	クライアント側のキャッシュを完全にクリアします。現時点から後のクエリーが

		サーバーから必ず新しいデータを取得するようにしたい場合に、このメソッドを呼び出してください。
⇒ <a href="#">CreateScope</a>		データアクセスの範囲を定義する <a href="#">ClientScope</a> を作成します。
⇒ <a href="#">CreateTransaction</a>		トランザクションスコープ内で行われた変更を簡単にキャンセルできる <a href="#">C1.Data.Transactions.ClientTransaction</a> を作成します。
⇒ <a href="#">Refresh</a>		この <a href="#">ClientCacheBase</a> に接続されているすべての <a href="#">C1DataSource</a> コントロールのデータを更新します。
⇒ <a href="#">RejectChanges</a>		この <a href="#">ClientCacheBase</a> に対する保留中のすべての変更を元に戻します。 <b>System.ServiceModel.DomainServices.Client.DomainContext.RejectChanges</b> の代わりにこのメソッドを呼び出すことをお勧めします。
⇒ <a href="#">SaveChanges</a>		すべての変更をサーバーに永続化します。 <b>System.Data.Objects.ObjectContext.SaveChanges()</b> の代わりにこのメソッドを呼び出すことをお勧めします。

[Top](#)

## See Also

### Reference

[ClientCacheBase Class](#)

[C1.Data Namespace](#)

## Methods

[C1.Data Namespace](#) : [ClientCacheBase Class](#)

For a list of all members of this type, see [ClientCacheBase members](#).

## Public Methods

Name	Description
⇒ <a href="#">BulkChanges</a>	エンティティに対する大規模な変更をグループ化するため、およびエンティティの状態を手作業で明示的に変更できるようにするために使用されます。

⇒ <a href="#">Cleanup Cache</a>	未使用のメモリを強制的に解放し、未使用のエンティティをコンテキストから強制的にデタッチします。 通常は自動的に行われるため、プログラマがコード内でこのメソッドを呼び出す必要はほとんどありません。
⇒ <a href="#">Clear</a>	クライアント側のキャッシュを完全にクリアします。現時点から後のクエリーがサーバーから必ず新しいデータを取得するようにしたい場合に、このメソッドを呼び出してください。
⇒ <a href="#">CreateScope</a>	データアクセスの範囲を定義する <a href="#">ClientScope</a> を作成します。
⇒ <a href="#">CreateTransaction</a>	トランザクションスコープ内で行われた変更を簡単にキャンセルできる <a href="#">C1.Data.Transactions.ClientTransaction</a> を作成します。
⇒ <a href="#">Refresh</a>	この <a href="#">ClientCacheBase</a> に接続されているすべての <a href="#">C1DataSource</a> コントロールのデータを更新します。
⇒ <a href="#">RejectChanges</a>	この <a href="#">ClientCacheBase</a> に対する保留中のすべての変更を元に戻します。 <b>System.ServiceModel.DomainServices.Client.DomainContext.RejectChanges</b> の代わりにこのメソッドを呼び出すことをお勧めします。
⇒ <a href="#">SaveChanges</a>	すべての変更をサーバーに永続化します。 <b>System.Data.Objects.ObjectContext.SaveChanges()</b> の代わりにこのメソッドを呼び出すことをお勧めします。

[Top](#)

## See Also

### Reference

[ClientCacheBase Class](#)

[C1.Data Namespace](#)

### BulkChanges Method

[Example](#)

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : BulkChanges Method

エンティティで変更を行うデリゲート。

エンティティに対する大規模な変更をグループ化するため、およびエンティティの状態を手作業で明示的に変更できるようにするために使用されます。

## Syntax

Visual Basic (Declaration)	
<pre>Public Sub BulkChanges( _     ByVal makeChanges As System.Action _ )</pre>	
C#	
<pre>public void BulkChanges(     System.Action makeChanges )</pre>	

### Parameters

*makeChanges*

エンティティで変更を行うデリゲート。

## Remarks

クライアント側のキャッシュの内部状態およびキャッシュに基づくすべての既存のクライアントビューは、そのまま維持され、指定された

*makeChanges*の実行中は更新されません。

デリゲートが実行（複数のエンティティの変更）を完了すると、

クライアント側キャッシュの内部状態が復元され、クライアントビューは、

デリゲートの実行中にエンティティに加えられた変更を反映するように更新（維持）されます。

このメソッドの呼び出しを検討する必要があるシナリオは、主に次の2つです。

1. 複数のエンティティに対して多くの変更を行う場合にこのメソッドを使用すると、パフォーマンスを向上させることができます。  
これは、変更処理が後で実行されるようにし、変更が行われるたびに処理が行われるのではなく、すべての変更が行われた後で1回だけ行われるためです。  
変更の量によっては、処理速度がかなり向上する可能性があります。
2. 次のいずれかのメソッドを呼び出してエンティティの状態を変更する必要がある場合は、このメソッドを使用しなければなりません。
  - System.Data.Objects.ObjectStateEntry.ChangeState/SetModified/AcceptChanges
  - System.Data.Objects.ObjectContext.AcceptAllChanges
  - System.ServiceModel.DomainServices.Client.DomainContext.RejectChanges
  - System.ServiceModel.DomainServices.Client.Entity.AcceptChanges/RejectChanges



- System.ServiceModel.DomainServices.Client.EntitySet.AcceptChanges/RejectChanges
- System.Windows.Controls.DomainDataSource.RejectChanges

これらのメソッドを [BulkChanges](#)

でラップせずに呼び出すと、クライアント側のキャッシュが破損する可能性があります

。

## Example

- [C#](#)

```
var scope = clientCache.CreateScope();
clientCache.BulkChanges(delegate {
    foreach(var detail in scope.GetItems<Order_Details>())
        detail.Discount *= 2;
});
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

### CleanupCache Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : CleanupCache Method

未使用のメモリを強制的に解放し、未使用のエンティティをコンテキストから強制的にデタッチします。

通常は自動的に行われるため、プログラマがコード内でこのメソッドを呼び出す必要はほとんどありません。

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> CleanupCache()	
C#	
<b>public void</b> CleanupCache()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

### Clear Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : Clear Method

クライアント側のキャッシュを完全にクリアします。現時点から後のクエリーがサーバーから必ず新しいデータを取得するようにしたい場合に、このメソッドを呼び出してください。

## Syntax

Visual Basic (Declaration)	
<code>Public Sub Clear()</code>	
C#	
<code>public void Clear()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

### CreateScope Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : CreateScope Method

データアクセスの範囲を定義する [ClientScope](#) を作成します。

## Syntax

Visual Basic (Declaration)	
<code>Public Function CreateScope() As ClientScope</code>	
C#	
<code>public ClientScope CreateScope()</code>	

## Return Value

新しい [ClientScope](#)。

## Remarks

通常、すべてのフォーム、ウィンドウ、ユーザーコントロールは、[ClientScope](#) を作成し、それを使用して エンティティにアクセスします。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

[ClientScope Class](#)

### CreateTransaction Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : CreateTransaction Method

トランザクションスコープ内で行われた変更を簡単にキャンセルできる [C1.Data.Transactions.ClientTransaction](#) を作成します。

## Syntax

Visual Basic (Declaration)	
<code>Public Function CreateTransaction() As ClientTransaction</code>	
C#	
<code>public ClientTransaction CreateTransaction()</code>	

## Return Value

新しい [C1.Data.Transactions.ClientTransaction](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

### Refresh Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : Refresh Method

この ClientCacheBase に接続されているすべての C1DataSource コントロールのデータを更新します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Sub Refresh()</pre>	
C#	
<pre>public void Refresh()</pre>	

## Remarks

このメソッドは、この ClientCacheBase に接続されている C1DataSource ごとに C1DataSource.Refresh() を呼び出します。クライアント上で行われた変更は維持されます。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

## RejectChanges Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : RejectChanges Method

この [ClientCacheBase](#) に対する保留中のすべての変更を元に戻します。

**System.ServiceModel.DomainServices.Client.DomainContext.RejectChanges**

の代わりにこのメソッドを呼び出すことをお勧めします。

## Syntax

Visual Basic (Declaration)

```
Public Sub RejectChanges()
```

C#

```
public void RejectChanges()
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

## SaveChanges Method

[C1.Data Namespace](#) > [ClientCacheBase Class](#) : SaveChanges Method

すべての変更をサーバーに永続化します。**System.Data.Objects.ObjectContext.SaveChanges()** の代わりにこのメソッドを呼び出すことをお勧めします。

## Syntax

Visual Basic (Declaration)

```
Public Sub SaveChanges()
```

C#

```
public void SaveChanges()
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCacheBase Class](#)

[ClientCacheBase Members](#)

## ClientScope

[C1.Data Namespace](#) : ClientScope Class

## Object Model

ClientScope

## Syntax

Visual Basic (Declaration)	
<b>Public Class</b> ClientScope	
C#	
<b>public class</b> ClientScope	

## Inheritance Hierarchy

System.Object

**C1.Data.ClientScope**

[C1.Data.Entities.EntityClientScope](#)

[C1.Silverlight.Data.RiaServices.RiaClientScope](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

Reference

[ClientScope Members](#)  
[C1.Data Namespace](#)

Overview

[C1.Data Namespace](#) : ClientScope Class

Object Model

ClientScope

Syntax

Visual Basic (Declaration)	
<code>Public Class ClientScope</code>	
C#	
<code>public class ClientScope</code>	

Inheritance Hierarchy

System.Object  
    **C1.Data.ClientScope**  
        [C1.Data.Entities.EntityClientScope](#)  
        [C1.Silverlight.Data.RiaServices.RiaClientScope](#)

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[ClientScope Members](#)  
[C1.Data Namespace](#)


Members

[Properties](#) [Methods](#)

[C1.Data Namespace](#) : ClientScope Class

The following tables list the members exposed by [ClientScope](#).

## Public Constructors

	Name	Description
	<a href="#">ClientScope Constructor</a>	




[Top](#)

## Public Properties

	Name	Description
	<a href="#">ClientCache</a>	

[Top](#)

## Public Methods

	Name	Description
	<a href="#">AddRef</a>	Overloaded.
	<a href="#">Dispose</a>	
	<a href="#">Release</a>	Overloaded.

[Top](#)

## See Also

### Reference

[ClientScope Class](#)  
[C1.Data Namespace](#)

## ClientScope Constructor

[C1.Data Namespace](#) > [ClientScope Class](#) : ClientScope Constructor

## Syntax

Visual Basic (Declaration)



```
Public Function New( _
    ByVal clientCache As ClientCacheBase _
)

C#

public ClientScope(
    ClientCacheBase clientCache
)
```

Parameters

*clientCache*

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference




- [ClientScope Class](#)
- [ClientScope Members](#)

Methods

[C1.Data Namespace](#) : ClientScope Class

For a list of all members of this type, see [ClientScope members](#).

Public Methods

	Name	Description
	<a href="#">AddRef</a>	Overloaded.
	<a href="#">Dispose</a>	
	<a href="#">Release</a>	Overloaded.

[Top](#)

See Also

Reference

[ClientScope Class](#)  
[C1.Data Namespace](#)

AddRef Method

[C1.Data Namespace](#) > [ClientScope Class](#) : AddRef Method

Overload List

Overload	Description
<a href="#">AddRef(Object)</a>	
<a href="#">AddRef(Type)</a>	

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[ClientScope Class](#)  
[ClientScope Members](#)

AddRef(Object) Method

[C1.Data Namespace](#) > [ClientScope Class](#) > [AddRef Method](#) : AddRef(Object) Method

Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Sub AddRef( _     ByVal entity As System.Object _ )</pre>	
C#	
<pre>public void AddRef(     System.object entity )</pre>	

Parameters

*entity*

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

- [ClientScope Class](#)
- [ClientScope Members](#)
- [Overload List](#)

AddRef(Type) Method  
[C1.Data Namespace](#) > [ClientScope Class](#) > [AddRef Method](#) : AddRef(Type) Method

Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Sub AddRef( _     ByVal entityType As System.Type _ )</pre>	
C#	
<pre>public void AddRef(     System.Type entityType )</pre>	

Parameters

*entityType*

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

## Reference

[ClientScope Class](#)  
[ClientScope Members](#)  
[Overload List](#)

## Dispose Method

[C1.Data Namespace](#) > [ClientScope Class](#) : Dispose Method

## Syntax

Visual Basic (Declaration)	
<code>Public Sub Dispose()</code>	
C#	
<code>public void Dispose()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientScope Class](#)  
[ClientScope Members](#)

## Release Method

[C1.Data Namespace](#) > [ClientScope Class](#) : Release Method

## Overload List

Overload	Description
<a href="#">Release(Object)</a>	
<a href="#">Release(Type)</a>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientScope Class](#)  
[ClientScope Members](#)

[Release\(Object\) Method](#)

[C1.Data Namespace](#) > [ClientScope Class](#) > [Release Method](#) : Release(Object) Method

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Release( _     ByVal entity As System.Object _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool Release(     System.object entity )</pre>	

### Parameters

*entity*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientScope Class](#)  
[ClientScope Members](#)  
[Overload List](#)

Release(Type) Method

[C1.Data Namespace](#) > [ClientScope Class](#) > [Release Method](#) : Release(Type) Method

Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Release( _     ByVal entityType As System.Type _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool Release(     System.Type entityType )</pre>	

Parameters

*entityType*

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

- [ClientScope Class](#)
- [ClientScope Members](#)
- [Overload List](#)

Properties

[C1.Data Namespace](#) : [ClientScope Class](#)

For a list of all members of this type, see [ClientScope members](#).

Public Properties

	Name	Description
	<a href="#">ClientCache</a>	

[Top](#)

## See Also

### Reference

[ClientScope Class](#)

[C1.Data Namespace](#)

### ClientCache Property

[C1.Data Namespace](#) > [ClientScope Class](#) : ClientCache Property

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property ClientCache As ClientCacheBase
```

C#

```
public ClientCacheBase ClientCache {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientScope Class](#)

[ClientScope Members](#)

## ClientView<T>

[C1.Data Namespace](#) : ClientView<T> Class

ビュー内の要素の型。

1つのクライアントビューを表します。これは、**System.ServiceModel.DomainServices.Client.DomainContext**などのリモートソースに接続された [ライブビュー](#)です。

## Object Model

ClientView<T>

## Syntax

Visual Basic (Declaration)	
<pre>Public Class ClientView(Of T)     Inherits C1.LiveLinq.LiveViews.View(Of T)     Implements C1.LiveLinq.IObservableSource(Of T)</pre>	
C#	
<pre>public class ClientView&lt;T&gt; : C1.LiveLinq.LiveViews.View&lt;T&gt;,     C1.LiveLinq.IObservableSource&lt;T&gt;</pre>	

## Type Parameters

*T*

ビュー内の要素の型。

## Inheritance Hierarchy

```
System.Object
  C1.LiveLinq.LiveViews.View
    C1.LiveLinq.LiveViews.View<T>
      C1.Data.ClientView<T>
        C1.Data.FilteredView<T>
        C1.Data.PagingView<T>
        C1.Data.ProgressiveView<T>
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Members](#)  
[C1.Data Namespace](#)

### Overview

[C1.Data Namespace](#) : ClientView<T> Class

ビュー内の要素の型。

1つのクライアントビューを表します。これは、**System.ServiceModel.DomainServices.Client.DomainContext** などのリモートソースに接続された **ライブビュー** です。



# Object Model

ClientView<T>

## Syntax

Visual Basic (Declaration)

```
Public Class ClientView(Of T)  
    Inherits C1.LiveLinq.LiveViews.View(Of T)  
    Implements C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class ClientView<T> : C1.LiveLinq.LiveViews.View<T>,  
C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

ビュー内の要素の型。

## Inheritance Hierarchy

System.Object

C1.LiveLinq.LiveViews.View

C1.LiveLinq.LiveViews.View<T>

**C1.Data.ClientView<T>**

C1.Data.FilteredView<T>

C1.Data.PagingView<T>

C1.Data.ProgressiveView<T>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Members](#)

[C1.Data Namespace](#)











## Members




[Properties](#) [Methods](#) [Events](#)

[C1.Data Namespace](#) : [ClientView<T>](#) Class

The following tables list the members exposed by [ClientView<T>](#).






### Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	データへのアクセスがあった場合に <a href="#">クライアントビュー</a> を自動的にロードするかどうかを示す boolean 値を取得または設定します。
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">IsLoaded</a>	<a href="#">クライアントビュー</a> がロードされたかどうかを示す値を取得します。
	<a href="#">IsLoading</a>	<a href="#">クライアントビュー</a> がロード中であるかどうかを示す値を取得します。
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is

		True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Scope</a>	このクライアントビューが属する <a href="#">クライアントスコープ</a> を取得します。
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## Public Methods

Name	Description
 <a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">C1.LiveLinq.LiveViews.View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">AsFiltered</a>	<a href="#">predicate</a> を使用してサーバー側でビューをフィルタ処理します。
 <a href="#">AsFilteredBound&lt;TKey&gt;</a>	キーセレクタ関数および設定可能な <b>P:C1.Data.FilteredView`1.FilterKey</b> と <a href="#">演算子</a> を使用して サーバー上のビューをフィルタ処理します。
 <a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )



⇒ <a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">CancelLoad</a>	現在のロード操作をキャンセルします。
⇒ <a href="#">Concat</a>	Overloaded. Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">GroupJoin</a>	Overloaded. Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">Join</a>	Overloaded. Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">Load</a>	クライアントビューのエンティティをロードします。
⇒ <a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <a href="#">OrderBy&lt;T&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from

Key>	C1.LiveLinq.LiveViews.View<T>)
⇒OrderByDescending<TKey>	Sorts the elements of a view in descending order. (Inherited from C1.LiveLinq.LiveViews.View<T>)
⇒Paging	Overloaded. このクライアントビューにページングを適用します。
⇒ProgressiveLoading	Overloaded. クライアントビューのロードをサーバーへの1回のトリップで行うのではなく、複数のバッチで行い、それぞれのロードにおけるページサイズを制限することで、すべてのページのロードが完了しないうちにユーザーが結果を確認して操作できるようにするように指定します。
⇒PurgeEmptyGroups	Remove empty groups from a grouping view. (Inherited from C1.LiveLinq.LiveViews.View)
⇒Rebuild	Re-populates the view by re-executing the view's query. (Inherited from C1.LiveLinq.LiveViews.View)
⇒Refresh	クライアント側のキャッシュを無視して、クライアントビューのエンティティをロードします。
⇒Select<TResult>	Projects each element of a view into a new form. (Inherited from C1.LiveLinq.LiveViews.View<T>)
⇒SelectMany	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from C1.LiveLinq.LiveViews.View<T>)
⇒SetTransaction	Sets the value of the Transaction property. (Inherited from C1.LiveLinq.LiveViews.View)
⇒ToString	Returns a string representing this view. (Inherited from C1.LiveLinq.LiveViews.View<T>)

⇒ <a href="#">Union</a>	Overloaded. Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">Loaded</a>	<a href="#">クライアントビューのロード</a> が正常に終了したときに発生します。 また、 <a href="#">ロード</a> 中に例外が生成された場合にも発生します。

[Top](#)

## See Also

### Reference

[ClientView<T> Class](#)



[C1.Data Namespace](#)

## Methods

[C1.Data Namespace](#) : [ClientView<T> Class](#)

For a list of all members of this type, see [ClientView<T> members](#).

## Public Methods

	Name	Description
⇒ 	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ 	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">C1.LiveLinq.LiveViews.View</a> to a View<dynamic> so it can be used for data

	binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <a href="#">AsFiltered</a>	<a href="#">predicate</a> を使用してサーバー側でビューをフィルタ処理します。
⇒ <a href="#">AsFilteredBound&lt;TKey&gt;</a>	キーセレクタ関数および設定可能な <b>P:C1.Data.FilteredView`1.FilterKey</b> と <a href="#">演算子</a> を使用して サーバー上のビューをフィルタ処理します。
⇒ <a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">CancelLoaded</a>	現在の <a href="#">ロード操作</a> をキャンセルします。
⇒ <a href="#">Concat</a>	Overloaded. Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">GroupJoin</a>	Overloaded. Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from

		<a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;)</a>
⇒Join		Overloaded. Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;)</a>
⇒Load		クライアントビューのエンティティをロードします。
⇒Maintain		Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒OrderBy<TKey>		Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;)</a>
⇒OrderByDescending<TKey>		Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;)</a>
⇒Paging		Overloaded. このクライアントビューにページングを適用します。
⇒ProgressiveLoading		Overloaded. クライアントビューのロードをサーバーへの 1 回のトリップで行うのではなく、複数のバッチで行い、それぞれのロードにおけるページサイズを制限することで、すべてのページのロードが完了しないうちにユーザーが結果を確認して操作できるようにするように指定します。
⇒PurgeEmptyGroups		Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒Rebuild		Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒Refresh		クライアント側のキャッシュを無視して、クライアントビューのエンティティをロードします。
⇒Select<TResult>		Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;)</a>
⇒SelectMany		Overloaded. Projects each element of this view to a collection of collections,



y	flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">Union</a>	Overloaded. Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[ClientView<T> Class](#)  
[C1.Data Namespace](#)

### AsFiltered Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : AsFiltered Method

各要素を適用して条件をテストする関数。

[predicate](#) を使用してサーバー側でビューをフィルタ処理します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Overridable Function AsFiltered( _     ByVal predicate As System.Linq.Expressions.Expression(Of Func(Of T, Boolean))) _     As ClientView(Of T)</pre>	
C#	

```
public virtual ClientView<T> AsFiltered(  
    System.Linq.Expressions.Expression<Func<T, bool>> predicate  
)
```

## Parameters

*predicate*

各要素を適用して条件をテストする関数。

## Return Value

*predicate*を満たすこのビューの要素を含むクライアントビュー。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

## AsFilteredBound<TKey> Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : [AsFilteredBound<TKey> Method](#)

フィルタ処理に使用される値の型。

指定されたビュー項目に対してフィルタ処理を行うためのキー値を返す関数。

キーセレクタ関数および設定可能な **P:C1.Data.FilteredView`1.FilterKey** と [演算子](#) を使用してサーバー上のビューをフィルタ処理します。

## Syntax

Visual Basic (Declaration)

```
Public Overridable Function AsFilteredBound(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
) As FilteredView(Of T)
```

C#

```
public virtual FilteredView<T> AsFilteredBound<TKey>(
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector
)
```

## Parameters

*keySelector*

指定されたビュー項目に対してフィルタ処理を行うためのキー値を返す関数。

## Type Parameters

*TKey*

フィルタ処理に使用される値の型。

## Return Value

条件を満たすキーを持つこのビューの要素を含む [FilteredView<T>](#)。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

## CancelLoad Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : CancelLoad Method

現在のロード操作をキャンセルします。

## Syntax

Visual Basic (Declaration)	
<pre>Public Sub CancelLoad()</pre>	
C#	
<pre>public void CancelLoad()</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

### Load Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : Load() Method

クライアントビューのエンティティをロードします。

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Load()	
C#	
<b>public void</b> Load()	

## Remarks

エンティティが既にキャッシュにある場合は、サーバーへのラウンドトリップは行われません。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

### Paging Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : Paging Method

このクライアントビューにページングを適用します。

# Overload List

Overload	Description
<a href="#">Paging&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Int32)</a>	<a href="#">このクライアントビュー</a> にページングを適用します。
<a href="#">Paging&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Boolean,Int32)</a>	<a href="#">このクライアントビュー</a> にページングを適用します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)

[Paging<TKey>\(Expression<Func<T,TKey>>,Int32\) Method](#)  
[C1.Data Namespace](#) > [ClientView<T> Class](#) > [Paging Method](#) :  
[Paging<TKey>\(Expression<Func<T,TKey>>,Int32\) Method](#)

ソートキーの型。

ソートキーを指定する関数。

[PageSize](#) プロパティの値は、1 ページにロードするアイテムの数を指定します。

[このクライアントビュー](#)にページングを適用します。

## Syntax

Visual Basic (Declaration)
<pre>Public Overloads Function Paging(Of TKey)( _     ByVal sortKeySelector As System.Linq.Expressions.Expression(Of Func(Of T,TKey)), _     ByVal pageSize As System.Integer _ ) As PagingView(Of T)</pre>

C#

```
public PagingView<T> Paging<TKey>(  
    System.Linq.Expressions.Expression<Func<T,TKey>> sortKeySelector,  
    System.int pageSize  
)
```

## Parameters

*sortKeySelector*

ソートキーを指定する関数。

*pageSize*

[PageSize](#) プロパティの値は、1 ページにロードするアイテムの数を指定します。

## Type Parameters

*TKey*

ソートキーの型。

## Return Value

ページブレイク付きのクライアントビュー。

## Remarks

ソートは必須です。エンティティのページングは、ソートなしでは行えません。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)  
[Overload List](#)

[Paging<TKey>\(Expression<Func<T,TKey>>,Boolean,Int32\) Method](#)

[C1.Data Namespace](#) > [ClientView<T> Class](#) > [Paging Method](#) :

[Paging<TKey>\(Expression<Func<T,TKey>>,Boolean,Int32\) Method](#)

ソートキーの型。

ソートキーを指定する関数。

ソートを昇順で行うかどうかを示す boolean 値（false の場合は降順）。

`PageSize` プロパティの値は、1 ページにロードするアイテムの数を指定します。

この[クライアントビュー](#)にページングを適用します。

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Paging(Of TKey)( _  
    ByVal sortKeySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey))), _  
    ByVal ascending As System.Boolean, _  
    ByVal pageSize As System.Integer _  
) As PagingView(Of T)
```

C#

```
public PagingView<T> Paging<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> sortKeySelector,  
    System.bool ascending,  
    System.int pageSize  
)
```

### Parameters

*sortKeySelector*

ソートキーを指定する関数。

*ascending*

ソートを昇順で行うかどうかを示す boolean 値（false の場合は降順）。

*pageSize*

`PageSize` プロパティの値は、1 ページにロードするアイテムの数を指定します。

### Type Parameters

*TKey*

ソートキーの型。

### Return Value

ページブレイク付きのクライアントビュー。

## Remarks

ソートは必須です。エンティティのページングは、ソートなしでは行えません。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)  
[Overload List](#)

### ProgressiveLoading Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : ProgressiveLoading Method

[クライアントビュー](#)のロードをサーバーへの1回のトリップで行うのではなく、複数のバッチで行い、それぞれのロードにおけるページサイズを制限することで、すべてのページのロードが完了しないうちにユーザーが結果を確認して操作できるようにするように指定します。

## Overload List

Overload	Description
<a href="#">ProgressiveLoading&lt;TKey&gt;(Expression&lt;Func&lt;T, TKey&gt;&gt;,Int32)</a>	<a href="#">クライアントビュー</a> のロードをサーバーへの1回のトリップで行うのではなく、複数のバッチで行い、それぞれのロードにおけるページサイズを制限することで、すべてのページのロードが完了しないうちにユーザーが結果を確認して操作できるようにするように指定します。
<a href="#">ProgressiveLoading&lt;TKey&gt;(Expression&lt;Func&lt;T, TKey&gt;&gt;,Boolean,Int32)</a>	<a href="#">クライアントビュー</a> のロードをサーバーへの1回のトリップで行うのではなく、複数のバッチで行い、それぞれのロードにおけるページサイズを制限することで、すべてのページのロードが完了しないうちにユーザ



	一が結果を確認して操作できるようにするように指定します。
--	------------------------------

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

[ProgressiveLoading<TKey>\(Expression<Func<T,TKey>>,Int32\) Method](#)

[C1.Data Namespace](#) > [ClientView<T> Class](#) > [ProgressiveLoading Method](#) :

[ProgressiveLoading<TKey>\(Expression<Func<T,TKey>>,Int32\) Method](#)

ソートキーの型。

ソートキーを指定する関数。

ページのサイズ。

[クライアントビュー](#)のロードをサーバーへの1回のトリップで行うのではなく、複数のバッチで行い、それぞれのロードにおけるページサイズを制限することで、すべてのページのロードが完了しないうちにユーザーが結果を確認して操作できるようにするように指定します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function ProgressiveLoading(Of TKey)( _     ByVal sortKeySelector As System.Linq.Expressions.Expression(Of Func(Of T,TKey)), _     ByVal LoadSize As System.Integer _ ) As ProgressiveView(Of T)</pre>	
C#	
<pre>public ProgressiveView&lt;T&gt; ProgressiveLoading&lt;TKey&gt;(     System.Linq.Expressions.Expression&lt;Func&lt;T,TKey&gt;&gt; sortKeySelector,     System.int LoadSize )</pre>	

## Parameters

*sortKeySelector*

ソートキーを指定する関数。

*loadSize*

ページのサイズ。

## Type Parameters

*TKey*

ソートキーの型。

## Return Value

ソースビューと同じエンティティのロードをプログレッシブに行う [クライアントビュー](#)。

## Remarks

ソートは必須です。エンティティのプログレッシブロードは、ソートなしでは行えません。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

[Overload List](#)

[ProgressiveLoading<TKey>\(Expression<Func<T,TKey>>,Boolean,Int32\) Method](#)

[C1.Data Namespace](#) > [ClientView<T> Class](#) > [ProgressiveLoading Method](#) :

[ProgressiveLoading<TKey>\(Expression<Func<T,TKey>>,Boolean,Int32\) Method](#)

ソートキーの型。

ソートキーを指定する関数。

ソートを昇順で行うかどうかを示す boolean 値（false の場合は降順）。

ページのサイズ。

クライアントビューのロードをサーバーへの1回のトリップで行うのではなく、複数のバッチで行い、それぞれのロードにおけるページサイズを制限することで、すべてのページのロードが完了しないうちにユーザーが結果を確認して操作できるようにするように指定します。

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function ProgressiveLoading(Of TKey)( _  
    ByVal sortKeySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)), _  
    ByVal ascending As System.Boolean, _  
    ByVal LoadSize As System.Integer _  
) As ProgressiveView(Of T)
```

C#

```
public ProgressiveView<T> ProgressiveLoading<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> sortKeySelector,  
    System.bool ascending,  
    System.int LoadSize  
)
```

### Parameters

*sortKeySelector*

ソートキーを指定する関数。

*ascending*

ソートを昇順で行うかどうかを示す boolean 値（false の場合は降順）。

*loadSize*

ページのサイズ。

### Type Parameters

*TKey*

ソートキーの型。

### Return Value

ソースビューと同じエンティティのロードをプログレッシブに行うクライアントビュー。

## Remarks

ソートは必須です。エンティティのプログレッシブロードは、ソートなしでは行えません。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)  
[Overload List](#)

### Refresh Method

[C1.Data Namespace](#) > [ClientView<T> Class](#) : Refresh Method

クライアント側のキャッシュを無視して、[クライアントビュー](#)のエンティティをロードします。

## Syntax

Visual Basic (Declaration)	
<code>Public Sub Refresh()</code>	
C#	
<code>public void Refresh()</code>	

## Remarks

サーバー上で発生した可能性があるすべての変更によってデータを更新するには、このメソッドを使用します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)





### Properties

[C1.Data Namespace](#) : [ClientView<T> Class](#)

For a list of all members of this type, see [ClientView<T> members](#).

### Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	データへのアクセスがあった場合に <a href="#">クライアントビュー</a> を自動的にロードするかどうかを示す boolean 値を取得または設定します。
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">IsLoaded</a>	<a href="#">クライアントビュー</a> がロードされたかどうかを示す値を取得します。
	<a href="#">IsLoading</a>	<a href="#">クライアントビュー</a> がロード中であるかどうかを示す値を取得します。
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

 <a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">Scope</a>	このクライアントビューが属する <a href="#">クライアントスコープ</a> を取得します。
 <a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## See Also

### Reference

[ClientView<T> Class](#)

[C1.Data Namespace](#)

### AutoLoad Property

[C1.Data Namespace](#) > [ClientView<T> Class](#) : AutoLoad Property

データへのアクセスがあった場合に[クライアントビュー](#)を自動的にロードするかどうかを示す boolean 値を取得または設定します。

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> AutoLoad <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool AutoLoad { <b>get</b> ; <b>set</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)

### IsLoaded Property

[C1.Data Namespace](#) > [ClientView<T> Class](#) : IsLoaded Property

クライアントビューがロードされたかどうかを示す値を取得します。

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property IsLoaded As System.Boolean</code>	
C#	
<code>public System.bool IsLoaded {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)

### IsLoading Property

[C1.Data Namespace](#) > [ClientView<T> Class](#) : IsLoading Property

クライアントビューがロード中であるかどうかを示す値を取得します。

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<code>Public ReadOnly Property IsLoading As System.Boolean</code>
C#
<code>public System.bool IsLoading {get;}</code>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)

[ClientView<T> Members](#)

### Scope Property

[C1.Data Namespace](#) > [ClientView<T> Class](#) : Scope Property

このクライアントビューが属する[クライアントスコープ](#)を取得します。

## Syntax

Visual Basic (Declaration)
<code>Public ReadOnly Property Scope As ClientScope</code>
C#
<code>public ClientScope Scope {get;}</code>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



[ClientView<T> Class](#)

[ClientView<T> Members](#)



# Events

>

Name	Description
 <a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">Loaded</a>	<a href="#">クライアントビューのロード</a> が正常に終了したときに発生します。 また、 <a href="#">ロード</a> 中に例外が生成された場合にも発生します。

[Top](#)

## See Also

### Reference

[ClientView<T> Class](#)  
[C1.Data Namespace](#)

### Loaded Event

[C1.Data Namespace](#) > [ClientView<T> Class](#) : Loaded Event

[クライアントビューのロード](#)が正常に終了したときに発生します。  
また、[ロード](#)中に例外が生成された場合にも発生します。

## Syntax

Visual Basic (Declaration)	
<code>Public Event Loaded As System.EventHandler(Of ClientViewLoadedEventArgs)</code>	
C#	
<code>public event System.EventHandler&lt;ClientViewLoadedEventArgs&gt; Loaded</code>	

## Event Data

The event handler receives an argument of type [ClientViewLoadedEventArgs](#) containing data related to this event. The following **ClientViewLoadedEventArgs** properties provide information specific to this event.

Property	Description
<a href="#">Error</a>	Gets the loading error if the loading failed.

<a href="#">HasError</a>	Gets a value indicating whether the loading has failed. If true, inspect the <a href="#">Error</a> property for details.
<a href="#">IsErrorHandled</a>	Gets a value indicating whether the loading error has been marked as handled by calling <a href="#">MarkErrorAsHandled</a> .
<a href="#">Items</a>	Gets all entities loaded by a <a href="#">client view</a> .
<a href="#">TotalItemCount</a>	Gets the total number of rows for the original query without any paging applied to it.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientView<T> Class](#)  
[ClientView<T> Members](#)

## ClientViewLoadedEventArgs

[C1.Data Namespace](#) : ClientViewLoadedEventArgs Class

[ClientView<T>.Loaded](#) イベントのデータを提供します。

## Object Model

ClientViewLoadedEventArgs

## Syntax

Visual Basic (Declaration)	
<pre>Public Class ClientViewLoadedEventArgs     Inherits System.EventArgs</pre>	
C#	
<pre>public class ClientViewLoadedEventArgs : System.EventArgs</pre>	

# Inheritance Hierarchy

System.Object  
System.EventArgs  
**C1.Data.ClientViewLoadedEventArgs**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Members](#)  
[C1.Data Namespace](#)

### Overview

[C1.Data Namespace](#) : ClientViewLoadedEventArgs Class

[ClientView<T>.Loaded](#) イベントのデータを提供します。

## Object Model

ClientViewLoadedEventArgs

## Syntax

Visual Basic (Declaration)	
<pre>Public Class ClientViewLoadedEventArgs     Inherits System.EventArgs</pre>	
C#	
<pre>public class ClientViewLoadedEventArgs : System.EventArgs</pre>	

## Inheritance Hierarchy

System.Object  
System.EventArgs  
**C1.Data.ClientViewLoadedEventArgs**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Members](#)  
[C1.Data Namespace](#)

## Members

[Properties](#) [Methods](#)

[C1.Data Namespace](#) : [ClientViewLoadedEventArgs](#) Class


The following tables list the members exposed by [ClientViewLoadedEventArgs](#).

## Public Properties

	Name	Description
	<a href="#">Error</a>	ロードが失敗した場合に、ロードエラーを取得します。
	<a href="#">HasError</a>	ロードが失敗したかどうかを示す値を取得します。 true の場合、 <a href="#">Error</a> プロパティの詳細を検査します。
	<a href="#">IsErrorHandled</a>	ロードエラーが <a href="#">MarkErrorAsHandled</a> の呼び出しによって「処理済み」のマークが付けられているかどうかを示す値を取得します。
	<a href="#">Items</a>	<a href="#">クライアントビュー</a> によってロードされるすべてのエンティティを取得します。
	<a href="#">TotalItemCount</a>	元のクエリーで、ページングが適用されていない行の合計数を取得します。
	<a href="#">ValidationErrors</a>	検証エラーを取得します。

[Top](#)

## Public Methods

	Name	Description
	<a href="#">MarkErrorAsHandled</a>	<a href="#">HasError</a> が true の場合、このメソッドはエラーに「処理済み」のマークを付けます。このメソッドが呼び出されない場合は、例外が生成されます。

[Top](#)

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)


[C1.Data Namespace](#)

## Methods

[C1.Data Namespace](#) : [ClientViewLoadedEventArgs Class](#)

For a list of all members of this type, see [ClientViewLoadedEventArgs members](#).

## Public Methods

	Name	Description
	<a href="#">MarkErrorAsHandled</a>	<a href="#">HasError</a> が true の場合、このメソッドはエラーに「処理済み」のマークを付けます。このメソッドが呼び出されない場合は、例外が生成されます。

[Top](#)

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)

[C1.Data Namespace](#)

### MarkErrorAsHandled Method

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : [MarkErrorAsHandled Method](#)

[HasError](#) が true の場合、このメソッドはエラーに「処理済み」のマークを付けます。このメソッドが呼び出されない場合は、例外が生成されます。

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> MarkErrorAsHandled()	
C#	
<b>public void</b> MarkErrorAsHandled()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)

[ClientViewLoadedEventArgs Members](#)


## Properties

[C1.Data Namespace](#) : ClientViewLoadedEventArgs Class

For a list of all members of this type, see [ClientViewLoadedEventArgs members](#).

## Public Properties

	Name	Description
	<a href="#">Error</a>	ロードが失敗した場合に、ロードエラーを取得します。
	<a href="#">HasError</a>	ロードが失敗したかどうかを示す値を取得します。 true の場合、 <a href="#">Error</a> プロパティの詳細を検査します。
	<a href="#">IsErrorHandled</a>	ロードエラーが <a href="#">MarkErrorAsHandled</a> の呼び出しによって「処理済み」のマークが付けられているかどうかを示す値を取得します。
	<a href="#">Items</a>	<a href="#">クライアントビュー</a> によってロードされるすべてのエンティティを取得します。
	<a href="#">TotalItemCount</a>	元のクエリーで、ページングが適用されていない行の合計数を取得します。

	unt	
	ValidationErrors	検証エラーを取得します。

[Top](#)

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)

[C1.Data Namespace](#)

### Error Property

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : Error Property

ロードが失敗した場合に、ロードエラーを取得します。

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Error As System.Exception</code>	
C#	
<code>public System.Exception Error {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)

[ClientViewLoadedEventArgs Members](#)

### HasError Property

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : HasError Property

ロードが失敗したかどうかを示す値を取得します。 true の場合、[Error](#) プロパティの詳細を検査します。

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property HasError As System.Boolean
```

C#

```
public System.bool HasError {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)

[ClientViewLoadedEventArgs Members](#)

### IsErrorHandled Property

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : IsErrorHandled Property

ロードエラーが [MarkErrorAsHandled](#)

の呼び出しによって「処理済み」のマークが付けられているかどうかを示す値を取得します。

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property IsErrorHandled As System.Boolean
```

C#

```
public System.bool IsErrorHandled {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



## Reference

[ClientViewLoadedEventArgs Class](#)

[ClientViewLoadedEventArgs Members](#)

### Items Property

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : Items Property

クライアントビューによってロードされるすべてのエンティティを取得します。

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> Items <b>As</b> System.Collections.IEnumerable	
C#	
<b>public</b> System.Collections.IEnumerable Items { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)

[ClientViewLoadedEventArgs Members](#)

### TotalItemCount Property

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : TotalItemCount Property

元のクエリーで、ページングが適用されていない行の合計数を取得します。

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> TotalItemCount <b>As</b> System.Integer	
C#	
<b>public</b> System.int TotalItemCount { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)

[ClientViewLoadedEventArgs Members](#)

### ValidationErrors Property

[C1.Data Namespace](#) > [ClientViewLoadedEventArgs Class](#) : ValidationErrors Property

検証エラーを取得します。

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> ValidationErrors <b>As</b> System.Collections.Generic.IEnumerable(Of ValidationResult)	
C#	
<b>public</b> System.Collections.Generic.IEnumerable<ValidationResult> ValidationErrors { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewLoadedEventArgs Class](#)

[ClientViewLoadedEventArgs Members](#)

## DataExtensions

[C1.Data Namespace](#) : DataExtensions Class

# Object Model

DataExtensions

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public MustInherit NotInheritable Class DataExtensions</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static class DataExtensions</pre>	

## Inheritance Hierarchy

System.Object  
    **C1.Data.DataExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DataExtensions Members](#)  
[C1.Data Namespace](#)

### Overview

[C1.Data Namespace](#) : DataExtensions Class

## Object Model

DataExtensions

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public MustInherit NotInheritable Class DataExtensions

C#

[System.Runtime.CompilerServices.Extension()  
public static class DataExtensions

## Inheritance Hierarchy

System.Object  
    **C1.Data.DataExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DataExtensions Members](#)  
[C1.Data Namespace](#)

## Members

[Methods](#)

[C1.Data Namespace](#) : DataExtensions Class

The following tables list the members exposed by [DataExtensions](#).

## Public Methods

	Name	Description
	<a href="#">ExecuteIn</a>	Overloaded.

[Top](#)

## See Also

### Reference

[DataExtensions Class](#)  
[C1.Data Namespace](#)

# Methods

[C1.Data Namespace](#) : DataExtensions Class

For a list of all members of this type, see [DataExtensions members](#).

## Public Methods

	Name	Description
	<a href="#">ExecuteIn</a>	Overloaded.

[Top](#)

## See Also

### Reference

[DataExtensions Class](#)

[C1.Data Namespace](#)

### ExecuteIn Method

[C1.Data Namespace](#) > [DataExtensions Class](#) : ExecuteIn Method

## Overload List

Overload	Description
<a href="#">ExecuteIn&lt;T&gt;(IEnumerable&lt;T&gt;,ClientScope)</a>	
<a href="#">ExecuteIn&lt;T&gt;(EntityQuery&lt;T&gt;,RiaClientScope)</a>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DataExtensions Class](#)

[DataExtensions Members](#)

ExecuteIn<T>(IEnumerable<T>,ClientScope) Method

[C1.Data Namespace](#) > [DataExtensions Class](#) > [ExecuteIn Method](#) :

ExecuteIn<T>(IEnumerable<T>,ClientScope) Method

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function ExecuteIn(Of T)( _
    ByVal query As System.Collections.Generic.IEnumerable(Of T), _
    ByVal scope As ClientScope _
) As System.Collections.Generic.IEnumerable(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static System.Collections.Generic.IEnumerable<T> ExecuteIn<T>(
    System.Collections.Generic.IEnumerable<T> query,
    ClientScope scope
)
```

### Parameters

*query*

*scope*

### Type Parameters

*T*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DataExtensions Class](#)  
[DataExtensions Members](#)  
[Overload List](#)

ExecuteIn<T>(EntityQuery<T>,RiaClientScope) Method

[C1.Data Namespace](#) > [DataExtensions Class](#) > [ExecuteIn Method](#) :

ExecuteIn<T>(EntityQuery<T>,RiaClientScope) Method

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function ExecuteIn(Of T As
System.ServiceModel.DomainServices.Client.Entity)( _
    ByVal query As System.ServiceModel.DomainServices.Client.EntityQuery(Of
T), _
    ByVal scope As RiaClientScope _
) As System.ServiceModel.DomainServices.Client.LoadOperation(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static System.ServiceModel.DomainServices.Client.LoadOperation<T>
ExecuteIn<T>(
    System.ServiceModel.DomainServices.Client.EntityQuery<T> query,
    RiaClientScope scope
)
where T: System.ServiceModel.DomainServices.Client.Entity
```

### Parameters

*query*

*scope*

### Type Parameters

*T*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DataExtensions Class](#)  
[DataExtensions Members](#)  
[Overload List](#)

# FilteredView<T>

[C1.Data Namespace](#) : FilteredView<T> Class

## Object Model

FilteredView<T>

## Syntax

Visual Basic (Declaration)

```
Public Class FilteredView(Of T)
    Inherits ClientView(Of T)
    Implements C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class FilteredView<T> : ClientView<T>,
    C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

## Inheritance Hierarchy

System.Object  
    [C1.LiveLinq.LiveViews.View](#)  
        [C1.LiveLinq.LiveViews.View<T>](#)  
            [C1.Data.ClientView<T>](#)  
                **C1.Data.FilteredView<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilteredView<T> Members](#)  
[C1.Data Namespace](#)



## Overview

[C1.Data Namespace](#) : [FilteredView<T>](#) Class

## Object Model

[FilteredView<T>](#)

## Syntax

Visual Basic (Declaration)

```
Public Class FilteredView(Of T)
    Inherits ClientView(Of T)
    Implements C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class FilteredView<T> : ClientView<T>,
    C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

## Inheritance Hierarchy

System.Object  
    [C1.LiveLinq.LiveViews.View](#)  
        [C1.LiveLinq.LiveViews.View<T>](#)  
            [C1.Data.ClientView<T>](#)  
                **[C1.Data.FilteredView<T>](#)**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilteredView<T> Members](#)  
[C1.Data Namespace](#)

## Members

[Fields](#) [Properties](#) [Methods](#) [Events](#)

[C1.Data Namespace](#) : [FilteredView<T>](#) Class

The following tables list the members exposed by [FilteredView<T>](#).

### Public Fields

	Name	Description
 <b>S</b>	<a href="#">Unfiltered</a>	

[Top](#)


### Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	データへのアクセスがあった場合に <a href="#">クライアントビュー</a> を自動的にロードするかどうかを示す boolean 値を取得または設定します。(Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">FilterKey</a>	
	<a href="#">FilterKeyType</a>	
	<a href="#">IsLoaded</a>	<a href="#">クライアントビュー</a> がロードされたかどうかを示す値を取得します。(Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsLoading</a>	<a href="#">クライアントビュー</a> がロード中であるかどうかを示す値を取得します。

		(Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Operator</a>	
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Scope</a>	このクライアントビューが属する <a href="#">クライアントスコープ</a> を取得します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## Public Methods

Name	Description
 <a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> .

ory	(Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">C1.LiveLinq.LiveViews.View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <a href="#">AsFiltered</a>	<a href="#">predicate</a> を使用してサーバー側でビューをフィルタ処理します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <a href="#">AsFilteredBound&lt;TKey&gt;</a>	キーセレクタ関数および設定可能な <b>P:C1.Data.FilteredView`1.FilterKey</b> と <b>演算子</b> を使用して サーバー上のビューをフィルタ処理します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">BindFilterKey</a>	Overloaded.
⇒ <a href="#">CancelLoad</a>	現在のロード操作をキャンセルします。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <a href="#">Concat</a>	Overloaded. Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

⇒ <a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">GroupJoin</a>	Overloaded. Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">Join</a>	Overloaded. Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">Load</a>	<a href="#">クライアントビュー</a> のエンティティをロードします。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <a href="#">OrderBy&lt;T Key&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <a href="#">Paging</a>	Overloaded. この <a href="#">クライアントビュー</a> にページングを適用します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <a href="#">ProgressiveLoading</a>	Overloaded. <a href="#">クライアントビュー</a> のロードをサーバーへの1回のトリップで行うのではなく、複数のバッチで行い、それぞれのロードにおけるページサイズを制限することで、すべてのページのロードが完了しないうちにユーザーが結果を確認して操作できるようにするように指定します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <a href="#">PurgeEmpty</a>	Remove empty groups from a grouping view. (Inherited from

yGroups	C1.LiveLinq.LiveViews.View)
⇒ Rebuild	Re-populates the view by re-executing the view's query. (Inherited from C1.LiveLinq.LiveViews.View)
⇒ Refresh	クライアント側のキャッシュを無視して、クライアントビューのエンティティをロードします。 (Inherited from C1.Data.ClientView<T>)
⇒ Select<TResult>	Projects each element of a view into a new form. (Inherited from C1.LiveLinq.LiveViews.View<T>)
⇒ SelectMany	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from C1.LiveLinq.LiveViews.View<T>)
⇒ SetTransaction	Sets the value of the Transaction property. (Inherited from C1.LiveLinq.LiveViews.View)
⇒ ToString	Returns a string representing this view. (Inherited from C1.LiveLinq.LiveViews.View<T>)
⇒ Union	Overloaded. Set union of two views. (Inherited from C1.LiveLinq.LiveViews.View<T>)
⇒ Where	Filters the source view based on a predicate. (Inherited from C1.LiveLinq.LiveViews.View<T>)

[Top](#)

## Public Events

	Name	Description
⚡	Changed	Occurs after an item of the view or the entire view has changed. (Inherited from C1.LiveLinq.LiveViews.View<T>)
⚡	Loaded	クライアントビューのロードが正常に終了したときに発生します。また、ロード中に例外が生成された場合にも発生します。 (Inherited

		from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
--	--	--

[Top](#)

## See Also

### Reference

[FilteredView<T> Class](#)





[C1.Data Namespace](#)

## Methods

[C1.Data Namespace](#) : [FilteredView<T> Class](#)

For a list of all members of this type, see [FilteredView<T> members](#).




## Public Methods

Name	Description
 <a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">C1.LiveLinq.LiveViews.View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">AsFiltered</a>	<a href="#">predicate</a> を使用してサーバー側でビューをフィルタ処理します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <a href="#">AsFilteredBound&lt;TKey&gt;</a>	キーセレクタ関数および設定可能な <b>P:C1.Data.FilteredView`1.FilterKey</b> と <a href="#">演算子</a> を使用して サーバー上のビューをフィルタ処理します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

⇒ BindFilterKey	Overloaded.
⇒ CancelLoad	現在のロード操作をキャンセルします。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ Concat	Overloaded. Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ Contains	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ DeferMaintenance	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ GetEnumerator	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ GroupBy	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ GroupJoin	Overloaded. Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ IndexOf	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ Join	Overloaded. Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ Load	クライアントビューのエンティティをロードします。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ Maintain	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )



⇒ <b>OrderBy&lt;TKey&gt;</b>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>OrderByDescending&lt;TKey&gt;</b>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>Paging</b>	Overloaded. このクライアントビューにページングを適用します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <b>ProgressiveLoading</b>	Overloaded. クライアントビューのロードをサーバーへの1回のトリップで行うのではなく、複数のバッチで行い、それぞれのロードにおけるページサイズを制限することで、すべてのページのロードが完了しないうちにユーザーが結果を確認して操作できるようにするように指定します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <b>PurgeEmptyGroups</b>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <b>Rebuild</b>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <b>Refresh</b>	クライアント側のキャッシュを無視して、クライアントビューのエンティティをロードします。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <b>Select&lt;TResult&gt;</b>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>SelectMany</b>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>SetTransaction</b>	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

 <a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">Union</a>	Overloaded. Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[FilteredView<T> Class](#)

[C1.Data Namespace](#)

### BindFilterKey Method

[C1.Data Namespace](#) > [FilteredView<T> Class](#) : BindFilterKey Method

## Overload List

Overload	Description
<a href="#">BindFilterKey(Binding)</a>	
<a href="#">BindFilterKey(Object,String)</a>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilteredView<T> Class](#)

[FilteredView<T> Members](#)

## BindFilterKey(Binding) Method

[C1.Data Namespace](#) > [FilteredView<T> Class](#) > [BindFilterKey Method](#) : BindFilterKey(Binding) Method

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function BindFilterKey( _     ByVal binding As System.Windows.Data.Binding _ ) As FilteredView(Of T)</pre>	
C#	
<pre>public FilteredView&lt;T&gt; BindFilterKey(     System.Windows.Data.Binding binding )</pre>	

### Parameters

*binding*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilteredView<T> Class](#)  
[FilteredView<T> Members](#)  
[Overload List](#)

## BindFilterKey(Object,String) Method

[C1.Data Namespace](#) > [FilteredView<T> Class](#) > [BindFilterKey Method](#) : BindFilterKey(Object,String) Method

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function BindFilterKey( _     ByVal source As System.Object, _     ByVal path As System.String _</pre>	

```
) As FilteredView(Of T)
```

C#

```
public FilteredView<T> BindFilterKey(  
    System.object source,  
    System.string path  
)
```

## Parameters

*source*

*path*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[FilteredView<T> Class](#)

[FilteredView<T> Members](#)



[Overload List](#)




## Properties


[C1.Data Namespace](#) : [FilteredView<T> Class](#)

For a list of all members of this type, see [FilteredView<T> members](#).

## Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	データへのアクセスがあった場合にクライアントビューを自動的にロードするかどうかを示す boolean 値を取得または設定します。(Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

 <b>CurrentItem</b>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>DeferredMaintenance</b>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>FilterKey</b>	
 <b>FilterKeyType</b>	
 <b>IsLoaded</b>	<b>クライアントビューがロードされたかどうかを示す値を取得します。</b> (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <b>IsLoading</b>	<b>クライアントビューがロード中であるかどうかを示す値を取得します。</b> (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <b>IsReadOnly</b>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>Item</b>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <b>MaintenanceMode</b>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>MoveToFirstOnReset</b>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>Operator</b>	
 <b>Order</b>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>Scope</b>	このクライアントビューが属する <b>クライアントスコープ</b> を取得します。

		(Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## See Also

### Reference

[FilteredView<T> Class](#)

[C1.Data Namespace](#)

### FilterKey Property

[C1.Data Namespace](#) > [FilteredView<T> Class](#) : FilterKey Property

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> FilterKey <b>As</b> System.Object	
C#	
<b>public</b> System.object FilterKey { <b>get</b> ; <b>set</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilteredView<T> Class](#)

[FilteredView<T> Members](#)

### FilterKeyType Property

[C1.Data Namespace](#) > [FilteredView<T> Class](#) : FilterKeyType Property

# Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property FilterKeyType As System.Type</code>	
C#	
<code>public System.Type FilterKeyType {get;}</code>	

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [FilteredView<T> Class](#)
- [FilteredView<T> Members](#)

## Operator Property

[C1.Data Namespace](#) > [FilteredView<T> Class](#) : Operator Property

# Syntax

Visual Basic (Declaration)	
<code>Public Property Operator As System.Windows.Controls.FilterOperator</code>	
C#	
<code>public System.Windows.Controls.FilterOperator Operator {get; set;}</code>	

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[FilteredView<T> Class](#)  
[FilteredView<T> Members](#)

## Fields

>

Name	Description
 <a href="#">S</a> <a href="#">Unfiltered</a>	

[Top](#)

## See Also

### Reference

[FilteredView<T> Class](#)  
[C1.Data Namespace](#)

### Unfiltered Field

[C1.Data Namespace](#) > [FilteredView<T> Class](#) : Unfiltered Field

## Syntax

Visual Basic (Declaration)	
<code>Public Shared ReadOnly Unfiltered As System.Object</code>	
C#	
<code>public static readonly System.object Unfiltered</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[FilteredView<T> Class](#)  
[FilteredView<T> Members](#)



# PagingView<T>

[C1.Data Namespace](#) : PagingView<T> Class

クライアントビュー内のエンティティの型。

ページブレーク付きの [クライアントビュー](#)を表します。

## Object Model

PagingView<T>

## Syntax

Visual Basic (Declaration)

```
Public Class PagingView(Of T)
    Inherits ClientView(Of T)
    Implements C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class PagingView<T> : ClientView<T>, C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

クライアントビュー内のエンティティの型。

## Remarks

ページングはサーバー上で実行されます。 [PageSize](#) プロパティと [PageIndex](#) プロパティで制御されます。

ソートは必須です。エンティティのページングは、ソートなしでは行えません。

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

[C1.Data.ClientView<T>](#)

**C1.Data.PagingView<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PagingView<T> Members](#)  
[C1.Data Namespace](#)

### Overview

[C1.Data Namespace](#) : [PagingView<T>](#) Class

[クライアントビュー](#)内のエンティティの型。

ページブレーク付きの [クライアントビュー](#)を表します。

## Object Model

[PagingView<T>](#)

### Syntax

Visual Basic (Declaration)	
<pre>Public Class PagingView(Of T)     Inherits ClientView(Of T)     Implements C1.LiveLinq.IObservableSource(Of T)</pre>	
C#	
<pre>public class PagingView&lt;T&gt; : ClientView&lt;T&gt;, C1.LiveLinq.IObservableSource&lt;T&gt;</pre>	

### Type Parameters

*T*

[クライアントビュー](#)内のエンティティの型。

### Remarks

ページングはサーバー上で実行されます。 [PageSize](#) プロパティと [PageIndex](#) プロパティで制御されます。

ソートは必須です。エンティティのページングは、ソートなしでは行えません。

## Inheritance Hierarchy

System.Object  
  C1.LiveLinq.LiveViews.View  
    C1.LiveLinq.LiveViews.View<T>  
      C1.Data.ClientView<T>  
        **C1.Data.PagingView<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PagingView<T> Members](#)  
[C1.Data Namespace](#)





### Members












[Properties](#) [Methods](#) [Events](#)



[C1.Data Namespace](#) : [PagingView<T>](#) Class

The following tables list the members exposed by [PagingView<T>](#).

### Public Properties

	Name	Description
	<a href="#">AutoLoad</a>	データへのアクセスがあった場合に <a href="#">クライアントビュー</a> を自動的にロードするかどうかを示す boolean 値を取得または設定します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMainte</a>	Gets the effective value of MaintenanceMode. (Inherited from

	nance	<a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">IsLoaded</a>	クライアントビューがロードされたかどうかを示す値を取得します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsLoading</a>	クライアントビューがロード中であるかどうかを示す値を取得します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">LoadSize</a>	1回のバッチでロードするエンティティの数を制御する値を取得または設定します。
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">PageCount</a>	このビュー内のページ数を取得します。
	<a href="#">PageIndex</a>	現在のページのインデックスを取得または設定します。
	<a href="#">PageSize</a>	1ページ内の項目数を取得または設定します。
	<a href="#">Scope</a>	このクライアントビューが属するクライアントスコープを取得します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )

 <b>TotalItemCount</b>	ページング適用前のビュー内のエンティティの合計数を取得します。
 <b>Transaction</b>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )


[Top](#)

## Public Methods

Name	Description
 <b>AsCollectionViewFactory</b>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>AsDynamic</b>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">C1.LiveLinq.LiveViews.View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>AsFiltered</b>	<a href="#">predicate</a> を使用してサーバー側でビューをフィルタ処理します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <b>AsFilteredBound&lt;TKey&gt;</b>	キーセレクタ関数および設定可能な <b>P:C1.Data.FilteredView`1.FilterKey</b> と <b>演算子</b> を使用して サーバー上のビューをフィルタ処理します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <b>AttachAggregationView</b>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <b>AttachView</b>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <b>CancelLoaded</b>	現在の <b>ロード操作</b> をキャンセルします。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )



⇒ <b>Concat</b>	Overloaded. Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>Contains</b>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>DeferMaintenance</b>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <b>GetEnumerator</b>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>GroupBy</b>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>GroupJoin</b>	Overloaded. Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>IndexOf</b>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>Join</b>	Overloaded. Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>Load</b>	クライアントビューのエンティティをロードします。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <b>Maintain</b>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <b>OrderBy&lt;T Key&gt;</b>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>OrderByDescending&lt;T Key&gt;</b>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

TKey>	
⇒Paging	Overloaded. このクライアントビューにページングを適用します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ProgressiveLoading	Overloaded. クライアントビューのロードをサーバーへの1回のトリップで行うのではなく、複数のバッチで行い、それぞれのロードにおけるページサイズを制限することで、すべてのページのロードが完了しないうちにユーザーが結果を確認して操作できるようにするように指定します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒PurgeEmptyGroups	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒Rebuild	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒Refresh	クライアント側のキャッシュを無視して、クライアントビューのエンティティをロードします。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒Select<TResult>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒SelectMany	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒SetTransaction	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ToString	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒Union	Overloaded. Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

 <b>Where</b>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
--	---

[Top](#)

## Public Events

	Name	Description
	<b>Changed</b>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<b>Loaded</b>	<a href="#">クライアントビューのロード</a> が正常に終了したときに発生します。また、 <a href="#">ロード</a> 中に例外が生成された場合にも発生します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[PagingView<T> Class](#)



[C1.Data Namespace](#)

## Properties

[C1.Data Namespace](#) : [PagingView<T> Class](#)





For a list of all members of this type, see [PagingView<T> members](#).

## Public Properties

	Name	Description
	<b>AutoLoad</b>	データへのアクセスがあった場合に <a href="#">クライアントビュー</a> を自動的にロードするかどうかを示す boolean 値を取得または設定します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<b>Count</b>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )



 <b>CurrentItem</b>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>DeferredMaintenance</b>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>IsLoaded</b>	<a href="#">クライアントビューがロード</a> されたかどうかを示す値を取得します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <b>IsLoading</b>	<a href="#">クライアントビューがロード中</a> であるかどうかを示す値を取得します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <b>IsReadOnly</b>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>Item</b>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <b>LoadSize</b>	1回のバッチでロードするエンティティの数を制御する値を取得または設定します。
 <b>MaintenanceMode</b>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>MoveToFirstOnReset</b>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>Order</b>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>PageCount</b>	この <a href="#">ビュー</a> 内のページ数を取得します。
 <b>PageIndex</b>	現在のページのインデックスを取得または設定します。

 <a href="#">PageSize</a>	1 ページ内の項目数を取得または設定します。
 <a href="#">Scope</a>	このクライアントビューが属する <a href="#">クライアントスコープ</a> を取得します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <a href="#">TotalItemCount</a>	ページング適用前のビュー内のエンティティの合計数を取得します。
 <a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## See Also

### Reference

[PagingView<T> Class](#)

[C1.Data Namespace](#)

### LoadSize Property

[C1.Data Namespace](#) > [PagingView<T> Class](#) : LoadSize Property

1 回のバッチでロードするエンティティの数を制御する値を取得または設定します。

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> LoadSize <b>As</b> System.Integer	
C#	
<b>public</b> System.int LoadSize { <b>get</b> ; <b>set</b> ;}	

## Remarks

エンティティは、[LoadSize](#) に最も近い [PageSize](#) の倍数を使用してロードされます。  
これにより、ページを部分的にロードすることなく、複数のページを一度にロードできます。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PagingView<T> Class](#)

[PagingView<T> Members](#)

### PageCount Property

[C1.Data Namespace](#) > [PagingView<T> Class](#) : PageCount Property

このビュー内のページ数を取得します。

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> PageCount <b>As</b> System.Integer	
C#	
<b>public</b> System.int PageCount { <b>get</b> ;}	

## Remarks

[PageSize](#) が 0 の場合は、[PageCount](#) も 0 になります。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PagingView<T> Class](#)

[PagingView<T> Members](#)

### PageIndex Property

[C1.Data Namespace](#) > [PagingView<T> Class](#) : PageIndex Property

現在のページのインデックスを取得または設定します。

## Syntax

Visual Basic (Declaration)

```
Public Property PageIndex As System.Integer
```

C#

```
public System.int PageIndex {get; set;}
```

## Remarks

このプロパティ値を無効な値に設定すると、無視されます。 0未満または [PageCount](#) 以上の値は無効です。

このビュー内に項目がない場合は、0だけがこのプロパティの有効な値です。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PagingView<T> Class](#)

[PagingView<T> Members](#)

### PageSize Property

[C1.Data Namespace](#) > [PagingView<T> Class](#) : PageSize Property

1 ページ内の項目数を取得または設定します。

## Syntax

Visual Basic (Declaration)

```
Public Property PageSize As System.Integer
```

C#

```
public System.int PageSize {get; set;}
```

## Remarks

ページングを無効にする場合は、このプロパティを0に設定します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PagingView<T> Class](#)

[PagingView<T> Members](#)

### TotalItemCount Property

[C1.Data Namespace](#) > [PagingView<T> Class](#) : TotalItemCount Property

ページング適用前のビュー内のエンティティの合計数を取得します。

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> TotalItemCount <b>As</b> System.Integer	
C#	
<b>public</b> System.int TotalItemCount { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PagingView<T> Class](#)

[PagingView<T> Members](#)

### ProgressiveView<T>

[C1.Data Namespace](#) : ProgressiveView<T> Class

## Object Model

## Syntax

Visual Basic (Declaration)

```
Public Class ProgressiveView(Of T)
    Inherits ClientView(Of T)
    Implements C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class ProgressiveView<T> : ClientView<T>,
    C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

## Inheritance Hierarchy

```
System.Object
    C1.LiveLinq.LiveViews.View
        C1.LiveLinq.LiveViews.View<T>
            C1.Data.ClientView<T>
                C1.Data.ProgressiveView<T>
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ProgressiveView<T> Members](#)  
[C1.Data Namespace](#)

### Overview

[C1.Data Namespace](#) : [ProgressiveView<T> Class](#)

## Object Model

# Syntax

Visual Basic (Declaration)	
<pre>Public Class ProgressiveView(Of T)     Inherits ClientView(Of T)     Implements C1.LiveLinq.IObservableSource(Of T)</pre>	
C#	
<pre>public class ProgressiveView&lt;T&gt; : ClientView&lt;T&gt;, C1.LiveLinq.IObservableSource&lt;T&gt;</pre>	

## Type Parameters

*T*

## Inheritance Hierarchy

System.Object  
    C1.LiveLinq.LiveViews.View  
        C1.LiveLinq.LiveViews.View<T>  
            C1.Data.ClientView<T>  
                **C1.Data.ProgressiveView<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ProgressiveView<T> Members](#)  
[C1.Data Namespace](#)













### Members

[Properties](#) [Methods](#) [Events](#)



[C1.Data Namespace](#) : [ProgressiveView<T> Class](#)

The following tables list the members exposed by [ProgressiveView<T>](#).

## Public Properties







	Name	Description
	<a href="#">AutoLoad</a>	データへのアクセスがあった場合に <a href="#">クライアントビュー</a> を自動的にロードするかどうかを示す boolean 値を取得または設定します。(Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">IsLoaded</a>	<a href="#">クライアントビュー</a> がロードされたかどうかを示す値を取得します。(Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsLoading</a>	<a href="#">クライアントビュー</a> がロード中であるかどうかを示す値を取得します。(Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">LoadSize</a>	
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if




		it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Scope</a>	このクライアントビューが属する <a href="#">クライアントスコープ</a> を取得します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )



[Top](#)

## Public Methods

Name	Description
 <a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">C1.LiveLinq.LiveViews.View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">AsFiltered</a>	<a href="#">predicate</a> を使用してサーバー側でビューをフィルタ処理します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <a href="#">AsFilteredBound&lt;TKey&gt;</a>	キーセレクタ関数および設定可能な <b>P:C1.Data.FilteredView`1.FilterKey</b> と <b>演算子</b> を使用して サーバー上のビューをフィルタ処理します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )



⇒  <b>CancelLoad</b>	現在のロード操作をキャンセルします。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒  <b>Concat</b>	Overloaded. Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <b>Contains</b>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <b>DeferMaintenance</b>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒  <b>GetEnumerator</b>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <b>GroupBy</b>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <b>GroupJoin</b>	Overloaded. Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <b>IndexOf</b>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <b>Join</b>	Overloaded. Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒  <b>Load</b>	クライアントビューのエンティティをロードします。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒  <b>Maintain</b>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒  <b>OrderBy&lt;T Key&gt;</b>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

⇒ <b>OrderByDescending&lt;TKey&gt;</b>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>Paging</b>	Overloaded. このクライアントビューにページングを適用します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <b>ProgressiveLoading</b>	Overloaded. クライアントビューのロードをサーバーへの1回のトリップで行うのではなく、複数のバッチで行い、それぞれのロードにおけるページサイズを制限することで、すべてのページのロードが完了しないうちにユーザーが結果を確認して操作できるようにするように指定します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <b>PurgeEmptyGroups</b>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <b>Rebuild</b>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <b>Refresh</b>	クライアント側のキャッシュを無視して、クライアントビューのエンティティをロードします。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
⇒ <b>Select&lt;TResult&gt;</b>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>SelectMany</b>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒ <b>SetTransaction</b>	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒ <b>ToString</b>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

 <b>Union</b>	Overloaded. Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <b>Where</b>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## Public Events

	Name	Description
	<b>Changed</b>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<b>Loaded</b>	<a href="#">クライアントビューのロード</a> が正常に終了したときに発生します。また、 <a href="#">ロード</a> 中に例外が生成された場合にも発生します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[ProgressiveView<T> Class](#)


[C1.Data Namespace](#)













## Properties


[C1.Data Namespace](#) : [ProgressiveView<T> Class](#)

For a list of all members of this type, see [ProgressiveView<T> members](#).

## Public Properties

	Name	Description
	<b>AutoLoad</b>	データへのアクセスがあった場合に <a href="#">クライアントビュー</a> を自動的にロードするかどうかを示す boolean 値を取得または設定します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )

 <b>Count</b>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>CurrentItem</b>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>DeferredMaintenance</b>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>IsLoaded</b>	クライアントビューがロードされたかどうかを示す値を取得します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <b>IsLoading</b>	クライアントビューがロード中であるかどうかを示す値を取得します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )
 <b>IsReadOnly</b>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>Item</b>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <b>LoadSize</b>	
 <b>MaintenanceMode</b>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>MoveToFirstOnReset</b>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>Order</b>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <b>Scope</b>	このクライアントビューが属するクライアントスコープを取得します。 (Inherited from <a href="#">C1.Data.ClientView&lt;T&gt;</a> )

	<b>Transaction</b>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
---	--------------------	---

[Top](#)

## See Also

### Reference

[ProgressiveView<T> Class](#)

[C1.Data Namespace](#)

### LoadSize Property

[C1.Data Namespace](#) > [ProgressiveView<T> Class](#) : LoadSize Property

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> LoadSize <b>As</b> System.Integer	
C#	
<b>public</b> System.int LoadSize { <b>get</b> ; <b>set</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ProgressiveView<T> Class](#)

[ProgressiveView<T> Members](#)

## SavedChangesEventArgs

[C1.Data Namespace](#) : SavedChangesEventArgs Class

C1DataSource.SavedChanges イベントのデータを提供します。

# Object Model

SavedChangesEventArgs

## Syntax

Visual Basic (Declaration)	
<pre>Public Class SavedChangesEventArgs     Inherits System.EventArgs</pre>	
C#	
<pre>public class SavedChangesEventArgs : System.EventArgs</pre>	

## Inheritance Hierarchy

System.Object  
    System.EventArgs  
        **C1.Data.SavedChangesEventArgs**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SavedChangesEventArgs Members](#)  
[C1.Data Namespace](#)

### Overview

[C1.Data Namespace](#) : SavedChangesEventArgs Class

C1DataSource.SavedChanges イベントのデータを提供します。

# Object Model

SavedChangesEventArgs

## Syntax

Visual Basic (Declaration)	
<pre>Public Class SavedChangesEventArgs     Inherits System.EventArgs</pre>	
C#	
<pre>public class SavedChangesEventArgs : System.EventArgs</pre>	

## Inheritance Hierarchy

System.Object  
 System.EventArgs  
**C1.Data.SavedChangesEventArgs**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SavedChangesEventArgs Members](#)  
[C1.Data Namespace](#)




## Members

[Properties](#) [Methods](#)

[C1.Data Namespace](#) : SavedChangesEventArgs Class

The following tables list the members exposed by [SavedChangesEventArgs](#).

## Public Properties

	Name	Description
	<a href="#">Error</a>	保存操作中に発生したエラーを示す値を取得します。
	<a href="#">HasError</a>	保存操作が失敗したかどうかを示す値を取得します。 true の場合、 <a href="#">Error</a> プロパティの詳細を検査します。
	<a href="#">IsErrorHand</a>	エラーが <a href="#">MarkErrorAsHandled</a>



	<a href="#">led</a>	の呼び出しによって「処理済み」のマークが付けられているかどうかを示す値を取得します。
--	---------------------	--

[Top](#)

## Public Methods

	Name	Description
⇒💎	<a href="#">MarkErrorAsHandled</a>	このメソッドが失敗した操作に対して呼び出された場合（ <a href="#">HasError</a> が true の場合）、エラーに処理済みのマークが付けられます。 そうでない場合は、例外が生成されます。

[Top](#)

## See Also

### Reference

[SavedChangesEventArgs Class](#)

[C1.Data Namespace](#)

## Methods

[C1.Data Namespace](#) : [SavedChangesEventArgs Class](#)

For a list of all members of this type, see [SavedChangesEventArgs members](#).

## Public Methods

	Name	Description
⇒💎	<a href="#">MarkErrorAsHandled</a>	このメソッドが失敗した操作に対して呼び出された場合（ <a href="#">HasError</a> が true の場合）、エラーに処理済みのマークが付けられます。 そうでない場合は、例外が生成されます。

[Top](#)

## See Also

### Reference

[SavedChangesEventArgs Class](#)

[C1.Data Namespace](#)

# MarkErrorAsHandled Method

C1.Data Namespace > [SavedChangesEventArgs Class](#) : MarkErrorAsHandled Method

このメソッドが失敗した操作に対して呼び出された場合（[HasError](#) が true の場合）、エラーに処理済みのマークが付けられます。  
そうでない場合は、例外が生成されます。

## Syntax

Visual Basic (Declaration)	
<code>Public Sub MarkErrorAsHandled()</code>	
C#	
<code>public void MarkErrorAsHandled()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference




[SavedChangesEventArgs Class](#)  
[SavedChangesEventArgs Members](#)

## Properties

C1.Data Namespace : [SavedChangesEventArgs Class](#)

For a list of all members of this type, see [SavedChangesEventArgs members](#).

## Public Properties

	Name	Description
	<a href="#">Error</a>	保存操作中に発生したエラーを示す値を取得します。
	<a href="#">HasError</a>	保存操作が失敗したかどうかを示す値を取得します。 true の場合、 <a href="#">Error</a> プロパティの詳細を検査します。
	<a href="#">IsErrorHandled</a>	エラーが <a href="#">MarkErrorAsHandled</a>

<a href="#">led</a>	の呼び出しによって「処理済み」のマークが付けられているかどうかを示す値を取得します。
---------------------	--

[Top](#)

## See Also

### Reference

[SavedChangesEventArgs Class](#)

[C1.Data Namespace](#)

### Error Property

[C1.Data Namespace](#) > [SavedChangesEventArgs Class](#) : Error Property

保存操作中に発生したエラーを示す値を取得します。

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Error As System.Exception</code>	
C#	
<code>public System.Exception Error {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SavedChangesEventArgs Class](#)

[SavedChangesEventArgs Members](#)

### HasError Property

[C1.Data Namespace](#) > [SavedChangesEventArgs Class](#) : HasError Property

保存操作が失敗したかどうかを示す値を取得します。 true の場合、[Error](#) プロパティの詳細を検査します。

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property HasError As System.Boolean</code>	
C#	
<code>public System.bool HasError {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SavedChangesEventArgs Class](#)

[SavedChangesEventArgs Members](#)

### IsErrorHandled Property

[C1.Data Namespace](#) > [SavedChangesEventArgs Class](#) : IsErrorHandled Property

エラーが [MarkErrorAsHandled](#)

の呼び出しによって「処理済み」のマークが付けられているかどうかを示す値を取得します。

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property IsErrorHandled As System.Boolean</code>	
C#	
<code>public System.bool IsErrorHandled {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference


## C1.Data.DataSource Namespace

### Overview

#### Classes

	Class	Description
	<a href="#">BaseControlHandler</a>	サポートされている型の GUI コントロールを <a href="#">C1DataSource</a> に接続するコントロールハンドラの基本クラス。 これにより、これらのコントロールに <a href="#">自動ルックアップ列</a> や <a href="#">仮想モード</a> などの追加機能を提供できます。
	<a href="#">ClientCollectionView</a>	<a href="#">ClientViewSource</a> や他の Studio for Entity Framework のデータソースが使用するコレクションビューの実装です。
	<a href="#">ClientViewSource</a>	GUI コントロールが連結できる <a href="#">C1.Data.ClientCacheBase</a> からデータを公開するデータソースオブジェクトです。 <a href="#">ClientViewSource</a> を使用すると、データを簡単に <a href="#">ロード</a> 、 <a href="#">フィルタ処理</a> 、 <a href="#">グループ化</a> 、および <a href="#">ソート</a> できます。
	<a href="#">ClientViewSourceException</a>	この例外は、 <a href="#">ClientViewSource</a> が正しく構成されていないこと、または <a href="#">Load</a> 操作中にエラーが発生したことを示します。

#### Enumerations

	Enumeration	Description
	<a href="#">VirtualModeKind</a>	<a href="#">ClientViewSource</a> の有効な仮想モードの列挙です。 <a href="#">VirtualMode</a> プロパティで使用されます。

#### See Also

#### Reference

[C1.Silverlight.Data.Entity.5 Assembly](#)

# Classes

## BaseControlHandler

[C1.Data.DataSource Namespace](#) : BaseControlHandler Class

サポートされている型の GUI コントロールを [C1.DataSource](#) に接続するコントロールハンドラの基本クラス。これにより、これらのコントロールに [自動ルックアップ列](#) や [仮想モード](#) などの追加機能を提供できます。

## Object Model

BaseControlHandler

## Syntax

Visual Basic (Declaration)

```
Public MustInherit Class BaseControlHandler
    Inherits System.Windows.DependencyObject
```

C#

```
public abstract class BaseControlHandler : System.Windows.DependencyObject
```

## Remarks

コントロールには、プラットフォーム固有のコントロールハンドラである [C1.Win.Data.ControlHandler](#)、[C1.WPF.Data.ControlHandler](#)、または [C1.Silverlight.Data.ControlHandler](#) を使用してください。

プラットフォームごとのサポートされている GUI コントロールのリストについては、各プラットフォームの [ControlHandler](#) のリファレンスを参照してください。

## Inheritance Hierarchy

System.Object

System.Windows.DependencyObject

**C1.Data.DataSource.BaseControlHandler**

[C1.Silverlight.Data.ControlHandler](#)

[C1.WPF.Data.ControlHandler](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Members](#)  
[C1.Data.DataSource Namespace](#)

## Overview

[C1.Data.DataSource Namespace](#) : BaseControlHandler Class

サポートされている型の GUI コントロールを C1DataSource に接続するコントロールハンドラの基本クラス。これにより、これらのコントロールに [自動ルックアップ列](#)や [仮想モード](#)などの追加機能を提供できます。

## Object Model

BaseControlHandler

## Syntax

Visual Basic (Declaration)

```
Public MustInherit Class BaseControlHandler
    Inherits System.Windows.DependencyObject
```

C#

```
public abstract class BaseControlHandler : System.Windows.DependencyObject
```

## Remarks

コントロールには、プラットフォーム固有のコントロールハンドラである C1.Win.Data.ControlHandler、C1.WPF.Data.ControlHandler、または C1.Silverlight.Data.ControlHandler を使用してください。

プラットフォームごとのサポートされている GUI コントロールのリストについては、各プラットフォームの ControlHandler のリファレンスを参照してください。

## Inheritance Hierarchy

System.Object  
    System.Windows.DependencyObject

## C1.Data.DataSource.BaseControlHandler

[C1.Silverlight.Data.ControlHandler](#)

[C1.WPF.Data.ControlHandler](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Members](#)

[C1.Data.DataSource Namespace](#)

## Members

[Fields](#) [Properties](#) [Methods](#)

[C1.Data.DataSource Namespace](#) : [BaseControlHandler Class](#)


The following tables list the members exposed by [BaseControlHandler](#).

## Public Fields




	Name	Description
 <b>S</b>	<a href="#">AutoLookupProperty</a>	<a href="#">AutoLookup</a> プロパティの DependencyProperty。
 <b>S</b>	<a href="#">VirtualModeProperty</a>	<a href="#">VirtualMode</a> プロパティの DependencyProperty。

[Top](#)

## Public Properties

	Name	Description
	<a href="#">AutoLookup</a>	ナビゲーション（外部キー、ルックアップ）プロパティに連結されているデータグリッド列をコンボボックス列に変換するかどうかを示す値を取得または設定します。  これにより、ユーザーは、ドロップダウンリストから値を選択して、正しい値を確認したり値を編集できます。デフォルト値は false です。



 <a href="#">Dispatcher</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">SupportsVirtualMode</a>	このコントロールハンドラが仮想モードをサポートするかどうかを示す値を取得します。
 <a href="#">VirtualMode</a>	<a href="#">ClientViewSource</a> で指定されている仮想モードをこのコントロールハンドラで管理するかどうかを示す値を取得または設定します。

[Top](#)

## Public Methods

	Name	Description
 <a href="#">Apply</a>		このコントロールハンドラの設定を現在のコントロールに強制的に適用させます。
 <a href="#">ClearValue</a>		(Inherited from System.Windows.DependencyObject)
 <a href="#">GetAnimationBaseValue</a>		(Inherited from System.Windows.DependencyObject)
 <a href="#">GetValue</a>		(Inherited from System.Windows.DependencyObject)
 <a href="#">ReadLocalValue</a>		(Inherited from System.Windows.DependencyObject)
 <a href="#">SetValue</a>		(Inherited from System.Windows.DependencyObject)

[Top](#)

## See Also

### Reference

[BaseControlHandler Class](#)

[C1.Data.DataSource Namespace](#)

## Methods

[C1.Data.DataSource Namespace](#) : [BaseControlHandler Class](#)

For a list of all members of this type, see [BaseControlHandler members](#).

# Public Methods

	Name	Description
⇒	<a href="#">Apply</a>	この <a href="#">コントロールハンドラ</a> の設定を現在のコントロールに強制的に適用させます。
⇒	<a href="#">ClearValue</a>	(Inherited from System.Windows.DependencyObject)
⇒	<a href="#">GetAnimationBaseValue</a>	(Inherited from System.Windows.DependencyObject)
⇒	<a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
⇒	<a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
⇒	<a href="#">SetValue</a>	(Inherited from System.Windows.DependencyObject)

[Top](#)

## See Also

### Reference

[BaseControlHandler Class](#)  
[C1.Data.DataSource Namespace](#)

### Apply Method

[C1.Data.DataSource Namespace](#) > [BaseControlHandler Class](#) : Apply Method

この[コントロールハンドラ](#)の設定を現在のコントロールに強制的に適用させます。

## Syntax

Visual Basic (Declaration)	
<code>Public Overridable Sub Apply()</code>	
C#	
<code>public virtual void Apply()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



[BaseControlHandler Class](#)  
[BaseControlHandler Members](#)

### Properties

[C1.Data.DataSource Namespace](#) : BaseControlHandler Class

For a list of all members of this type, see [BaseControlHandler members](#).

## Public Properties

Name	Description
 <a href="#">AutoLookup</a>	ナビゲーション（外部キー、ルックアップ）プロパティに連結されているデータグリッド列をコンボボックス列に変換するかどうかを示す値を取得または設定します。  これにより、ユーザーは、ドロップダウンリストから値を選択して、正しい値を確認したり値を編集できます。デフォルト値は false です。
 <a href="#">Dispatcher</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">SupportsVirtualMode</a>	このコントロールハンドラが仮想モードをサポートするかどうかを示す値を取得します。
 <a href="#">VirtualMode</a>	<a href="#">ClientViewSource</a>  で指定されている仮想モードをこのコントロールハンドラで管理するかどうかを示す値を取得または設定します。

[Top](#)

## See Also

### Reference

[BaseControlHandler Class](#)  
[C1.Data.DataSource Namespace](#)

## AutoLookup Property

[C1.Data.DataSource Namespace](#) > [BaseControlHandler Class](#) : AutoLookup Property

ナビゲーション（外部キー、ルックアップ）プロパティに連結されているデータグリッド列をコンボボックス列に変換するかどうかを示す値を取得または設定します。

これにより、ユーザーは、ドロップダウンリストから値を選択して、正しい値を確認したり値を編集できます。デフォルト値は false です。

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.DefaultValueAttribute(>>
<System.ComponentModel.DescriptionAttribute("Gets or sets a value indicating
whether data grid columns bound to navigation (foreign key, lookup)
properties must be converted to combo box columns.")>
Public Property AutoLookup As System.Boolean
```

C#

```
[System.ComponentModel.DefaultValue()]
[System.ComponentModel.Description("Gets or sets a value indicating whether
data grid columns bound to navigation (foreign key, lookup) properties must
be converted to combo box columns.")]
public System.bool AutoLookup {get; set;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Class](#)

[BaseControlHandler Members](#)

## SupportsVirtualMode Property

[C1.Data.DataSource Namespace](#) > [BaseControlHandler Class](#) : SupportsVirtualMode Property

この[コントロールハンドラ](#)が仮想モードをサポートするかどうかを示す値を取得します。

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.BrowsableAttribute(False)&gt; Public Overridable ReadOnly Property SupportsVirtualMode As System.Boolean</pre>	
C#	
<pre>[System.ComponentModel.Browsable(false)] public virtual System.bool SupportsVirtualMode {get;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Class](#)

[BaseControlHandler Members](#)

[VirtualMode Property](#)

### VirtualMode Property

[C1.Data.DataSource Namespace](#) > [BaseControlHandler Class](#) : VirtualMode Property

[ClientViewSource](#)

で指定されている仮想モードをこのコントロールハンドラで管理するかどうかを示す値を取得または設定します。

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DescriptionAttribute("Gets or sets a value indicating whether virtual mode specified in a ClientViewSource is managed by this control.")&gt; &lt;System.ComponentModel.DefaultValueAttribute()&gt; Public Property VirtualMode As System.Boolean</pre>	
C#	
<pre>[System.ComponentModel.Description("Gets or sets a value indicating whether virtual mode specified in a ClientViewSource is managed by this control.")] [System.ComponentModel.DefaultValue()]</pre>	

```
public System.bool VirtualMode {get; set;}
```

## Remarks

このプロパティを true に設定しても、有効となるのは、関連付けられている [ClientViewSource](#) の [VirtualMode](#) が [Managed](#) に設定されている場合だけです。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Class](#)  
[BaseControlHandler Members](#)

### Fields

[C1.Data.DataSource Namespace](#) : [BaseControlHandler Class](#)

For a list of all members of this type, see [BaseControlHandler members](#).

## Public Fields

	Name	Description
 <b>S</b>	<a href="#">AutoLookupProperty</a>	<a href="#">AutoLookup</a> プロパティの <a href="#">DependencyProperty</a> 。
 <b>S</b>	<a href="#">VirtualModeProperty</a>	<a href="#">VirtualMode</a> プロパティの <a href="#">DependencyProperty</a> 。

[Top](#)

## See Also

### Reference

[BaseControlHandler Class](#)  
[C1.Data.DataSource Namespace](#)

### AutoLookupProperty Field

[C1.Data.DataSource Namespace](#) > [BaseControlHandler Class](#) : [AutoLookupProperty Field](#)

[AutoLookup](#) プロパティの [DependencyProperty](#)。

## Syntax

Visual Basic (Declaration)	
<code>Public Shared ReadOnly AutoLookupProperty As System.Windows.DependencyProperty</code>	
C#	
<code>public static readonly System.Windows.DependencyProperty AutoLookupProperty</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[BaseControlHandler Class](#)

[BaseControlHandler Members](#)

VirtualModeProperty Field

[C1.Data.DataSource Namespace](#) > [BaseControlHandler Class](#) : VirtualModeProperty Field

[VirtualMode](#) プロパティの DependencyProperty。

## Syntax

Visual Basic (Declaration)	
<code>Public Shared ReadOnly VirtualModeProperty As System.Windows.DependencyProperty</code>	
C#	
<code>public static readonly System.Windows.DependencyProperty VirtualModeProperty</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[BaseControlHandler Class](#)

[BaseControlHandler Members](#)

# ClientCollectionView

[C1.Data.DataSource Namespace](#) : ClientCollectionView Class

[ClientViewSource](#) や他の Studio for Entity Framework のデータソースが使用するコレクションビューの実装です。

## Object Model

ClientCollectionView

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>  
Public Class ClientCollectionView
```

C#

```
[System.Reflection.DefaultMember("Item")]  
public class ClientCollectionView
```

## Inheritance Hierarchy

System.Object

**C1.Data.DataSource.ClientCollectionView**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Members](#)

[C1.Data.DataSource Namespace](#)



## Overview

[C1.Data.DataSource Namespace](#) : ClientCollectionView Class

[ClientViewSource](#) や他の Studio for Entity Framework のデータソースが使用するコレクションビューの実装です。

## Object Model

ClientCollectionView

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>  
Public Class ClientCollectionView
```

C#

```
[System.Reflection.DefaultMember("Item")]  
public class ClientCollectionView
```

## Inheritance Hierarchy

System.Object

**C1.Data.DataSource.ClientCollectionView**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Members](#)

[C1.Data.DataSource Namespace](#)

## Members

[Properties](#) [Methods](#) [Events](#)

[C1.Data.DataSource Namespace](#) : ClientCollectionView Class

The following tables list the members exposed by [ClientCollectionView](#).







## Public Properties

	Name	Description
	<a href="#">CanAdd</a>	<a href="#">Add</a> メソッドがサポートされているかどうかを示す値を取得します。
	<a href="#">CanChangePage</a>	<a href="#">PageIndex</a> 値を変更できるかどうかを示す値を取得します。
	<a href="#">CanRemove</a>	項目をコレクションから削除できるかどうかを示す値を取得します。 。
	<a href="#">CollectionViewFactory</a>	<b>System.Windows.Data.CollectionViewSource</b> のソースとして使用できる <b>System.ComponentModel.ICollectionViewFactory</b> のインスタンスを取得します。
	<a href="#">Count</a>	<a href="#">ClientCollectionView</a> に含まれる要素の数を取得します。
	<a href="#">CurrentItem</a>	ビュー内の現在の項目を取得します。
	<a href="#">CurrentPosition</a>	コレクションビュー内の <a href="#">CurrentItem</a> の順序位置を取得します。
	<a href="#">IsEmpty</a>	結果のビューが空かどうかを示す値を返します。
	<a href="#">IsPageChanging</a>	ページインデックスが変更されているかどうかを示す値を取得します。
	<a href="#">Item</a>	指定された <a href="#">index</a> にある要素を取得します。
	<a href="#">PageCount</a>	このビューのページ数を取得します。
	<a href="#">PageIndex</a>	現在のページの 0 から始まるインデックスを取得します。
	<a href="#">PageSize</a>	1 ページに表示する項目数を取得または設定します。
	<a href="#">TotalItemCount</a>	ページング適用前のビュー内の項目の合計数を取得します。

[Top](#)

## Public Methods

	Name	Description
⇒	<b>Add</b>	新しい <b>entity</b> をクライアント側のキャッシュおよび関連付けられているコンテキストに追加します。 <b>entity</b> は、基底のクエリーと一致する場合、この <b>コレクションビュー</b> に表示されます。
⇒	<b>AsLive&lt;T&gt;</b>	この <b>ClientCollectionView</b> を <b>ライブビュー</b> に変換します。
⇒	<b>Contains</b>	指定された項目がこのコレクションビューに属するかどうかを示す値を返します。
⇒	<b>IndexOf</b>	<b>ClientCollectionView</b> 内の特定の <b>item</b> のインデックスを判定します。
⇒	<b>MoveCurrentTo</b>	指定された項目をビュー内の <b>CurrentItem</b> に設定します。
⇒	<b>MoveCurrentToFirst</b>	ビュー内の最初の項目を <b>CurrentItem</b> として設定します。
⇒	<b>MoveCurrentToLast</b>	ビュー内の最後の項目を <b>CurrentItem</b> として設定します。
⇒	<b>MoveCurrentToNext</b>	ビュー内の <b>CurrentItem</b> の後の項目を <b>System.ComponentModel.ICollectionView.CurrentItem</b> として設定します。
⇒	<b>MoveCurrentToPosition</b>	指定されたインデックスにある項目をビュー内の <b>CurrentItem</b> として設定します。
⇒	<b>MoveCurrentToPrevious</b>	ビュー内の <b>CurrentItem</b> の前の項目を <b>CurrentItem</b> として設定します。
⇒	<b>MoveToFirstPage</b>	最初のページを現在のページとして設定します。

⇒  <a href="#">MoveToLastPage</a>	最後のページを現在のページとして設定します。
⇒  <a href="#">MoveToNextPage</a>	現在のページの後のページに移動します。
⇒  <a href="#">MoveToPage</a>	最初のページを現在のページとして設定します。
⇒  <a href="#">MoveToPreviousPage</a>	現在のページの前のページに移動します。
⇒  <a href="#">Remove</a>	指定された項目をコレクションから削除します。
⇒  <a href="#">RemoveAt</a>	コレクションから、指定された位置にある項目を削除します。

[Top](#)

## Public Events

	Name	Description
	<a href="#">CurrentChanged</a>	このインタフェースを実装する場合は、現在の項目が変更された後でこのイベントを発生させます。
	<a href="#">CurrentChanging</a>	このインタフェースを実装する場合は、現在の項目を変更する前にこのイベントを発生させます。イベントハンドラは、このイベントをキャンセルできます。
	<a href="#">PageChanged</a>	<a href="#">PageIndex</a> が変更された後で発生します。
	<a href="#">PageChanging</a>	<a href="#">PageIndex</a> を変更する前に発生します。
	<a href="#">PropertyChanged</a>	プロパティ値が変更されたときに発生します。

[Top](#)

## See Also

### Reference

## Methods

C1.Data.DataSource Namespace : ClientCollectionView Class

For a list of all members of this type, see [ClientCollectionView members](#).

## Public Methods

Name	Description
 <a href="#">Add</a>	新しい <a href="#">entity</a> をクライアント側のキャッシュおよび関連付けられているコンテキストに追加します。 <a href="#">entity</a> は、基底のクエリーと一致する場合、この <a href="#">コレクションビュー</a> に表示されます。
 <a href="#">AsLive&lt;T&gt;</a>	この <a href="#">ClientCollectionView</a> を <a href="#">ライブビュー</a> に変換します。
 <a href="#">Contains</a>	指定された項目がこのコレクションビューに属するかどうかを示す値を返します。
 <a href="#">IndexOf</a>	<a href="#">ClientCollectionView</a> 内の特定の <a href="#">item</a> のインデックスを判定します。
 <a href="#">MoveCurrentTo</a>	指定された項目をビュー内の <a href="#">CurrentItem</a> に設定します。
 <a href="#">MoveCurrentToFirst</a>	ビュー内の最初の項目を <a href="#">CurrentItem</a> として設定します。
 <a href="#">MoveCurrentToLast</a>	ビュー内の最後の項目を <a href="#">CurrentItem</a> として設定します。
 <a href="#">MoveCurrentToNext</a>	ビュー内の <a href="#">CurrentItem</a> の後の項目を <b>System.ComponentModel.ICollectionView.CurrentItem</b> として設定します。
 <a href="#">MoveCurrentToPosition</a>	指定されたインデックスにある項目をビュー内の <a href="#">CurrentItem</a> として設定します。

 <a href="#">MoveCurrentToPrevious</a>	ビュー内の <a href="#">CurrentItem</a> の前の項目を <a href="#">CurrentItem</a> として設定します。
 <a href="#">MoveToFirstPage</a>	最初のページを現在のページとして設定します。
 <a href="#">MoveToLastPage</a>	最後のページを現在のページとして設定します。
 <a href="#">MoveToNextPage</a>	現在のページの後のページに移動します。
 <a href="#">MoveToPage</a>	最初のページを現在のページとして設定します。
 <a href="#">MoveToPreviousPage</a>	現在のページの前のページに移動します。
 <a href="#">Remove</a>	指定された項目をコレクションから削除します。
 <a href="#">RemoveAt</a>	コレクションから、指定された位置にある項目を削除します。

[Top](#)

# See Also

## Reference

[ClientCollectionView Class](#)  
[C1.Data.DataSource Namespace](#)

## Add Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : Add Method

追加する新しいエンティティ。

新しい [entity](#)

をクライアント側のキャッシュおよび関連付けられているコンテキストに追加します。 [entity](#) は、基底のクエリーと一致する場合、この [コレクションビュー](#) に表示されます。

# Syntax

Visual Basic (Declaration)

```
Public Sub Add( _  
    ByVal entity As System.Object _  
)
```

C#	
<pre>public void Add(     System.object entity )</pre>	

## Parameters

*entity*

追加する新しいエンティティ。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## AsLive<T> Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : AsLive<T> Method

このコレクションビュー内の要素の型。

この [ClientCollectionView](#) を [ライブビュー](#) に変換します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Function AsLive(Of T)() As View(Of T)</pre>	
C#	
<pre>public View&lt;T&gt; AsLive&lt;T&gt;()</pre>	

## Type Parameters

*T*

このコレクションビュー内の要素の型。

## Return Value

結果の[ライブビュー](#)。

## Exceptions

Exception	Description
<b>System.NotSupportedException</b>	<a href="#">ClientViewSource</a> が <a href="#">仮想モード</a> です。

## Remarks

このメソッドは [ClientCollectionView](#) を一切変更せず、[ライブビュー機能](#)の公開のみを行います。

このメソッドは、[仮想モード](#)の [ClientViewSource](#) ではサポートされません。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

### Contains Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : Contains Method

チェックするオブジェクト。

指定された項目がこのコレクションビューに属するかどうかを示す値を返します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Function Contains( _     ByVal item As System.Object _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool Contains(</pre>	



```
System.object item
)
```

## Parameters

*item*

チェックするオブジェクト。

## Return Value

項目がこのコレクションビューに属する場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## IndexOf Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : IndexOf Method

[ClientCollectionView](#) で見つける項目。

[ClientCollectionView](#) 内の特定の *item* のインデックスを判定します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Function IndexOf( _     ByVal item As System.Object _ ) As System.Integer</pre>	
C#	
<pre>public System.int IndexOf(     System.object item )</pre>	

## Parameters

*item*

[ClientCollectionView](#) で見つける項目。

## Return Value

リスト内で見つかった場合は [item](#) のインデックス、そうでない場合は -1。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## MoveCurrentTo Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveCurrentTo Method

[CurrentItem](#) として設定する項目。

指定された項目をビュー内の [CurrentItem](#) に設定します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Function MoveCurrentTo( _     ByVal item As System.Object _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool MoveCurrentTo(     System.object item )</pre>	

## Parameters

*item*

[CurrentItem](#) として設定する項目。

## Return Value

結果の [CurrentItem](#) がビュー内である場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveCurrentToFirst Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveCurrentToFirst Method

ビュー内の最初の項目を [CurrentItem](#) として設定します。

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> MoveCurrentToFirst() <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool MoveCurrentToFirst()	

### Return Value

結果の [CurrentItem](#) がビュー内の項目である場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## MoveCurrentToLast Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveCurrentToLast Method

ビュー内の最後の項目を [CurrentItem](#) として設定します。

## Syntax

Visual Basic (Declaration)

```
Public Function MoveCurrentToLast() As System.Boolean
```

C#

```
public System.bool MoveCurrentToLast()
```

### Return Value

結果の [CurrentItem](#) がビュー内の項目である場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## MoveCurrentToNext Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveCurrentToNext Method

ビュー内の [CurrentItem](#) の後の項目を

**System.ComponentModel.ICollectionView.CurrentItem** として設定します。

## Syntax

Visual Basic (Declaration)

```
Public Function MoveCurrentToNext() As System.Boolean
```

C#

```
public System.bool MoveCurrentToNext()
```

## Return Value

結果の [CurrentItem](#) がビュー内の項目である場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveCurrentToPosition Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveCurrentToPosition Method

**System.ComponentModel.ICollectionView.CurrentItem** として設定するインデックス。

指定されたインデックスにある項目をビュー内の [CurrentItem](#) として設定します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Function MoveCurrentToPosition( _     ByVal position As System.Integer _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool MoveCurrentToPosition(     System.int position )</pre>	

### Parameters

*position*

**System.ComponentModel.ICollectionView.CurrentItem**  
として設定するインデックス。

### Return Value

結果の [CurrentItem](#) がビュー内の項目である場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveCurrentToPrevious Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveCurrentToPrevious Method

ビュー内の [CurrentItem](#) の前の項目を [CurrentItem](#) として設定します。

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> MoveCurrentToPrevious() <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool MoveCurrentToPrevious()	

### Return Value

結果の [CurrentItem](#) がビュー内の項目である場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveToFirstPage Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveToFirstPage Method

最初のページを現在のページとして設定します。

## Syntax

Visual Basic (Declaration)

```
Public Function MoveToFirstPage() As System.Boolean
```

C#

```
public System.bool MoveToFirstPage()
```

### Return Value

操作が成功した場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveToLastPage Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveToLastPage Method

最後のページを現在のページとして設定します。

## Syntax

Visual Basic (Declaration)

```
Public Function MoveToLastPage() As System.Boolean
```

C#

```
public System.bool MoveToLastPage()
```

### Return Value

操作が成功した場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveToNextPage Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveToNextPage Method

現在のページの後のページに移動します。

## Syntax

Visual Basic (Declaration)

```
Public Function MoveToNextPage() As System.Boolean
```

C#

```
public System.bool MoveToNextPage()
```

### Return Value

操作が成功した場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveToPage Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveToPage Method



移動先のページのインデックス。

最初のページを現在のページとして設定します。

## Syntax

Visual Basic (Declaration)

```
Public Function MoveToPage( _  
    ByVal pageIndex As System.Integer _  
) As System.Boolean
```

C#

```
public System.bool MoveToPage(  
    System.int pageIndex  
)
```

### Parameters

*pageIndex*

移動先のページのインデックス。

### Return Value

操作が成功した場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### MoveToPreviousPage Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : MoveToPreviousPage Method

現在のページの前のページに移動します。

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> MoveToPreviousPage() <b>As</b> System.Boolean	
C#	
<b>public</b> System. <b>bool</b> MoveToPreviousPage()	

### Return Value

操作が成功した場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### Remove Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : Remove Method

削除する項目。

指定された項目をコレクションから削除します。

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Remove( _ <b>ByVal</b> item <b>As</b> System.Object _ )	
C#	
<b>public void</b> Remove( System. <b>object</b> item )	

### Parameters

*item*

削除する項目。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

### RemoveAt Method

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : RemoveAt Method

削除する項目の位置。

コレクションから、指定された位置にある項目を削除します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Sub RemoveAt( _     ByVal index As System.Integer _ )</pre>	
C#	
<pre>public void RemoveAt(     System.int index )</pre>	

### Parameters

*index*

削除する項目の位置。

## Exceptions

Exception	Description
<b>System.ArgumentOutOfRangeException</b>	index

	が0未満であるか、コレクションビュー内のアイテム数を超えて
--	-------------------------------

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference







[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

## Properties

[C1.Data.DataSource Namespace](#) : [ClientCollectionView Class](#)

For a list of all members of this type, see [ClientCollectionView members](#).

## Public Properties

	Name	Description
	<a href="#">CanAdd</a>	<a href="#">Add</a> メソッドがサポートされているかどうかを示す値を取得します。
	<a href="#">CanChangePage</a>	<a href="#">PageIndex</a> 値を変更できるかどうかを示す値を取得します。
	<a href="#">CanRemove</a>	項目をコレクションから削除できるかどうかを示す値を取得します。 。
	<a href="#">CollectionViewFactory</a>	<b>System.Windows.Data.CollectionViewSource</b> のソースとして使用できる <b>System.ComponentModel.ICollectionViewFactory</b> のインスタンスを取得します。
	<a href="#">Count</a>	<a href="#">ClientCollectionView</a> に含まれる要素の数を取得します。
	<a href="#">CurrentItem</a>	ビュー内の現在の項目を取得します。

	<a href="#">CurrentPosition</a>	コレクションビュー内の <a href="#">CurrentItem</a> の順序位置を取得します。
	<a href="#">IsEmpty</a>	結果のビューが空かどうかを示す値を返します。
	<a href="#">IsPageChanging</a>	ページインデックスが変更されているかどうかを示す値を取得します。
	<a href="#">Item</a>	指定された <a href="#">index</a> にある要素を取得します。
	<a href="#">PageCount</a>	このビューのページ数を取得します。
	<a href="#">PageIndex</a>	現在のページの 0 から始まるインデックスを取得します。
	<a href="#">PageSize</a>	1 ページに表示する項目数を取得または設定します。
	<a href="#">TotalItemCount</a>	ページング適用前のビュー内の項目の合計数を取得します。

[Top](#)

## See Also

### Reference

[ClientCollectionView Class](#)  
[C1.Data.DataSource Namespace](#)

### CanAdd Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CanAdd Property

[Add](#) メソッドがサポートされているかどうかを示す値を取得します。

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> CanAdd <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool CanAdd { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

### CanChangePage Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CanChangePage Property

[PageIndex](#) 値を変更できるかどうかを示す値を取得します。

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property CanChangePage As System.Boolean</code>	
C#	
<code>public System.bool CanChangePage {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

### CanRemove Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CanRemove Property

項目をコレクションから削除できるかどうかを示す値を取得します。

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<code>Public ReadOnly Property CanRemove As System.Boolean</code>
C#
<code>public System.bool CanRemove {get;}</code>

### Property Value

項目をコレクションから削除できる場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### CollectionViewFactory Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CollectionViewFactory Property

**System.Windows.Data.CollectionViewSource** のソースとして使用できる **System.ComponentModel.ICollectionViewFactory** のインスタンスを取得します。

## Syntax

Visual Basic (Declaration)
<code>Public ReadOnly Property CollectionViewFactory As System.ComponentModel.ICollectionViewFactory</code>
C#
<code>public System.ComponentModel.ICollectionViewFactory CollectionViewFactory {get;}</code>

### Property Value

**System.ComponentModel.ICollectionView** と同じ [ClientCollectionView](#) を返すファクトリ。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

### Count Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : Count Property

[ClientCollectionView](#) に含まれる要素の数を取得します。

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Count As System.Integer</code>	
C#	
<code>public System.int Count {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

### CurrentItem Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CurrentItem Property

ビュー内の現在の項目を取得します。

## Syntax

Visual Basic (Declaration)	
----------------------------	--



Public ReadOnly Property CurrentItem As System.Object
---

C#
----

public System.object CurrentItem {get;}
---

### Property Value

ビューの現在の項目。現在の項目がない場合は null。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### CurrentPosition Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CurrentPosition Property

コレクションビュー内の [CurrentItem](#) の順序位置を取得します。

## Syntax

Visual Basic (Declaration)
----------------------------

Public ReadOnly Property CurrentPosition As System.Integer
--

C#
----

public System.int CurrentPosition {get;}
--

### Property Value

コレクションビュー内の [CurrentItem](#) の順序位置。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### IsEmpty Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : IsEmpty Property

結果のビューが空かどうかを示す値を返します。

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property IsEmpty As System.Boolean</code>	
C#	
<code>public System.bool IsEmpty {get;}</code>	

### Property Value

結果のビューが空である場合は true、そうでない場合は false。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### IsPageChanging Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : IsPageChanging Property

ページインデックスが変更されているかどうかを示す値を取得します。

## Syntax

Visual Basic (Declaration)	
----------------------------	--

```
Public ReadOnly Property IsPageChanging As System.Boolean
```

```
C#
```

```
public System.bool IsPageChanging {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### Item Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : Item Property

取得する要素の 0 から始まるインデックス。

指定された [index](#) にある要素を取得します。

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Default Property Item( _  
    ByVal index As System.Integer _  
) As System.Object
```

```
C#
```

```
public System.object this[  
    System.int index  
]; {get;}
```

### Parameters

*index*

取得する要素の 0 から始まるインデックス。

### Property Value

指定されたインデックスにある要素。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

### PageCount Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : PageCount Property

このビューのページ数を取得します。

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> PageCount <b>As</b> System.Integer	
C#	
<b>public</b> System.int PageCount { <b>get</b> ;}	

## Remarks

[PageSize](#) が 0 の場合は、[PageIndex](#) も 0 になります。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## PageIndex Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : PageIndex Property

現在のページの 0 から始まるインデックスを取得します。

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property PageIndex As System.Integer</code>	
C#	
<code>public System.int PageIndex {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

## PageSize Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : PageSize Property

1 ページに表示する項目数を取得または設定します。

## Syntax

Visual Basic (Declaration)	
<code>Public Property PageSize As System.Integer</code>	
C#	
<code>public System.int PageSize {get; set;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

## TotalItemCount Property

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : TotalItemCount Property

ページング適用前のビュー内の項目の合計数を取得します。

# Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property TotalItemCount As System.Integer</code>	
C#	
<code>public System.int TotalItemCount {get;}</code>	

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2




# See Also

## Reference


[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)


## Events

>

Name	Description
 <a href="#">CurrentC</a> <a href="#">hanged</a>	このインタフェースを実装する場合は、現在の項目が変更された後でこのイベントを発生させます。
 <a href="#">CurrentC</a> <a href="#">hanging</a>	このインタフェースを実装する場合は、現在の項目を変更する前にこのイベントを発生させます。イベントハンドラは、このイベントをキャンセルできます。
 <a href="#">PageCha</a>	<a href="#">PageIndex</a> が変更された後で発生します。

nged

 **PageChanging** [PageIndex](#) を変更する前に発生します。

 **PropertyChanged** プロパティ値が変更されたときに発生します。

[Top](#)

## See Also

### Reference

[ClientCollectionView Class](#)

[C1.Data.DataSource Namespace](#)

### CurrentChanged Event

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CurrentChanged Event

このインタフェースを実装する場合は、現在の項目が変更された後でこのイベントを発生させます。

## Syntax

Visual Basic (Declaration)

```
Public Event CurrentChanged As System.EventHandler
```

C#

```
public event System.EventHandler CurrentChanged
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## CurrentChanging Event

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : CurrentChanging Event

このインタフェースを実装する場合は、現在の項目を変更する前にこのイベントを発生させます。イベントハンドラは、このイベントをキャンセルできます。

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> CurrentChanging <b>As</b> System.ComponentModel.CurrentChangingEventHandler	
C#	
<b>public event</b> System.ComponentModel.CurrentChangingEventHandler CurrentChanging	

## Event Data

The event handler receives an argument of type `System.ComponentModel.CurrentChangingEventArgs` containing data related to this event. The following **CurrentChangingEventArgs** properties provide information specific to this event.

Property	Description
<b>Cancel</b>	Gets or sets a value that indicates whether to cancel the event.
<b>IsCancelable</b>	Gets a value that indicates whether the event is cancelable.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

## PageChanged Event

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : PageChanged Event



[PageIndex](#) が変更された後で発生します。

## Syntax

Visual Basic (Declaration)	
<code>Public Event PageChanged As System.EventHandler(Of EventArgs)</code>	
C#	
<code>public event System.EventHandler&lt;EventArgs&gt; PageChanged</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

### PageChanging Event

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : PageChanging Event

[PageIndex](#) を変更する前に発生します。

## Syntax

Visual Basic (Declaration)	
<code>Public Event PageChanging As System.EventHandler(Of PageChangingEventArgs)</code>	
C#	
<code>public event System.EventHandler&lt;PageChangingEventArgs&gt; PageChanging</code>	

## Event Data

The event handler receives an argument of type `System.ComponentModel.PageChangingEventArgs` containing data related to this event. The following **PageChangingEventArgs** properties provide information specific to this event.

Property	Description
----------	-------------

<b>Cancel</b>	(Inherited from System.ComponentModel.CancelEventArgs)
<b>NewPageIndex</b>	

## Remarks

このイベントを使用すると、**System.ComponentModel.CancelEventArgs.Cancel** を true に設定することで、進行中の **PageIndex** の変更をキャンセルできます。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)  
[ClientCollectionView Members](#)

### PropertyChanged Event

[C1.Data.DataSource Namespace](#) > [ClientCollectionView Class](#) : PropertyChanged Event

プロパティ値が変更されたときに発生します。

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> PropertyChanged <b>As</b> System.ComponentModel.PropertyChangedEventHandler	
C#	
<b>public event</b> System.ComponentModel.PropertyChangedEventHandler PropertyChanged	

## Event Data

The event handler receives an argument of type **System.ComponentModel.PropertyChangedEventArgs** containing data related to this event. The following **PropertyChangedEventArgs** properties provide information specific to this event.

Property	Description
<b>PropertyName</b>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientCollectionView Class](#)

[ClientCollectionView Members](#)

## ClientViewSource

[C1.Data.DataSource Namespace](#) : ClientViewSource Class

GUI コントロールが連結できる [C1.Data.ClientCacheBase](#) からデータを公開するデータソースオブジェクトです。 [ClientViewSource](#) を使用すると、データを簡単にロード、フィルタ処理、グループ化、およびソート できます。

## Object Model

ClientViewSource

## Syntax

Visual Basic (Declaration)	
<pre>Public Class ClientViewSource     Inherits System.Windows.DependencyObject</pre>	
C#	
<pre>public class ClientViewSource : System.Windows.DependencyObject</pre>	

## Inheritance Hierarchy

```
System.Object
    System.Windows.DependencyObject
        C1.Data.DataSource.ClientViewSource
```

[C1.Data.Entities.EntityViewSource](#)  
[C1.Silverlight.Data.RiaServices.RiaViewSource](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Members](#)  
[C1.Data.DataSource Namespace](#)

### Overview

[C1.Data.DataSource Namespace](#) : ClientViewSource Class

GUI コントロールが連結できる [C1.Data.ClientCacheBase](#) からデータを公開するデータソースオブジェクトです。 [ClientViewSource](#) を使用すると、データを簡単にロード、フィルタ処理、グループ化、およびソート できます。

## Object Model

ClientViewSource

## Syntax

Visual Basic (Declaration)	
<pre>Public Class ClientViewSource     Inherits System.Windows.DependencyObject</pre>	
C#	
<pre>public class ClientViewSource : System.Windows.DependencyObject</pre>	

## Inheritance Hierarchy

System.Object  
System.Windows.DependencyObject  
**C1.Data.DataSource.ClientViewSource**  
[C1.Data.Entities.EntityViewSource](#)  
[C1.Silverlight.Data.RiaServices.RiaViewSource](#)

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ClientViewSource Members](#)  
[C1.Data.DataSource Namespace](#)

## Members

[Properties](#) [Methods](#) [Events](#)

[C1.Data.DataSource Namespace](#) : [ClientViewSource Class](#)




The following tables list the members exposed by [ClientViewSource](#).

## Public Constructors

	Name	Description
	<a href="#">ClientViewSource Constructor</a>	<a href="#">ClientViewSource</a> クラスの新しいインスタンスを初期化します。

[Top](#)

## Public Properties



	Name	Description
	<a href="#">AutoLoad</a>	起動時や、 <a href="#">ClientViewSource</a> によって作成されたクエリーが影響を受ける変更が発生したときに、 <a href="#">Load</a> が自動的に呼び出されるかどうかを示す値を取得または設定します。 デフォルトは True です。
	<a href="#">BaseView</a>	<a href="#">ClientViewSource</a> がクエリーを作成するための基礎として使用する <a href="#">C1.Data.ClientView&lt;T&gt;</a> のインスタンスを取得または設定します。
	<a href="#">CacheTime</a>	仮想モードでロードされたエンティティが、必要かどうかのチェックがされることなくキャッシュに維持される時間を取得または設定します。

out	<p><a href="#">CacheTimeout</a></p> <p>よりも長い時間、エンティティが使用されることも必要と判断されることもなかった場合、<a href="#">ClientViewSource</a> はそのエンティティをキャッシュから削除できます。</p>
 <a href="#">CurrentClientView</a>	エンティティのロードに使用される現在の <a href="#">クライアントビュー</a> 、または <a href="#">仮想モード</a> では null を取得します。
 <a href="#">DataView</a>	最後のロード操作の結果であるエンティティの現在のビューを取得します。
 <a href="#">Dispatcher</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">FilterDescriptors</a>	ロード実行時に使用される <a href="#">FilterDescriptor</a> オブジェクトのコレクションを取得します。
 <a href="#">FilterOperator</a>	フィルタコレクション内の <a href="#">FilterDescriptors</a> どうしを結合するために使用される論理演算子を取得または設定します。デフォルト値は、 <a href="#">FilterDescriptorLogicalOperator.And</a> です。
 <a href="#">GroupDescriptors</a>	読み込んだエンティティをいくつかのグループに組織化するために使用される <a href="#">GroupDescriptor</a> オブジェクトのコレクションを取得します。
 <a href="#">IsLoadingData</a>	<a href="#">ClientViewSource</a> が現在データをロード中かどうかを示す値を取得します。
 <a href="#">LoadCommand</a>	この <a href="#">ClientViewSource</a> で <a href="#">Load</a> を呼び出す <b>System.Windows.Input.ICommand</b> を取得します。
 <a href="#">LoadDelay</a>	自動データロード操作が開始されるまでの遅延時間を取得または設定します。自動ロードを促す変更が発生した時点から 結果としての <a href="#">Load</a> が開始される時点までが遅延時間です。デフォルトの遅延時間は、25 ミリ秒です。
 <a href="#">LoadSize</a>	<a href="#">Load</a> が実行されるたびにロードされる項目の最大数を取得または設定します。0 の場合は、要求されたエンティティがすべてロードされます。デフォルトは0です。

 <b>MoveToFirstOnLoad</b>	<b>Load</b> 操作完了後の最初の項目を現在の項目にすることを示す値を取得または設定します（現在の項目が他の方法で設定されていない場合）。
 <b>Name</b>	C1DataSource.ViewSources コレクション内で参照される、この <b>ClientViewSource</b> の名前を取得します。デフォルトでは、これは EntitySetName（Entity Framework の場合）または QueryName（RIA サービスの場合）によって決定されますが、 <b>NameOverride</b> でオーバーライドできます。
 <b>NameOverride</b>	<b>Name</b> プロパティの値をオーバーライドする値を取得または設定します。
 <b>PageSize</b>	<b>DataView</b> の各ページに表示される項目数、または <b>仮想モード</b> 時に各クエリーで取得する項目数を取得または設定します。0 はページングが無効であることを示します。
 <b>SortDescriptors</b>	データのソートに使用される <b>SortDescriptor</b> オブジェクトのコレクションを取得します。
 <b>VirtualMode</b>	<b>ClientViewSource</b> が仮想モードかどうかを示す値を取得または設定します。仮想モードは、遅延やパフォーマンスの低下を招くことなく、またページングの手間をかける必要もなく、GUI コントロールを大規模なデータセットに直接連結できる革新的な技術です。デフォルトでは、仮想モードは無効です（デフォルト値は <b>VirtualModeKind.None</b> ）。

[Top](#)



## Public Methods

Name	Description
 <b>ClearValue</b>	(Inherited from System.Windows.DependencyObject)
 <b>DeferLoad</b>	ロードに関する複数のプロパティに対する変更をグループ化して、結果として行われるロード操作を遅延させるために使用されます。これによってロード操作は最後に、つまり、このメソッドから返されるオブ

		ジェクトが破棄されるときに、1 回だけ実行されます。
⇒	 <a href="#">GetAnimationBaseValue</a>	(Inherited from System.Windows.DependencyObject)
⇒	 <a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
⇒	 <a href="#">Load</a>	ロード操作を開始します。保留中のすべてのロードは、暗黙的にキャンセルされます。
⇒	 <a href="#">LoadRange</a>	<a href="#">仮想モード</a> の場合に、エンティティの特定範囲をロードします。
⇒	 <a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
⇒	 <a href="#">Refresh</a>	クライアント側のキャッシュを無視して、ロード操作を開始します。保留中のすべてのロードは、暗黙的にキャンセルされます。
⇒	 <a href="#">SetValue</a>	(Inherited from System.Windows.DependencyObject)

[Top](#)

## Public Events

	Name	Description
	<a href="#">LoadedData</a>	ロード操作が完了したとき、またはロード操作中に例外が生成されたときに発生します。
	<a href="#">PropertyChanged</a>	プロパティ値が変更されたときに発生します。

[Top](#)

## See Also

### Reference

[ClientViewSource Class](#)

[C1.Data.DataSource Namespace](#)



# ClientViewSource Constructor

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : ClientViewSource Constructor

[ClientViewSource](#) クラスの新しいインスタンスを初期化します。

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public ClientViewSource()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)  
[ClientViewSource Members](#)




## Methods

[C1.Data.DataSource Namespace](#) : [ClientViewSource Class](#)

For a list of all members of this type, see [ClientViewSource members](#).

## Public Methods

	Name	Description
⇒	<a href="#">ClearValue</a>	(Inherited from System.Windows.DependencyObject)
⇒	<a href="#">DeferLoad</a>	ロードに関係する複数のプロパティに対する変更をグループ化して、結果として行われるロード操作を遅延させるために使用されます。 これによってロード操作は最後に、つまり、このメソッドから返されるオブジェクトが破棄されるときに、1 回だけ実行されます。

 <a href="#">GetAnimationBaseValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">Load</a>	ロード操作を開始します。保留中のすべてのロードは、暗黙的にキャンセルされます。
 <a href="#">LoadRange</a>	<a href="#">仮想モード</a> の場合に、エンティティの特定範囲をロードします。
 <a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">Refresh</a>	クライアント側のキャッシュを無視して、ロード操作を開始します。保留中のすべてのロードは、暗黙的にキャンセルされます。
 <a href="#">SetValue</a>	(Inherited from System.Windows.DependencyObject)

[Top](#)

## See Also

### Reference

[ClientViewSource Class](#)

[C1.Data.DataSource Namespace](#)

### DeferLoad Method

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : DeferLoad Method

ロードに関する複数のプロパティに対する変更をグループ化して、結果として行われるロード操作を遅延させるために使用されます。

これによってロード操作は最後に、つまり、このメソッドから返されるオブジェクトが破棄されるときに、1回だけ実行されます。

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> DeferLoad() <b>As</b> System.IDisposable	
C#	

```
public System.IDisposable DeferLoad()
```

### Return Value

**System.IDisposable.Dispose** メソッドを使用して破棄されるときに **Load** 操作をトリガする **System.IDisposable** オブジェクト。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### Load Method

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : Load() Method

ロード操作を開始します。保留中のすべてのロードは、暗黙的にキャンセルされます。

## Syntax

Visual Basic (Declaration)	
<pre>Public Sub Load()</pre>	
C#	
<pre>public void Load()</pre>	

## Remarks

エンティティが既にキャッシュにある場合は、サーバーへのラウンドトリップは行われません。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

## LoadRange Method

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : LoadRange Method

ロードする最初の項目のインデックス。

ロードするエンティティの数。

仮想モードの場合に、エンティティの特定範囲をロードします。

## Syntax

Visual Basic (Declaration)	
<pre>Public Sub LoadRange( _     ByVal start As System.Integer, _     ByVal length As System.Integer _ )</pre>	
C#	
<pre>public void LoadRange(     System.int start,     System.int length )</pre>	

## Parameters

*start*

ロードする最初の項目のインデックス。

*length*

ロードするエンティティの数。

## Exceptions

Exception	Description
<b>System.InvalidOperationException</b>	<a href="#">VirtualMode</a> が <a href="#">VirtualModeKind.None</a> です。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)  
[ClientViewSource Members](#)

### Refresh Method

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : Refresh() Method

クライアント側のキャッシュを無視して、ロード操作を開始します。保留中のすべてのロードは、暗黙的にキャンセルされます。

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Refresh()	
C#	
<b>public void</b> Refresh()	

## Remarks

サーバー上で発生した可能性があるすべての変更によってデータを更新するには、このメソッドを使用します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)  
[ClientViewSource Members](#)

### Properties

[C1.Data.DataSource Namespace](#) : ClientViewSource Class

For a list of all members of this type, see [ClientViewSource members](#).

## Public Properties

Name	Description
 <a href="#">AutoLoad</a>	起動時や、 <a href="#">ClientViewSource</a> によって作成されたクエリーが影響を受ける変更が発生したときに、 <a href="#">Load</a> が自動的に呼び出されるかどうかを示す値を取得または設定します。デフォルトは True です。
 <a href="#">BaseView</a>	<a href="#">ClientViewSource</a> がクエリーを作成するための基礎として使用する <a href="#">C1.Data.ClientView&lt;T&gt;</a> のインスタンスを取得または設定します。
 <a href="#">CacheTimeout</a>	仮想モードでロードされたエンティティが、必要かどうかのチェックがされることなくキャッシュに維持される時間を取得または設定します。 <a href="#">CacheTimeout</a> よりも長い時間、エンティティが使用されることも必要と判断されることもなかった場合、 <a href="#">ClientViewSource</a> はそのエンティティをキャッシュから削除できます。
 <a href="#">CurrentClientView</a>	エンティティのロードに使用される現在の <a href="#">クライアントビュー</a> 、または <a href="#">仮想モード</a> では null を取得します。
 <a href="#">DataView</a>	最後のロード操作の結果であるエンティティの現在のビューを取得します。
 <a href="#">Dispatcher</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">FilterDescriptors</a>	ロード実行時に使用される <a href="#">FilterDescriptor</a> オブジェクトのコレクションを取得します。
 <a href="#">FilterOperator</a>	フィルタコレクション内の <a href="#">FilterDescriptors</a> どうしを結合するために使用される論理演算子を取得または設定します。デフォルト値は、 <a href="#">FilterDescriptorLogicalOperator.And</a> です。
 <a href="#">GroupDescriptors</a>	読み込んだエンティティをいくつかのグループに組織化するために使用される <a href="#">GroupDescriptor</a> オブジェクトのコレクションを取得します。

 <b>IsLoadingData</b>	<b>ClientViewSource</b> が現在データをロード中かどうかを示す値を取得します。
 <b>LoadCommand</b>	この <b>ClientViewSource</b> で <b>Load</b> を呼び出す <b>System.Windows.Input.ICommand</b> を取得します。
 <b>LoadDelay</b>	自動データロード操作が開始されるまでの遅延時間を取得または設定します。自動ロードを促す変更が発生した時点から 結果としての <b>Load</b> が開始される時点までが遅延時間です。 デフォルトの遅延時間は、25 ミリ秒です。
 <b>LoadSize</b>	<b>Load</b> が実行されるたびにロードされる項目の最大数を取得または設定します。0 の場合は、要求されたエンティティがすべてロードされます。デフォルトは0 です。
 <b>MoveToFirstOnLoad</b>	<b>Load</b> 操作完了後の最初の項目を現在の項目にすることを示す値を取得または設定します（現在の項目が他の方法で設定されていない場合）。
 <b>Name</b>	C1DataSource.ViewSources コレクション内で参照される、この <b>ClientViewSource</b> の名前を取得します。 デフォルトでは、これは EntitySetName（Entity Framework の場合）または QueryName（RIA サービスの場合）によって決定されますが、 <b>NameOverride</b> でオーバーライドできます。
 <b>NameOverride</b>	<b>Name</b> プロパティの値をオーバーライドする値を取得または設定します。
 <b>PageSize</b>	<b>DataView</b> の各ページに表示される項目数、または <b>仮想モード</b> 時に各クエリーで取得する項目数を取得または設定します。0 はページングが無効であることを示します。
 <b>SortDescriptors</b>	データのソートに使用される <b>SortDescriptor</b> オブジェクトのコレクションを取得します。
 <b>VirtualMode</b>	<b>ClientViewSource</b> が仮想モードかどうかを示す値を取得または設定します。仮想モードは、遅延やパフォーマンスの低下を招くことなく、またページング

		の手間をかける必要もなく、GUI コントロールを大規模なデータセットに直接連結できる革新的な技術です。 デフォルトでは、仮想モードは無効です（デフォルト値は <a href="#">VirtualModeKind.None</a> ）。
--	--	---

[Top](#)

## See Also

### Reference

[ClientViewSource Class](#)

[C1.Data.DataSource Namespace](#)

### AutoLoad Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : AutoLoad Property

起動時や、[ClientViewSource](#)

によって作成されたクエリーが影響を受ける変更が発生したときに、[Load](#)  
が自動的に呼び出されるかどうかを示す値を取得または設定します。 デフォルトは True  
です。

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.CategoryAttribute("Common")&gt; &lt;System.ComponentModel.DefaultValueAttribute()&gt; Public Property AutoLoad As System.Boolean</pre>	
C#	
<pre>[System.ComponentModel.Category("Common")] [System.ComponentModel.DefaultValue()] public System.bool AutoLoad {get; set;}</pre>	

## Remarks

[AutoLoad](#) が true の場合、ロードクエリーに影響を与えるプロパティが変更されると、  
指定された [LoadDelay](#) 経過後に [Load](#) が呼び出されます。  
クエリーに影響を与えるプロパティの例としては、[PageSize](#)、[FilterOperator](#)  
などがあります。また、[FilterDescriptors](#) のような依存オブジェクトコレクションの変更や、  
それらのコレクションに含まれる要素の依存プロパティの変更も、クエリーに影響を与え、  
自動 [Load](#) を促します。



# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ClientViewSource Class](#)  
[ClientViewSource Members](#)

## BaseView Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : BaseView Property

[ClientViewSource](#) がクエリーを作成するための基礎として使用する [C1.Data.ClientView<T>](#) のインスタンスを取得または設定します。

# Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute()&gt; Public Property BaseView As View</pre>	
C#	
<pre>[System.ComponentModel.DefaultValue()] public View BaseView {get; set;}</pre>	

# Exceptions

Exception	Description
<b>System.ArgumentException</b>	値が null でなく、 <a href="#">C1.Data.ClientView&lt;T&gt;</a> のインスタンスでもありません。

# Remarks

[ClientViewSource](#) は、BaseView に対してフィルタ処理、ソート、グループ化、ページングなどを適用します。

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### CacheTimeout Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : CacheTimeout Property

仮想モードでロードされたエンティティが、必要かどうかのチェックがされることなくキャッシュに維持される時間を取得または設定します。 [CacheTimeout](#) よりも長い時間、エンティティが使用されることも必要と判断されることもなかった場合、[ClientViewSource](#) はそのエンティティをキャッシュから削除できます。

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute(&gt;&gt; Public Property CacheTimeout As System.TimeSpan</pre>	
C#	
<pre>[System.ComponentModel.DefaultValue()] public System.TimeSpan CacheTimeout {get; set;}</pre>	

## Remarks

[VirtualMode](#) が [VirtualModeKind.None](#) の場合、このプロパティは使用されません。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

## CurrentClientView Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : CurrentClientView Property

エンティティのロードに使用される現在のクライアントビュー、または仮想モードでは null を取得します。

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.BrowsableAttribute(False)>  
Public Property CurrentClientView As View
```

C#

```
[System.ComponentModel.Browsable(false)]  
public View CurrentClientView {get; set;}
```

## Remarks

[CurrentClientView](#) を使用すると、[CurrentClientView](#) にライブビュー演算子を適用することで、[ClientViewSource](#) の上にクライアントビューを構築することができます。

エンティティのロードに使用されるクエリーが変更されるたびに、このプロパティの値が変更され、[PropertyChanged](#) イベントが発生します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

## DataView Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : DataView Property

最後のロード操作の結果であるエンティティの現在のビューを取得します。

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.BrowsableAttribute(False)&gt; Public ReadOnly Property DataView As ClientCollectionView</pre>	
C#	
<pre>[System.ComponentModel.Browsable(false)] public ClientCollectionView DataView {get;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### FilterDescriptors Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : FilterDescriptors Property

ロード実行時に使用される [FilterDescriptor](#) オブジェクトのコレクションを取得します。

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.CategoryAttribute("Common")&gt; Public ReadOnly Property FilterDescriptors As System.Windows.Controls.FilterDescriptorCollection</pre>	
C#	
<pre>[System.ComponentModel.Category("Common")] public System.Windows.Controls.FilterDescriptorCollection FilterDescriptors {get;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### FilterOperator Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : FilterOperator Property

フィルタコレクション内の [FilterDescriptors](#)

どうしを結合するために使用される論理演算子を取得または設定します。

デフォルト値は、[FilterDescriptorLogicalOperator.And](#) です。

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.CategoryAttribute("Common")>  
Public Property FilterOperator As  
System.Windows.Controls.FilterDescriptorLogicalOperator
```

C#

```
[System.ComponentModel.Category("Common")]  
public System.Windows.Controls.FilterDescriptorLogicalOperator FilterOperator  
{get; set;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### GroupDescriptors Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : GroupDescriptors Property

読み込んだエンティティをいくつかのグループに組織化するために使用される

[GroupDescriptor](#) オブジェクトのコレクションを取得します。

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.CategoryAttribute("Common")>  
Public ReadOnly Property GroupDescriptors As  
System.Windows.Controls.GroupDescriptorCollection
```

C#

```
[System.ComponentModel.Category("Common")]  
public System.Windows.Controls.GroupDescriptorCollection GroupDescriptors  
{get;}
```

## Remarks

グループ化は WPF および Silverlight でのみ機能します。WinForms のデータ連結はグループ化をサポートしていないため、これは無視されます。

### GroupDescriptor

が適用された場合、データは本質的にグループ化プロパティに基づいてソートされます。グループ化プロパティを降順でソートするには、**Descending** 方向を使用して、そのプロパティの [SortDescriptors](#) コレクションに [SortDescriptor](#) を追加します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### IsLoadingData Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : IsLoadingData Property

[ClientViewSource](#) が現在データをロード中かどうかを示す値を取得します。

## Syntax

Visual Basic (Declaration)

<b>Public ReadOnly Property</b> IsLoadingData <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool IsLoadingData { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### LoadCommand Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : LoadCommand Property

この [ClientViewSource](#) で [Load](#) を呼び出す **System.Windows.Input.ICommand** を取得します。

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> LoadCommand <b>As</b> System.Windows.Input.ICommand	
C#	
<b>public</b> System.Windows.Input.ICommand LoadCommand { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

## LoadDelay Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : LoadDelay Property

自動データロード操作が開始されるまでの遅延時間を取得または設定します。  
自動ロードを促す変更が発生した時点から 結果としての [Load](#)  
が開始される時点までが遅延時間です。 デフォルトの遅延時間は、25 ミリ秒です。

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.DefaultValueAttribute(>  
Public Property LoadDelay As System.TimeSpan
```

C#

```
[System.ComponentModel.DefaultValue()]  
public System.TimeSpan LoadDelay {get; set;}
```

## Remarks

指定された時間間隔内に発生した複数の変更が、1 つの [Load](#)  
操作に集約されます。変更が発生するたびに、遅延タイマーがリセットされます。  
これにより、多数の変更が発生しても、それらが直前に発生した変更から指定された遅延時間  
以内に発生する限りは、  
1 回の呼び出しに結合することができます。変更が発生しないまま遅延タイマーの時間が経過  
すると、[Load](#) が呼び出されます。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server  
2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or  
later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

## LoadSize Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : LoadSize Property

[Load](#) が実行されるたびにロードされる項目の最大数を取得または設定します。  
0 の場合は、要求されたエンティティがすべてロードされます。 デフォルトは 0 です。



## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.DefaultValueAttribute()>  
Public Property LoadSize As System.Integer
```

C#

```
[System.ComponentModel.DefaultValue()]  
public System.int LoadSize {get; set;}
```

## Remarks

[PageSize](#) と [LoadSize](#) がどちらも 0 でない場合、エンティティは、[LoadSize](#) に最も近い [PageSize](#) の倍数を使用してロードされます。  
これにより、ページを部分的にロードすることなく、複数のページを一度にロードできます。

[ClientViewSource](#) が [仮想モード](#) の場合、このプロパティは無視されます。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### MoveToFirstOnLoad Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : MoveToFirstOnLoad Property

[Load](#) 操作完了後の最初の項目を現在の項目にすることを示す値を取得または設定します（現在の項目が他の方法で設定されていない場合）。

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.DefaultValueAttribute()>  
Public Property MoveToFirstOnLoad As System.Boolean
```

C#	
----	--

[System.ComponentModel.DefaultValue()] <code>public System.bool MoveToFirstOnLoad {get; set;}</code>	
---	--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### Name Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : Name Property

C1DataSource.ViewSources コレクション内で参照される、この [ClientViewSource](#) の名前を取得します。 デフォルトでは、これは EntitySetName (Entity Framework の場合) または QueryName (RIA サービスの場合) によって決定されますが、[NameOverride](#) でオーバーライドできます。

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<code>Public Overridable ReadOnly Property Name As System.String</code>	
---	--

C#	
----	--

<code>public virtual System.string Name {get;}</code>	
---	--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

## NameOverride Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : NameOverride Property

**Name** プロパティの値をオーバーライドする値を取得または設定します。

## Syntax

Visual Basic (Declaration)	
<System.ComponentModel.DefaultValueAttribute(>> <b>Public Property</b> NameOverride <b>As</b> System.String	
C#	
[System.ComponentModel.DefaultValue()] <b>public</b> System.string NameOverride {get; set;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

## PageSize Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : PageSize Property

**DataSource** の各ページに表示される項目数、  
または**仮想モード**時に各クエリーで取得する項目数を  
取得または設定します。0はページングが無効であることを示します。

## Syntax

Visual Basic (Declaration)	
<System.ComponentModel.CategoryAttribute("Common")> <System.ComponentModel.DefaultValueAttribute(>>	

```
Public Property PageSize As System.Integer
```

```
C#
```

```
[System.ComponentModel.Category("Common")]  
[System.ComponentModel.DefaultValue()]  
public System.int PageSize {get; set;}
```

## Remarks

仮想モードでない場合に、 ページサイズが0以外の値のときは、各 [Load](#) 操作でロードされるエンティティの数は、サーバー側のページングを使用して制限されます。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### SortDescriptors Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : SortDescriptors Property

データのソートに使用される [SortDescriptor](#) オブジェクトのコレクションを取得します。

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.CategoryAttribute("Common")>  
Public ReadOnly Property SortDescriptors As  
System.Windows.Controls.SortDescriptorCollection
```

```
C#
```

```
[System.ComponentModel.Category("Common")]  
public System.Windows.Controls.SortDescriptorCollection SortDescriptors  
{get;}
```

## Remarks

[Load](#) 操作では、[SortDescriptors](#) を使用して サーバー側でソートが実行されます。  
ロードされたエンティティに対してクライアント上で変更が加えられた場合、指定されたソートはクライアント側でも適用され、[DataView](#) に変更が反映されます。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### VirtualMode Property

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : VirtualMode Property

[ClientViewSource](#) が仮想モードかどうかを示す値を取得または設定します。  
仮想モードは、遅延やパフォーマンスの低下を招くことなく、またページングの手間をかける必要もなく、GUI コントロールを大規模なデータセットに直接連結できる革新的な技術です。  
デフォルトでは、仮想モードは無効です（デフォルト値は [VirtualModeKind.None](#)）。

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.CategoryAttribute("Common")&gt; &lt;System.ComponentModel.DefaultValueAttribute()&gt; Public Property VirtualMode As VirtualModeKind</pre>	
C#	
<pre>[System.ComponentModel.Category("Common")] [System.ComponentModel.DefaultValue()] public VirtualModeKind VirtualMode {get; set;}</pre>	

## Remarks

仮想モードでは、[ClientViewSource](#) はインテリジェントに、またエンドユーザーと開発者の双方から透過的に、画面に表示する必要があるエンティティだけをロードします。

仮想モードを有効にするには、[VirtualMode](#) が true に設定されたコントロールハンドラがある場合は、このプロパティを [Managed](#) に設定します。 そうでない場合は、[VirtualModeKind.Unmanaged](#) に設定します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



[ClientViewSource Class](#)

[ClientViewSource Members](#)

[VirtualModeKind Enumeration](#)

### Events

>

Name	Description
 <a href="#">LoadedData</a>	ロード操作が完了したとき、またはロード操作中に例外が生成されたときに発生します。
 <a href="#">PropertyChanged</a>	プロパティ値が変更されたときに発生します。

[Top](#)

## See Also

### Reference

[ClientViewSource Class](#)

[C1.Data.DataSource Namespace](#)

### LoadedData Event

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : LoadedData Event

ロード操作が完了したとき、またはロード操作中に例外が生成されたときに発生します。

## Syntax

Visual Basic (Declaration)

<b>Public Event</b> LoadedData <b>As</b> System.EventHandler(Of ClientViewLoadedEventArgs)
C#
<b>public event</b> System.EventHandler<ClientViewLoadedEventArgs> LoadedData

## Event Data

The event handler receives an argument of type [ClientViewLoadedEventArgs](#) containing data related to this event. The following **ClientViewLoadedEventArgs** properties provide information specific to this event.

Property	Description
<a href="#">Error</a>	Gets the loading error if the loading failed.
<a href="#">HasError</a>	Gets a value indicating whether the loading has failed. If true, inspect the <a href="#">Error</a> property for details.
<a href="#">IsErrorHandled</a>	Gets a value indicating whether the loading error has been marked as handled by calling <a href="#">MarkErrorAsHandled</a> .
<a href="#">Items</a>	Gets all entities loaded by a <a href="#">client view</a> .
<a href="#">TotalItemCount</a>	Gets the total number of rows for the original query without any paging applied to it.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)

[ClientViewSource Members](#)

### PropertyChanged Event

[C1.Data.DataSource Namespace](#) > [ClientViewSource Class](#) : PropertyChanged Event

プロパティ値が変更されたときに発生します。

# Syntax

Visual Basic (Declaration)	
<code>Public Event PropertyChanged As System.ComponentModel.PropertyChangedEventHandler</code>	
C#	
<code>public event System.ComponentModel.PropertyChangedEventHandler PropertyChanged</code>	

## Event Data

The event handler receives an argument of type `System.ComponentModel.PropertyChangedEventArgs` containing data related to this event. The following **PropertyChangedEventArgs** properties provide information specific to this event.

Property	Description
<b>PropertyName</b>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSource Class](#)  
[ClientViewSource Members](#)

## ClientViewSourceException

[C1.Data.DataSource Namespace](#) : ClientViewSourceException Class

この例外は、[ClientViewSource](#) が正しく構成されていないこと、または [Load](#) 操作中にエラーが発生したことを示します。

## Object Model

ClientViewSourceException



# Syntax

Visual Basic (Declaration)	
<pre>Public Class ClientViewSourceException     Inherits System.Exception</pre>	
C#	
<pre>public class ClientViewSourceException : System.Exception</pre>	

# Inheritance Hierarchy

System.Object  
System.Exception  
**C1.Data.DataSource.ClientViewSourceException**

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ClientViewSourceException Members](#)  
[C1.Data.DataSource Namespace](#)

## Overview

[C1.Data.DataSource Namespace](#) : ClientViewSourceException Class

この例外は、[ClientViewSource](#) が正しく構成されていないこと、または [Load](#) 操作中にエラーが発生したことを示します。

# Object Model

ClientViewSourceException

# Syntax

Visual Basic (Declaration)	
<pre>Public Class ClientViewSourceException</pre>	

<b>Inherits</b> <code>System.Exception</code>
C#
<code>public class ClientViewSourceException : System.Exception</code>

## Inheritance Hierarchy

System.Object  
 System.Exception  
**C1.Data.DataSource.ClientViewSourceException**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSourceException Members](#)  
[C1.Data.DataSource Namespace](#)


## Members

[Properties](#) [Methods](#) [Events](#)

[C1.Data.DataSource Namespace](#) : [ClientViewSourceException Class](#)


The following tables list the members exposed by [ClientViewSourceException](#).








## Public Constructors

	Name	Description
	<a href="#">ClientViewSourceException Constructor</a>	Overloaded.

[Top](#)





## Public Properties

	Name	Description
	<a href="#">Data</a>	(Inherited from <code>System.Exception</code> )

	<a href="#">HelpLink</a>	(Inherited from System.Exception)
	<a href="#">HResult</a>	(Inherited from System.Exception)
	<a href="#">InnerException</a>	(Inherited from System.Exception)
	<a href="#">Message</a>	(Inherited from System.Exception)
	<a href="#">Source</a>	(Inherited from System.Exception)
	<a href="#">StackTrace</a>	(Inherited from System.Exception)
	<a href="#">TargetSite</a>	(Inherited from System.Exception)


[Top](#)

## Public Methods

	Name	Description
	<a href="#">GetBaseException</a>	(Inherited from System.Exception)
	<a href="#">GetObjectData</a>	(Inherited from System.Exception)
	<a href="#">GetType</a>	(Inherited from System.Exception)
	<a href="#">ToString</a>	(Inherited from System.Exception)

[Top](#)

## Protected Events

	Name	Description
	<a href="#">SerializeObjectState</a>	(Inherited from System.Exception)

[Top](#)

## See Also

### Reference

## ClientViewSourceException Constructor

C1.Data.DataSource Namespace > ClientViewSourceException Class : ClientViewSourceException  
Constructor

### Overload List

Overload	Description
<a href="#">ClientViewSourceException Constructor()</a>	<a href="#">ClientViewSourceException</a> クラスの新しいインスタンスを初期化します。
<a href="#">ClientViewSourceException Constructor(String)</a>	指定されたエラーメッセージを使用して、 <a href="#">ClientViewSourceException</a> クラスの新しいインスタンスを初期化します。
<a href="#">ClientViewSourceException Constructor(String,Exception)</a>	指定されたエラーメッセージ、およびこの例外の原因となった内部例外への参照を使用して、 <a href="#">ClientViewSourceException</a> クラスの新しいインスタンスを初期化します。

### Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

### See Also

#### Reference

[ClientViewSourceException Class](#)  
[ClientViewSourceException Members](#)

#### ClientViewSourceException Constructor()

C1.Data.DataSource Namespace > ClientViewSourceException Class > ClientViewSourceException  
Constructor : ClientViewSourceException Constructor()

[ClientViewSourceException](#) クラスの新しいインスタンスを初期化します。

### Syntax

Visual Basic (Declaration)

```
Public Function New()
```

```
C#
```

```
public ClientViewSourceException()
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSourceException Class](#)

[ClientViewSourceException Members](#)

[Overload List](#)

### ClientViewSourceException Constructor(String)

[C1.Data.DataSource Namespace](#) > [ClientViewSourceException Class](#) > [ClientViewSourceException](#)

**Constructor** : ClientViewSourceException Constructor(String)

この例外の理由について説明するエラーメッセージ。

指定されたエラーメッセージを使用して、[ClientViewSourceException](#) クラスの新しいインスタンスを初期化します。

## Syntax

Visual Basic (Declaration)

```
Public Function New( _  
    ByVal message As System.String _  
)
```

```
C#
```

```
public ClientViewSourceException(  
    System.string message  
)
```

### Parameters

*message*

この例外の理由について説明するエラーメッセージ。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSourceException Class](#)  
[ClientViewSourceException Members](#)  
[Overload List](#)

### ClientViewSourceException Constructor(String,Exception)

[C1.Data.DataSource Namespace](#) > [ClientViewSourceException Class](#) > [ClientViewSourceException Constructor](#) : ClientViewSourceException Constructor(String,Exception)

この例外の理由について説明するエラーメッセージ。

現在の例外の原因になった例外。

指定されたエラーメッセージ、およびこの例外の原因となった内部例外への参照を使用して、[ClientViewSourceException](#) クラスの新しいインスタンスを初期化します。

## Syntax

Visual Basic (Declaration)

```
Public Function New( _  
    ByVal message As System.String, _  
    ByVal inner As System.Exception _  
)
```

C#

```
public ClientViewSourceException(  
    System.string message,  
    System.Exception inner  
)
```

### Parameters

*message*

この例外の理由について説明するエラーメッセージ。

*inner*

現在の例外の原因になった例外。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientViewSourceException Class](#)  
[ClientViewSourceException Members](#)  
[Overload List](#)

## Enumerations

### VirtualModeKind

[C1.Data.DataSource Namespace](#) : VirtualModeKind Enumeration

[ClientViewSource](#) の有効な仮想モードの列挙です。[VirtualMode](#) プロパティで使用されます。

## Syntax

Visual Basic (Declaration)	
<pre>Public Enum VirtualModeKind     Inherits System.Enum</pre>	
C#	
<pre>public enum VirtualModeKind : System.Enum</pre>	

## Members

Member	Description
<b>Managed</b>	仮想モードが <a href="#">ClientViewSource</a> に連結された GUI コントロールによって管理されます。この GUI コントロールには、 <a href="#">VirtualMode</a> プロパティが true に設定された <a href="#">コントロールハンドラ</a> が必要です。
<b>None</b>	仮想モードが無効になります。

<b>Unmanaged</b>	<p>仮想モードは、<a href="#">コントロールハンドラ</a>によって管理されるのではなく、自身に連結されているコントロールを認識していない <a href="#">ClientViewSource</a> 自身によって管理されます。</p> <p>このオプションは、コントロールハンドラにより仮想モードをサポートする GUI コントロールがない場合にのみ使用してください。 このオプションにより、任意の GUI 連結コントロールを使用した仮想モードの使用が可能になりますが（コントロールハンドラの有無にかかわらず）、<a href="#">Managed</a> オプションを使用できない場合にのみ、注意して使用してください。</p> <p>詳細については、Studio for Entity Framework ドキュメントの「プログラミングガイド」を参照してください。</p>
------------------	--

## Inheritance Hierarchy

System.Object  
  System.ValueType  
    System.Enum  
      **C1.Data.DataSource.VirtualModeKind**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.Data.DataSource Namespace](#)

# C1.Data.Transactions Namespace

## Overview

[Inheritance Hierarchy](#)

## Classes

	Class	Description
	<a href="#">ClientTransaction</a>	



# See Also

## Reference

[C1.Silverlight.Data.Entity.5 Assembly](#)

# Classes

## ClientTransaction

[C1.Data.Transactions Namespace](#) : ClientTransaction Class

# Object Model

ClientTransaction

# Syntax

Visual Basic (Declaration)	
<pre>Public Class ClientTransaction     Implements C1.LiveLinq.ITransaction</pre>	
C#	
<pre>public class ClientTransaction : C1.LiveLinq.ITransaction</pre>	

# Inheritance Hierarchy

System.Object  
    **C1.Data.Transactions.ClientTransaction**

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ClientTransaction Members](#)

[C1.Data.Transactions Namespace](#)

# Overview

[C1.Data.Transactions Namespace](#) : ClientTransaction Class

## Object Model

ClientTransaction

## Syntax

Visual Basic (Declaration)	
<pre>Public Class ClientTransaction     Implements C1.LiveLinq.ITransaction</pre>	
C#	
<pre>public class ClientTransaction : C1.LiveLinq.ITransaction</pre>	

## Inheritance Hierarchy

System.Object  
    **C1.Data.Transactions.ClientTransaction**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference


[ClientTransaction Members](#)  
[C1.Data.Transactions Namespace](#)

## Members

[Properties](#) [Methods](#) [Events](#)  
  
[C1.Data.Transactions Namespace](#) : ClientTransaction Class

The following tables list the members exposed by [ClientTransaction](#).

## Public Constructors

	Name	Description
	<a href="#">ClientTransaction Constructor</a>	

[Top](#)

## Public Properties

	Name	Description
	<a href="#">HasChanges</a>	
	<a href="#">State</a>	

[Top](#)

## Public Methods

	Name	Description
	<a href="#">Commit</a>	
	<a href="#">Dispose</a>	
	<a href="#">Rollback</a>	
	<a href="#">Scope</a>	
	<a href="#">ScopeDataContext</a>	

[Top](#)

## Public Events

	Name	Description
	<a href="#">PropertyChanged</a>	

[Top](#)

## See Also

## Reference

[ClientTransaction Class](#)

[C1.Data.Transactions Namespace](#)

## ClientTransaction Constructor

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : ClientTransaction Constructor

## Syntax

Visual Basic (Declaration)

```
Public Function New( _  
    ByVal parent As ClientTransaction _  
)
```

C#

```
public ClientTransaction(  
    ClientTransaction parent  
)
```

## Parameters

*parent*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ClientTransaction Class](#)

[ClientTransaction Members](#)

## Methods

[C1.Data.Transactions Namespace](#) : ClientTransaction Class

For a list of all members of this type, see [ClientTransaction members](#).

## Public Methods

	Name	Description
	<a href="#">Commit</a>	
	<a href="#">Dispose</a>	
	<a href="#">Rollback</a>	
	<a href="#">Scope</a>	
	<a href="#">ScopeDataContext</a>	

[Top](#)

## See Also

### Reference

[ClientTransaction Class](#)

[C1.Data.Transactions Namespace](#)

### Commit Method

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : Commit Method

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> <a href="#">Commit()</a>	
C#	
<b>public void</b> <a href="#">Commit()</a>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Class](#)  
[ClientTransaction Members](#)

Dispose Method

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : Dispose Method

Syntax

Visual Basic (Declaration)	
<code>Public Sub Dispose()</code>	
C#	
<code>public void Dispose()</code>	

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[ClientTransaction Class](#)  
[ClientTransaction Members](#)

Rollback Method

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : Rollback Method

Syntax

Visual Basic (Declaration)	
<code>Public Sub Rollback()</code>	
C#	
<code>public void Rollback()</code>	

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ClientTransaction Class](#)  
[ClientTransaction Members](#)

## Scope Method

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : Scope Method

# Syntax

Visual Basic (Declaration)	
<code>Public Function Scope() As System.IDisposable</code>	
C#	
<code>public System.IDisposable Scope()</code>	

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ClientTransaction Class](#)  
[ClientTransaction Members](#)

## ScopeDataContext Method

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : ScopeDataContext Method

# Syntax

Visual Basic (Declaration)	
<code>Public Function ScopeDataContext( _     ByVal entity As System.Object _ ) As System.Object</code>	
C#	

```
public System.object ScopeDataContext(  
    System.object entity  
)
```

Parameters

*entity*

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

- [ClientTransaction Class](#)
- [ClientTransaction Members](#)

Properties

[C1.Data.Transactions Namespace](#) : ClientTransaction Class

For a list of all members of this type, see [ClientTransaction members](#).

Public Properties

	Name	Description
	<a href="#">HasChanges</a>	
	<a href="#">State</a>	

[Top](#)

See Also

Reference

- [ClientTransaction Class](#)
- [C1.Data.Transactions Namespace](#)

HasChanges Property

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : HasChanges Property



## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property HasChanges As System.Boolean</code>	
C#	
<code>public System.bool HasChanges {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Class](#)

[ClientTransaction Members](#)

### State Property

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : State Property

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property State As TransactionState</code>	
C#	
<code>public TransactionState State {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Class](#)  
[ClientTransaction Members](#)

Events

[C1.Data.Transactions Namespace](#) : [ClientTransaction Class](#)

For a list of all members of this type, see [ClientTransaction members](#).

Public Events

	Name	Description
	<a href="#">PropertyChanged</a>	

[Top](#)

See Also

Reference

[ClientTransaction Class](#)  
[C1.Data.Transactions Namespace](#)

PropertyChanged Event

[C1.Data.Transactions Namespace](#) > [ClientTransaction Class](#) : [PropertyChanged Event](#)

Syntax

Visual Basic (Declaration)	
<code>Public Event PropertyChanged As System.ComponentModel.PropertyChangedEventHandler</code>	
C#	
<code>public event System.ComponentModel.PropertyChangedEventHandler PropertyChanged</code>	

Event Data

The event handler receives an argument of type `System.ComponentModel.PropertyChangedEventArgs` containing data related to this event. The following **PropertyChangedEventArgs** properties provide information specific to this event.

Property	Description
----------	-------------

<b>PropertyName</b>	
---------------------	--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ClientTransaction Class](#)

[ClientTransaction Members](#)

# C1.Data.Util Namespace

## Overview

[Inheritance Hierarchy](#)

## Classes

	Class	Description
	<a href="#">DesignTime</a>	

## See Also

### Reference

[C1.Silverlight.Data.Entity.5 Assembly](#)

## Classes

## DesignTime

[C1.Data.Util Namespace](#) : DesignTime Class

## Object Model

DesignTime

## Syntax

Visual Basic (Declaration)	
<code>Public Class DesignTime</code>	
C#	
<code>public class DesignTime</code>	

## Inheritance Hierarchy

System.Object

**C1.Data.Util.DesignTime**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[DesignTime Members](#)

[C1.Data.Util Namespace](#)

## Overview

[C1.Data.Util Namespace](#) : DesignTime Class

## Object Model

DesignTime

## Syntax

Visual Basic (Declaration)	
<code>Public Class DesignTime</code>	
C#	
<code>public class DesignTime</code>	

## Inheritance Hierarchy

System.Object

**C1.Data.Util.DesignTime**

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference


[DesignTime Members](#)  
[C1.Data.Util Namespace](#)

## Members

[C1.Data.Util Namespace](#) : DesignTime Class

The following tables list the members exposed by [DesignTime](#).

## Public Constructors

	Name	Description
	<a href="#">DesignTime Constructor</a>	

[Top](#)

# See Also

## Reference

[DesignTime Class](#)  
[C1.Data.Util Namespace](#)

## DesignTime Constructor

[C1.Data.Util Namespace](#) > [DesignTime Class](#) : DesignTime Constructor

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public DesignTime()</code>	

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also


## Reference

[DesignTime Class](#)  
[DesignTime Members](#)

# C1.Silverlight.Data Namespace

## Overview

## Classes

	Class	Description
	<a href="#">ControlHandler</a>	サポートされているタイプの GUI コントロールを C1DataSource に接続するコントロールハンドラを表します。 これにより、これらのコントロールに <a href="#">ルックアップ列</a> や <a href="#">仮想モード</a> などの追加機能を提供できます。

# See Also

## Reference

[C1.Silverlight.Data.Entity.5 Assembly](#)

## Classes

## ControlHandler

[C1.Silverlight.Data Namespace](#) : ControlHandler Class

サポートされているタイプの GUI コントロールを C1DataSource に接続するコントロールハンドラを表します。 これにより、これらのコントロールに [ルックアップ列](#)や[仮想モード](#)などの追加機能を提供できます。

## Object Model



## Syntax

Visual Basic (Declaration)

```
Public Class ControlHandler  
    Inherits C1.Data.DataSource.BaseControlHandler
```

C#

```
public class ControlHandler : C1.Data.DataSource.BaseControlHandler
```

## Remarks

GUI コントロールに[コントロールハンドラ](#)を接続するには、このクラスのインスタンスを GUI コントロールの [C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandlerProperty](#) 添付プロパティに割り当てます。

サポートされている GUI

コントロールには、**System.Windows.Controls.DataGrid**、C1.Silverlight.FlexGrid.C1FlexGrid、C1.Silverlight.DataGrid.C1DataGrid があります。

### **System.Windows.Controls.DataGrid**

には[ルックアップ列](#)に必要な機能がないため、ルックアップ列はサポートされません。

## Inheritance Hierarchy

System.Object

System.Windows.DependencyObject

[C1.Data.DataSource.BaseControlHandler](#)

**C1.Silverlight.Data.ControlHandler**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ControlHandler Members](#)

[C1.Silverlight.Data Namespace](#)

## Overview

[C1.Silverlight.Data Namespace](#) : ControlHandler Class

サポートされているタイプの GUI コントロールを C1DataSource に接続するコントロールハンドラを表します。これにより、これらのコントロールに [ルックアップ列](#)や[仮想モード](#)などの追加機能を提供できます。

## Object Model



## Syntax

Visual Basic (Declaration)	
<pre>Public Class ControlHandler     Inherits C1.Data.DataSource.BaseControlHandler</pre>	
C#	
<pre>public class ControlHandler : C1.Data.DataSource.BaseControlHandler</pre>	

## Remarks

GUI コントロールに[コントロールハンドラ](#)を接続するには、このクラスのインスタンスを GUI コントロールの [C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandlerProperty](#) 添付プロパティに割り当てます。

サポートされている GUI コントロールには、**System.Windows.Controls.DataGrid**、C1.Silverlight.FlexGrid.C1FlexGrid、C1.Silverlight.DataGrid.C1DataGrid があります。

### **System.Windows.Controls.DataGrid**

には[ルックアップ列](#)に必要な機能がないため、ルックアップ列はサポートされません。

## Inheritance Hierarchy

System.Object  
System.Windows.DependencyObject  
[C1.Data.DataSource.BaseControlHandler](#)  
**C1.Silverlight.Data.ControlHandler**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



## Reference

[ControlHandler Members](#)

[C1.Silverlight.Data Namespace](#)


## Members

[Fields](#) [Properties](#) [Methods](#)

[C1.Silverlight.Data Namespace](#) : [ControlHandler Class](#)


The following tables list the members exposed by [ControlHandler](#).

## Public Constructors

	Name	Description
	<a href="#">ControlHandler Constructor</a>	




[Top](#)



## Public Fields

	Name	Description
	<a href="#">DataSourceProperty</a>	<a href="#">DataSource</a> プロパティの <a href="#">DependencyProperty</a> 。

[Top](#)





## Public Properties

	Name	Description
	<a href="#">AutoLookup</a>	ナビゲーション（外部キー、ルックアップ）プロパティに連結されているデータグリッド列をコンボボックス列に変換するかどうかを示す値を取得または設定します。 これにより、ユーザーは、ドロップダウンリストから値を選択して、正しい値を確認したり値を編集できます。デフォルト値は false です。 (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )
	<a href="#">DataSource</a>	コントロールハンドラの <a href="#">データソース</a> を取得または設定します。
	<a href="#">Dispatcher</a>	(Inherited from <a href="#">System.Windows.DependencyObject</a> )

 <a href="#">SupportsVirtualMode</a>	この <a href="#">コントロールハンドラ</a> が仮想モードをサポートするかどうかを示す値を取得します。 (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )
 <a href="#">VirtualMode</a>	<a href="#">C1.Data.DataSource.ClientViewSource</a> で指定されている仮想モードをこのコントロールハンドラで管理するかどうかを示す値を取得または設定します。 (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )

[Top](#)

## Public Methods

Name	Description
 <a href="#">Apply</a>	この <a href="#">コントロールハンドラ</a> の設定を現在のコントロールに強制的に適用させます。 (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )
 <a href="#">ClearValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">GetAnimationBaseValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
 <a href="#">SetValue</a>	(Inherited from System.Windows.DependencyObject)

[Top](#)

## See Also

### Reference

[ControlHandler Class](#)

[C1.Silverlight.Data Namespace](#)

## ControlHandler Constructor

[C1.Silverlight.Data Namespace](#) > [ControlHandler Class](#) : ControlHandler Constructor

# Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public ControlHandler()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ControlHandler Class](#)  
[ControlHandler Members](#)

## Properties

[C1.Silverlight.Data Namespace](#) : ControlHandler Class

For a list of all members of this type, see [ControlHandler members](#).

## Public Properties

Name	Description
 <a href="#">AutoLookup</a>	ナビゲーション（外部キー、ルックアップ）プロパティに連結されているデータグリッド列をコンボボックス列に変換するかどうかを示す値を取得または設定します。  これにより、ユーザーは、ドロップダウンリストから値を選択して、正しい値を確認したり値を編集できます。デフォルト値は false です。(Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )
 <a href="#">DataSource</a>	コントロールハンドラの <a href="#">データソース</a> を取得または設定します。
 <a href="#">Dispatcher</a>	(Inherited from System.Windows.DependencyObject)

 SupportsVirtualMode	このコントロールハンドラが仮想モードをサポートするかどうかを示す値を取得します。 (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )
 VirtualMode	<a href="#">C1.Data.DataSource.ClientViewSource</a> で指定されている仮想モードをこのコントロールハンドラで管理するかどうかを示す値を取得または設定します。 (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )

[Top](#)

## See Also

### Reference

[ControlHandler Class](#)

[C1.Silverlight.Data Namespace](#)

### DataSource Property

[C1.Silverlight.Data Namespace](#) > [ControlHandler Class](#) : DataSource Property

コントロールハンドラのデータソースを取得または設定します。

## Syntax

Visual Basic (Declaration)	
<b>Public Property</b> DataSource As C1DataSource	
C#	
<b>public</b> C1DataSource DataSource { <b>get</b> ; <b>set</b> ;}	

## Remarks

GUI コントロールが C1DataSource に直接連結されない場合は、このプロパティを設定する必要があります。たとえば、GUI コントロールが [ライブビュー](#)に連結される場合には設定が必要です。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ControlHandler Class](#)


[ControlHandler Members](#)

## Fields

[C1.Silverlight.Data Namespace](#) : [ControlHandler Class](#)

For a list of all members of this type, see [ControlHandler members](#).

## Public Fields

	Name	Description
 <b>S</b>	<a href="#">DataSourceProperty</a>	<a href="#">DataSource</a> プロパティの <a href="#">DependencyProperty</a> 。

[Top](#)

## See Also

### Reference

[ControlHandler Class](#)

[C1.Silverlight.Data Namespace](#)

### DataSourceProperty Field

[C1.Silverlight.Data Namespace](#) > [ControlHandler Class](#) : [DataSourceProperty](#) Field

[DataSource](#) プロパティの [DependencyProperty](#)。

## Syntax

Visual Basic (Declaration)	
<b>Public Shared ReadOnly</b> <a href="#">DataSourceProperty</a> <b>As</b> <a href="#">System.Windows.DependencyProperty</a>	
C#	
<b>public static readonly</b> <a href="#">System.Windows.DependencyProperty</a> <a href="#">DataSourceProperty</a>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ControlHandler Class](#)  
[ControlHandler Members](#)

# C1.Silverlight.Data.RiaServices Namespace

## Overview

## Classes

Class	Description
 <a href="#">C1DataSource</a>	GUI コントロールから <a href="#">RiaClientCache</a> 内のデータへのデータ連結を簡略化するデータソースコントロール。複数のコントロールをさまざまなクエリーに連結するために使用できます。
 <a href="#">RiaClientCache</a>	RIA サービスに固有のクライアント側キャッシュを表します。
 <a href="#">RiaClientScope</a>	
 <a href="#">RiaServicesExtensions</a>	
 <a href="#">RiaViewSource</a>	<a href="#">C1.Data.DataSource.ClientViewSource</a> クラスの RIA サービス固有バージョン。
 <a href="#">RiaViewSourceCollection</a>	<a href="#">RiaViewSource</a> オブジェクトの監視可能なコレクション。

# See Also

## Reference

[C1.Silverlight.Data.Entity.5 Assembly](#)

# Classes

## C1DataSource

[C1.Silverlight.Data.RiaServices Namespace](#) : C1DataSource Class

GUI コントロールから [RiaClientCache](#) 内のデータへのデータ連結を簡略化するデータソースコントロール。複数のコントロールをさまざまなクエリに連結するために使用できます。

## Object Model

C1DataSource

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>
<System.Windows.Markup.ContentPropertyAttribute("ViewSources")>
Public Class C1DataSource
    Inherits System.Windows.Controls.Control
```

C#

```
[System.Reflection.DefaultMember("Item")]
[System.Windows.Markup.ContentProperty("ViewSources")]
public class C1DataSource : System.Windows.Controls.Control
```

## Remarks

[RiaClientCache](#) 内のデータにコントロールを連結するには、[C1DataSource](#) を XAML ファイルに追加し、[コンテキストタイプ](#)を指定し、[ViewSources](#) コレクションに [RiaViewSource](#) オブジェクトを挿入して（クエリーベースの）ビューを定義し、次のように GUI コントロールを連結します。<DataGrid ItemsSource="{Binding Customers, ElementName=c1DataSource}"/> ここで、Customers は [ViewSources](#) コレクション内の [RiaViewSource](#) の名前、c1DataSource は [C1DataSource](#) の名前です。

## Inheritance Hierarchy

System.Object  
System.Windows.DependencyObject  
System.Windows.UIElement  
System.Windows.FrameworkElement  
System.Windows.Controls.Control  
**C1.Silverlight.Data.RiaServices.C1DataSource**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Members](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

## Overview

[C1.Silverlight.Data.RiaServices Namespace](#) : C1DataSource Class

GUI コントロールから [RiaClientCache](#) 内のデータへのデータ連結を簡略化するデータソースコントロール。複数のコントロールをさまざまなクエリーに連結するために使用できます。

## Object Model

C1DataSource

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>
<System.Windows.Markup.ContentPropertyAttribute("ViewSources")>
Public Class C1DataSource
    Inherits System.Windows.Controls.Control
```

C#

```
[System.Reflection.DefaultMember("Item")]
[System.Windows.Markup.ContentProperty("ViewSources")]
public class C1DataSource : System.Windows.Controls.Control
```

## Remarks

[RiaClientCache](#) 内のデータにコントロールを連結するには、[C1DataSource](#) を XAML ファイルに追加し、[コンテキストタイプ](#)を指定し、[ViewSources](#) コレクションに [RiaViewSource](#) オブジェクトを挿入して（クエリーベースの）ビューを定義し、次のように GUI コントロールを連結します。 <DataGrid ItemsSource="{Binding Customers,



ElementName=c1DataSource}"/> ここで、Customers は [ViewSources](#) コレクション内の [RiaViewSource](#) の名前、 c1DataSource は [C1DataSource](#) の名前です。

## Inheritance Hierarchy

System.Object  
  System.Windows.DependencyObject  
    System.Windows.UIElement  
      System.Windows.FrameworkElement  
        System.Windows.Controls.Control  
          **C1.Silverlight.Data.RiaServices.C1DataSource**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Members](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)


## Members

[Fields](#) [Properties](#) [Methods](#) [Events](#)

[C1.Silverlight.Data.RiaServices Namespace](#) : C1DataSource Class


The following tables list the members exposed by [C1DataSource](#).

## Public Constructors

	Name	Description
	<a href="#">C1DataSource Constructor</a>	<a href="#">C1DataSource</a> クラスの新しいインスタンスを初期化します。

[Top](#)
















## Public Fields


















	Name	Description
 	<a href="#">ControlHandlerProperty</a>	C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandler


















		添付プロパティを識別します。
--	--	----------------













[Top](#)

## Public Properties

	Name	Description
	<a href="#">ActualHeight</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ActualWidth</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">AllowDrop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Background</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">BorderBrush</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">BorderThickness</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">CacheMode</a>	(Inherited from System.Windows.UIElement)
	<a href="#">CharacterSpacing</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">ClientCache</a>	この <a href="#">C1DataSource</a> がデータにアクセスするために使用する <a href="#">RiaClientCache</a> を取得または設定します。
	<a href="#">ClientScope</a>	この <a href="#">C1DataSource</a> が属する <a href="#">クライアントスコープ</a> を取得します。
	<a href="#">Clip</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Cursor</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DataContext</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DesiredSize</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.DependencyObject)


 <a href="#">DomainContext</a>	<a href="#">ClientCache</a> の接続先の <b>System.ServiceModel.DomainServices.Client.DomainContext</b> を取得します。
 <a href="#">DomainContextTypeName</a>	デフォルトのクライアントキャッシュの取得に使用される <b>System.ServiceModel.DomainServices.Client.DomainContext</b> タイプの完全名を取得または設定します。
 <a href="#">Effect</a>	(Inherited from System.Windows.UIElement)
 <a href="#">FlowDirection</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">FontFamily</a>	(Inherited from System.Windows.Controls.Control)
 <a href="#">FontSize</a>	(Inherited from System.Windows.Controls.Control)
 <a href="#">FontStretch</a>	(Inherited from System.Windows.Controls.Control)
 <a href="#">FontStyle</a>	(Inherited from System.Windows.Controls.Control)
 <a href="#">FontWeight</a>	(Inherited from System.Windows.Controls.Control)
 <a href="#">Foreground</a>	(Inherited from System.Windows.Controls.Control)
 <a href="#">Height</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">HorizontalAlignment</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">HorizontalContentAlignment</a>	(Inherited from System.Windows.Controls.Control)
 <a href="#">IsEnabled</a>	(Inherited from System.Windows.Controls.Control)
 <a href="#">IsHitTestVisible</a>	(Inherited from System.Windows.UIElement)
 <a href="#">IsTabStop</a>	(Inherited from System.Windows.Controls.Control)
 <a href="#">Item</a>	Overloaded. <a href="#">ViewSources</a> コレクション内の指定された <a href="#">name</a>

		の <a href="#">RiaViewSource</a> の <a href="#">C1.Data.DataSource.ClientCollectionView</a> を取得します。
	<a href="#">Language</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Margin</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">MaxHeight</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">MaxWidth</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">MinHeight</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">MinWidth</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Name</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Opacity</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OpacityMask</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Padding</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">Parent</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Projection</a>	(Inherited from System.Windows.UIElement)
	<a href="#">RefreshInterval</a>	サーバーで行われた変更に基づいてデータをリフレッシュする自動 <a href="#">Refresh</a> 操作の 間隔を取得または設定します。
	<a href="#">RenderSize</a>	(Inherited from System.Windows.UIElement)
	<a href="#">RenderTransform</a>	(Inherited from System.Windows.UIElement)
	<a href="#">RenderTransformOrigin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Resources</a>	(Inherited from System.Windows.FrameworkElement)

	<a href="#">Style</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">TabIndex</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">TabNavigation</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">Tag</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Template</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">Triggers</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">UseLayoutRounding</a>	(Inherited from System.Windows.UIElement)
	<a href="#">VerticalAlignment</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">VerticalContentAlignment</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">ViewSources</a>	この <a href="#">C1DataSource</a> で（クエリーベースの）ビューを定義する <a href="#">RiaViewSource</a> オブジェクトのコレクションを取得します。
	<a href="#">Visibility</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Width</a>	(Inherited from System.Windows.FrameworkElement)


[Top](#)

## Protected Properties











	Name	Description
	<a href="#">DefaultStyleKey</a>	(Inherited from System.Windows.Controls.Control)

[Top](#)

## Public Methods







	Name	Description
	<a href="#">AddHandler</a>	(Inherited from System.Windows.UIElement)



















≡	<a href="#">ApplyTemplate</a>	(Inherited from System.Windows.Controls.Control)
≡	<a href="#">Arrange</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">CaptureMouse</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">ClearValue</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">FindName</a>	(Inherited from System.Windows.FrameworkElement)
≡	<a href="#">Focus</a>	(Inherited from System.Windows.Controls.Control)
≡	<a href="#">GetAnimationBaseValue</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">GetBindingExpression</a>	(Inherited from System.Windows.FrameworkElement)
≡ S	<a href="#">GetControlHandler</a>	指定された <a href="#">control</a> から C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandler 添付プロパティの値を取得します。
≡	<a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">InvalidateArrange</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">InvalidateMeasure</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">Load</a>	<a href="#">ViewSources</a> コレクション内のすべての <a href="#">RiaViewSource</a> オブジェクトをロードします。
≡	<a href="#">Measure</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">OnApplyTemplate</a>	(Inherited from System.Windows.FrameworkElement)
≡	<a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">Refresh</a>	<a href="#">ViewSources</a> コレクション内のすべての <a href="#">RiaViewSource</a> オブジェクトをリフレッシュします。

	<a href="#">RejectChanges</a>	<a href="#">DomainContext</a> 内のすべてのエンティティの変更を拒否します。
	<a href="#">ReleaseMouseCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">RemoveHandler</a>	(Inherited from System.Windows.UIElement)
	<a href="#">SaveChanges</a>	すべての変更をサーバーに送信します。
	<a href="#">SetBinding</a>	(Inherited from System.Windows.FrameworkElement)
 	<a href="#">SetControlHandler</a>	指定された <a href="#">control</a> に C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandler 添付プロパティの値を設定します。
	<a href="#">SetValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">TransformToVisual</a>	(Inherited from System.Windows.UIElement)
	<a href="#">UpdateLayout</a>	(Inherited from System.Windows.UIElement)

[Top](#)

## Protected Methods



















	Name	Description
	<a href="#">ArrangeOverride</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">GetTemplateChild</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">MeasureOverride</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnCreateAutomationPeer</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnDragEnter</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnDragLeave</a>	(Inherited from System.Windows.Controls.Control)













	<a href="#">OnDragOver</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnDrop</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnGotFocus</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnKeyDown</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnKeyUp</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnLostFocus</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnLostMouseCapture</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseEnter</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseLeave</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseLeftButtonDown</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseLeftButtonUp</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseMove</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseRightButtonDown</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseRightButtonUp</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseWheel</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnTextInput</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnTextInputStart</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnTextInputUpdate</a>	(Inherited from System.Windows.Controls.Control)

[Top](#)

## Public Events



	Name	Description
	<a href="#">BindingValidationError</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DataContextChanged</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DragEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">DragLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">DragOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Drop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">GotFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsEnabledChanged</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">KeyDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">KeyUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">LayoutUpdated</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Loaded</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">LostFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">LostMouseCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MediaCommand</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseLeftButtonDown</a>	(Inherited from System.Windows.UIElement)

	<a href="#">MouseLeftButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseRightButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseRightButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseWheel</a>	(Inherited from System.Windows.UIElement)
	<a href="#">SavedChanges</a>	送信操作の完了後に発生します。
	<a href="#">SavingChanges</a>	変更が送信される前に発生します。
	<a href="#">SizeChanged</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">TextInput</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TextInputStart</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TextInputUpdate</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Unloaded</a>	(Inherited from System.Windows.FrameworkElement)

[Top](#)

## See Also

### Reference

[C1DataSource Class](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

## C1DataSource Constructor

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : C1DataSource Constructor

[C1DataSource](#) クラスの新しいインスタンスを初期化します。

## Syntax

Visual Basic (Declaration)

```
Public Function New()
```

```
C#
```

```
public C1DataSource()
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

## Methods




[C1.Silverlight.Data.RiaServices Namespace](#) : [C1DataSource Class](#)

For a list of all members of this type, see [C1DataSource members](#).

## Public Methods














	Name	Description
≡	<a href="#">AddHandler</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">ApplyTemplate</a>	(Inherited from System.Windows.Controls.Control)
≡	<a href="#">Arrange</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">CaptureMouse</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">ClearValue</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">FindName</a>	(Inherited from System.Windows.FrameworkElement)
≡	<a href="#">Focus</a>	(Inherited from System.Windows.Controls.Control)
≡	<a href="#">GetAnimationBaseValue</a>	(Inherited from System.Windows.DependencyObject)












≡◆	<a href="#">GetBindingExpression</a>	(Inherited from System.Windows.FrameworkElement)
≡◆ S	<a href="#">GetControlHandler</a>	指定された <a href="#">control</a> から C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandler 添付プロパティの値を取得します。
≡◆	<a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
≡◆	<a href="#">InvalidateArrange</a>	(Inherited from System.Windows.UIElement)
≡◆	<a href="#">InvalidateMeasure</a>	(Inherited from System.Windows.UIElement)
≡◆	<a href="#">Load</a>	<a href="#">ViewSources</a> コレクション内のすべての <a href="#">RiaViewSource</a> オブジェクトをロードします。
≡◆	<a href="#">Measure</a>	(Inherited from System.Windows.UIElement)
≡◆	<a href="#">OnApplyTemplate</a>	(Inherited from System.Windows.FrameworkElement)
≡◆	<a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
≡◆	<a href="#">Refresh</a>	<a href="#">ViewSources</a> コレクション内のすべての <a href="#">RiaViewSource</a> オブジェクトをリフレッシュします。
≡◆	<a href="#">RejectChanges</a>	<a href="#">DomainContext</a> 内のすべてのエンティティの変更を拒否します。
≡◆	<a href="#">ReleaseMouseCapture</a>	(Inherited from System.Windows.UIElement)
≡◆	<a href="#">RemoveHandler</a>	(Inherited from System.Windows.UIElement)
≡◆	<a href="#">SaveChanges</a>	すべての変更をサーバーに送信します。
≡◆	<a href="#">SetBinding</a>	(Inherited from System.Windows.FrameworkElement)
≡◆ S	<a href="#">SetControlHandler</a>	指定された <a href="#">control</a> に C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandler

		添付プロパティの値を設定します。
≡ 	<a href="#">SetValue</a>	(Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">TransformToVisual</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">UpdateLayout</a>	(Inherited from System.Windows.UIElement)

[Top](#)

## Protected Methods

	Name	Description
	<a href="#">ArrangeOverride</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">GetTemplateChild</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">MeasureOverride</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnCreateAutomationPeer</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnDragEnter</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnDragLeave</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnDragOver</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnDrop</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnGotFocus</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnKeyDown</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnKeyUp</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnLostFocus</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnLostMouseCapture</a>	(Inherited from System.Windows.Controls.Control)

	<a href="#">OnMouseEnter</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseLeave</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseLeftButtonDown</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseLeftButtonUp</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseMove</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseRightButtonDown</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseRightButtonUp</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseWheel</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnTextInput</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnTextInputStart</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnTextInputUpdate</a>	(Inherited from System.Windows.Controls.Control)

[Top](#)

## See Also

### Reference

[C1DataSource Class](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

### GetControlHandler Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : GetControlHandler Method

プロパティ値の取得元のコントロール。

指定された [control](#) から C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandler 添付プロパティの値を取得します。

## Syntax

Visual Basic (Declaration)

```
Public Shared Function GetControlHandler( _
    ByVal control As System.Windows.DependencyObject _
) As BaseControlHandler
```

C#

```
public static BaseControlHandler GetControlHandler(
    System.Windows.DependencyObject control
)
```

## Parameters

*control*

プロパティ値の取得元のコントロール。

## Return Value

C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandler 添付プロパティの値。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

[ControlHandlerProperty Field](#)

## Load Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : Load Method

[ViewSources](#) コレクション内のすべての [RiaViewSource](#) オブジェクトをロードします。

## Syntax

Visual Basic (Declaration)

```
Public Sub Load()
```

C#

```
public void Load()
```

## Remarks

このメソッドは、[ViewSources](#) コレクションのすべての要素に対して [Load](#) を呼び出します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### Refresh Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : Refresh Method

[ViewSources](#) コレクション内のすべての [RiaViewSource](#) オブジェクトをリフレッシュします。

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Refresh()	
C#	
<b>public void</b> Refresh()	

## Remarks

このメソッドは、[ViewSources](#) コレクションのすべての要素に対して [C1.Data.DataSource.ClientViewSource.Refresh](#) を呼び出します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



[C1DataSource Class](#)

[C1DataSource Members](#)

## RejectChanges Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : RejectChanges Method

[DomainContext](#) 内のすべてのエンティティの変更を拒否します。

## Syntax

Visual Basic (Declaration)	
<code>Public Sub RejectChanges()</code>	
C#	
<code>public void RejectChanges()</code>	

## Remarks

[DomainContext](#) 内のすべてのエンティティの変更が拒否されます。これには、この [C1DataSource](#) を使用してロードされなかったエンティティも含まれます。これにより、[コレクションビュー](#) で保留中の追加または編集もキャンセルされます。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

## SaveChanges Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : SaveChanges Method

すべての変更をサーバーに送信します。

## Syntax

Visual Basic (Declaration)	
<code>Public Sub SaveChanges()</code>	

C#

```
public void SaveChanges()
```

## Remarks

### DomainContext

内で変更／追加／削除されたすべてのエンティティについて変更が送信されます。  
これには、この [C1DataSource](#) を使用してロードされなかったエンティティも含まれます。  
これにより、[コレクションビュー](#)で保留中の追加または編集もコミットされます。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### SetControlHandler Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : SetControlHandler Method

C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandler

添付プロパティの設定先のオブジェクト。

設定するプロパティ値。

指定された [control](#) に C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandler  
添付プロパティの値を設定します。

## Syntax

Visual Basic (Declaration)

```
Public Shared Sub SetControlHandler( _  
    ByVal control As System.Windows.DependencyObject, _  
    ByVal handler As BaseControlHandler _  
)
```

C#

```
public static void SetControlHandler(
```

```
System.Windows.DependencyObject control,  
BaseControlHandler handler  
)
```

### Parameters

*control*

C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandler  
添付プロパティの設定先のオブジェクト。

*handler*

設定するプロパティ値。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference





- [C1DataSource Class](#)
- [C1DataSource Members](#)
- [ControlHandlerProperty Field](#)

## Properties



















[C1.Silverlight.Data.RiaServices Namespace](#) : [C1DataSource Class](#)




















For a list of all members of this type, see [C1DataSource members](#).





## Public Properties

	Name	Description
	<a href="#">ActualHeight</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ActualWidth</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">AllowDrop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Background</a>	(Inherited from System.Windows.Controls.Control)

	<a href="#">BorderBrush</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">BorderThickness</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">CacheMode</a>	(Inherited from System.Windows.UIElement)
	<a href="#">CharacterSpacing</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">ClientCache</a>	この <a href="#">C1DataSource</a> がデータにアクセスするために使用する <a href="#">RiaClientCache</a> を取得または設定します。
	<a href="#">ClientScope</a>	この <a href="#">C1DataSource</a> が属する <a href="#">クライアントスコープ</a> を取得します。
	<a href="#">Clip</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Cursor</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DataContext</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DesiredSize</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">DomainContext</a>	<a href="#">ClientCache</a> の接続先の <b>System.ServiceModel.DomainServices.Client.DomainContext</b> を取得します。
	<a href="#">DomainContextTypeName</a>	デフォルトのクライアントキャッシュの取得に使用される <b>System.ServiceModel.DomainServices.Client.DomainContext</b> タイプの完全名を取得または設定します。
	<a href="#">Effect</a>	(Inherited from System.Windows.UIElement)
	<a href="#">FlowDirection</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">FontFamily</a>	(Inherited from System.Windows.Controls.Control)


	FontSize	(Inherited from System.Windows.Controls.Control)
	FontStretch	(Inherited from System.Windows.Controls.Control)
	FontStyle	(Inherited from System.Windows.Controls.Control)
	FontWeight	(Inherited from System.Windows.Controls.Control)
	Foreground	(Inherited from System.Windows.Controls.Control)
	Height	(Inherited from System.Windows.FrameworkElement)
	HorizontalAlignment	(Inherited from System.Windows.FrameworkElement)
	HorizontalContentAlignment	(Inherited from System.Windows.Controls.Control)
	IsEnabled	(Inherited from System.Windows.Controls.Control)
	IsHitTestVisible	(Inherited from System.Windows.UIElement)
	IsTabStop	(Inherited from System.Windows.Controls.Control)
	Item	Overloaded. <a href="#">ViewSources</a> コレクション内の指定された <a href="#">name</a> の <a href="#">RiaViewSource</a> の <a href="#">C1.Data.DataSource.ClientCollectionView</a> を取得します。
	Language	(Inherited from System.Windows.FrameworkElement)
	Margin	(Inherited from System.Windows.FrameworkElement)
	MaxHeight	(Inherited from System.Windows.FrameworkElement)
	MaxWidth	(Inherited from System.Windows.FrameworkElement)
	MinHeight	(Inherited from System.Windows.FrameworkElement)
	MinWidth	(Inherited from System.Windows.FrameworkElement)

	Name	(Inherited from System.Windows.FrameworkElement)
	Opacity	(Inherited from System.Windows.UIElement)
	OpacityMask	(Inherited from System.Windows.UIElement)
	Padding	(Inherited from System.Windows.Controls.Control)
	Parent	(Inherited from System.Windows.FrameworkElement)
	Projection	(Inherited from System.Windows.UIElement)
	RefreshInterval	サーバーで行われた変更に基づいてデータをリフレッシュする自動 <a href="#">Refresh</a> 操作の 間隔を取得または設定します。
	RenderSize	(Inherited from System.Windows.UIElement)
	RenderTransform	(Inherited from System.Windows.UIElement)
	RenderTransformOrigin	(Inherited from System.Windows.UIElement)
	Resources	(Inherited from System.Windows.FrameworkElement)
	Style	(Inherited from System.Windows.FrameworkElement)
	TabIndex	(Inherited from System.Windows.Controls.Control)
	TabNavigation	(Inherited from System.Windows.Controls.Control)
	Tag	(Inherited from System.Windows.FrameworkElement)
	Template	(Inherited from System.Windows.Controls.Control)
	Triggers	(Inherited from System.Windows.FrameworkElement)
	UseLayoutRounding	(Inherited from System.Windows.UIElement)
	VerticalAlignment	(Inherited from System.Windows.FrameworkElement)

	<a href="#">VerticalContentAlignment</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">ViewSources</a>	この <a href="#">C1DataSource</a> で（クエリベースの）ビューを定義する <a href="#">RiaViewSource</a> オブジェクトのコレクションを取得します。
	<a href="#">Visibility</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Width</a>	(Inherited from System.Windows.FrameworkElement)

[Top](#)

## Protected Properties

	Name	Description
	<a href="#">DefaultStyleKey</a>	(Inherited from System.Windows.Controls.Control)

[Top](#)

## See Also

### Reference

[C1DataSource Class](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

### ClientCache Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : ClientCache Property

この [C1DataSource](#) がデータにアクセスするために使用する [RiaClientCache](#) を取得または設定します。

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute(&gt; Public Property ClientCache As RiaClientCache</pre>	
C#	
<pre>[System.ComponentModel.DefaultValue()] public RiaClientCache ClientCache {get; set;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### ClientScope Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : ClientScope Property

この [C1DataSource](#) が属する [クライアントスコープ](#) を取得します。

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property ClientScope As RiaClientScope</code>	
C#	
<code>public RiaClientScope ClientScope {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### DomainContext Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : DomainContext Property

[ClientCache](#) の接続先の **System.ServiceModel.DomainServices.Client.DomainContext** を取得します。

## Syntax



Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute()&gt; Public ReadOnly Property DomainContext As System.ServiceModel.DomainServices.Client.DomainContext</pre>	
C#	
<pre>[System.ComponentModel.DefaultValue()] public System.ServiceModel.DomainServices.Client.DomainContext DomainContext {get;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### DomainContextTypeName Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : DomainContextTypeName Property

デフォルトのクライアントキャッシュの取得に使用される

**System.ServiceModel.DomainServices.Client.DomainContext**

タイプの完全名を取得または設定します。

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute()&gt; &lt;System.ComponentModel.CategoryAttribute("Common")&gt; Public Property DomainContextTypeName As System.String</pre>	
C#	
<pre>[System.ComponentModel.DefaultValue()] [System.ComponentModel.Category("Common")] public System.string DomainContextTypeName {get; set;}</pre>	

## Remarks

プロパティ値は、**System.Type.GetType(System.String)** メソッドの 'typeName' パラメータの要件を満たしている必要があります。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### Item Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : Item Property

[ViewSources](#) コレクション内の指定された [name](#) の [RiaViewSource](#) の [C1.Data.DataSource.ClientCollectionView](#) を取得します。

## Overload List

Overload	Description
<a href="#">Item(String)</a>	<a href="#">ViewSources</a> コレクション内の指定された <a href="#">name</a> の <a href="#">RiaViewSource</a> の <a href="#">C1.Data.DataSource.ClientCollectionView</a> を取得します。
<a href="#">Item(Int32)</a>	<a href="#">ViewSources</a> コレクション内の指定された <a href="#">index</a> にある <a href="#">RiaViewSource</a> の <a href="#">C1.Data.DataSource.ClientCollectionView</a> を取得します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

## Item(String) Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) > [Item Property](#) : Item(String) Property

[C1.Data.DataSource.ClientCollectionView](#) の取得元の [RiaViewSource](#) の名前。

[ViewSources](#) コレクション内の指定された [name](#) の [RiaViewSource](#) の

[C1.Data.DataSource.ClientCollectionView](#) を取得します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads ReadOnly Property Item( _     ByVal name As System.String _ ) As ClientCollectionView</pre>	
C#	
<pre>public ClientCollectionView Item(     System.string name ) {get;}</pre>	

### Parameters

*name*

[C1.Data.DataSource.ClientCollectionView](#) の取得元の [RiaViewSource](#) の名前。

### Property Value

[RiaViewSource](#) の [C1.Data.DataSource.ClientCollectionView](#)。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)  
[C1DataSource Members](#)  
[Overload List](#)

## Item(Int32) Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) > [Item Property](#) : Item(Int32) Property

[C1.Data.DataSource.ClientCollectionView](#) の取得元の [RiaViewSource](#) のインデックス。

[ViewSources](#) コレクション内の指定された [index](#) にある [RiaViewSource](#) の [C1.Data.DataSource.ClientCollectionView](#) を取得します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads ReadOnly Property Item( _     ByVal index As System.Integer _ ) As ClientCollectionView</pre>	
C#	
<pre>public ClientCollectionView Item(     System.int index ) {get;}</pre>	

### Parameters

*index*

[C1.Data.DataSource.ClientCollectionView](#) の取得元の [RiaViewSource](#) のインデックス。

### Property Value

[RiaViewSource](#) の [C1.Data.DataSource.ClientCollectionView](#)。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)  
[C1DataSource Members](#)  
[Overload List](#)

### RefreshInterval Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : RefreshInterval Property

サーバーで行われた変更に基づいてデータをリフレッシュする自動 [Refresh](#) 操作の間隔を取得または設定します。

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.CategoryAttribute("Common")>
<System.ComponentModel.DefaultValueAttribute()>
<System.ComponentModel.DescriptionAttribute("Gets or sets the interval
between automatic Refresh operations.")>
Public Property RefreshInterval As System.TimeSpan
```

C#

```
[System.ComponentModel.Category("Common")]
[System.ComponentModel.DefaultValue()]
[System.ComponentModel.Description("Gets or sets the interval between
automatic Refresh operations.")]
public System.TimeSpan RefreshInterval {get; set;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### ViewSources Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : ViewSources Property

この [C1DataSource](#) で（クエリーベースの）ビューを定義する [RiaViewSource](#) オブジェクトのコレクションを取得します。

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.CategoryAttribute("Common")>
Public ReadOnly Property ViewSources As RiaViewSourceCollection
```

C#

```
[System.ComponentModel.Category("Common")]  
public RiaViewSourceCollection ViewSources {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

## Fields

[C1.Silverlight.Data.RiaServices Namespace](#) : [C1DataSource Class](#)

For a list of all members of this type, see [C1DataSource members](#).

## Public Fields

	Name	Description
 <b>S</b>	<a href="#">ControlHandlerProperty</a>	C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandler 添付プロパティを識別します。

[Top](#)

## See Also

### Reference

[C1DataSource Class](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

## ControlHandlerProperty Field

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : ControlHandlerProperty Field

C1.Silverlight.Data.RiaServices.C1DataSource.ControlHandler 添付プロパティを識別します。

## Syntax

Visual Basic (Declaration)

<b>Public Shared ReadOnly</b> ControlHandlerProperty <b>As</b> System.Windows.DependencyProperty	
C#	
<b>public static readonly</b> System.Windows.DependencyProperty ControlHandlerProperty	

## Remarks

この添付プロパティを使用して、コントロールに[コントロールハンドラ](#)を接続します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)






[C1DataSource Members](#)




















## Events

[C1.Silverlight.Data.RiaServices Namespace](#) : [C1DataSource Class](#)

For a list of all members of this type, see [C1DataSource members](#).

## Public Events

	Name	Description
	<a href="#">BindingValidationError</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DataContextChanged</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DragEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">DragLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">DragOver</a>	(Inherited from System.Windows.UIElement)

	Drop	(Inherited from System.Windows.UIElement)
	GotFocus	(Inherited from System.Windows.UIElement)
	IsEnabledChanged	(Inherited from System.Windows.Controls.Control)
	KeyDown	(Inherited from System.Windows.UIElement)
	KeyUp	(Inherited from System.Windows.UIElement)
	LayoutUpdated	(Inherited from System.Windows.FrameworkElement)
	Loaded	(Inherited from System.Windows.FrameworkElement)
	LostFocus	(Inherited from System.Windows.UIElement)
	LostMouseCapture	(Inherited from System.Windows.UIElement)
	MediaCommand	(Inherited from System.Windows.UIElement)
	MouseEnter	(Inherited from System.Windows.UIElement)
	MouseLeave	(Inherited from System.Windows.UIElement)
	MouseLeftButtonDown	(Inherited from System.Windows.UIElement)
	MouseLeftButtonUp	(Inherited from System.Windows.UIElement)
	MouseMove	(Inherited from System.Windows.UIElement)
	MouseRightButtonDown	(Inherited from System.Windows.UIElement)
	MouseRightButtonUp	(Inherited from System.Windows.UIElement)
	MouseWheel	(Inherited from System.Windows.UIElement)
	SavedChanges	送信操作の完了後に発生します。



	<a href="#">SavingChanges</a>	変更が送信される前に発生します。
	<a href="#">SizeChanged</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">TextInput</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TextInputStart</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TextInputUpdate</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Unloaded</a>	(Inherited from System.Windows.FrameworkElement)

[Top](#)

## See Also

### Reference

[C1DataSource Class](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

### SavedChanges Event

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : SavedChanges Event

送信操作の完了後に発生します。

## Syntax

Visual Basic (Declaration)

```
Public Event SavedChanges As System.EventHandler(Of SavedChangesEventArgs)
```

C#

```
public event System.EventHandler<SavedChangesEventArgs> SavedChanges
```

## Event Data

The event handler receives an argument of type [SavedChangesEventArgs](#) containing data related to this event. The following **SavedChangesEventArgs** properties provide information specific to this event.

Property	Description
----------	-------------

<a href="#">Error</a>	Gets a value showing the error that occurred during a save operation.
<a href="#">HasError</a>	Gets a value indicating whether the save operation has failed. If true, inspect the <a href="#">Error</a> property for details.
<a href="#">IsErrorHandled</a>	Gets a value indicating whether the error has been marked as handled by calling <a href="#">MarkErrorAsHandled</a> .

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)  
[C1DataSource Members](#)  
[SaveChanges Method](#)

### SavingChanges Event

[C1.Silverlight.Data.RiaServices Namespace](#) > [C1DataSource Class](#) : SavingChanges Event

変更が送信される前に発生します。

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> SavingChanges <b>As</b> System.EventHandler(Of CancelEventArgs)	
C#	
<b>public event</b> System.EventHandler<CancelEventArgs> SavingChanges	

## Event Data

The event handler receives an argument of type System.ComponentModel.CancelEventArgs containing data related to this event. The following **CancelEventArgs** properties provide information specific to this event.

Property	Description
----------	-------------

<b>Cancel</b>	
---------------	--

## Remarks

このイベントは [SaveChanges](#) メソッドから発生され、この操作が開始される前にハンドラが操作をキャンセルできるようにします。ハンドラが

**System.ComponentModel.CancelEventArgs.Cancel** を True に

設定すると、この操作は中断され、これに続く SavedChanges イベント は発生しません。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

[SaveChanges Method](#)

## RiaClientCache

[C1.Silverlight.Data.RiaServices Namespace](#) : RiaClientCache Class

RIA サービスに固有のクライアント側キャッシュを表します。

## Object Model

RiaClientCache

## Syntax

Visual Basic (Declaration)	
<pre><b>Public Class</b> RiaClientCache     <b>Inherits</b> C1.Data.ClientCacheBase</pre>	
C#	
<pre><b>public class</b> RiaClientCache : <a href="#">C1.Data.ClientCacheBase</a></pre>	

## Remarks

通常は、アプリケーションの起動時にこのクラスの 1 つのインスタンスが作成され、アプリケーションの全ライフタイムにわたって存在します。

一方、各ユーザーコントロールは、[CreateScope](#) メソッドを呼び出すことで作成される [RiaClientScope](#) を使用してデータにアクセスします。

## Inheritance Hierarchy

System.Object  
    [C1.Data.ClientCacheBase](#)  
        **C1.Silverlight.Data.RiaServices.RiaClientCache**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientCache Members](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)

### Overview

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaClientCache Class](#)

RIA サービスに固有のクライアント側キャッシュを表します。

## Object Model

RiaClientCache

## Syntax

Visual Basic (Declaration)	
<pre>Public Class RiaClientCache     Inherits C1.Data.ClientCacheBase</pre>	
C#	
<pre>public class RiaClientCache : C1.Data.ClientCacheBase</pre>	

## Remarks

通常は、アプリケーションの起動時にこのクラスの 1 つのインスタンスが作成され、アプリケーションの全ライフタイムにわたって存在します。

一方、各ユーザーコントロールは、[CreateScope](#) メソッドを呼び出すことで作成される [RiaClientScope](#) を使用してデータにアクセスします。

## Inheritance Hierarchy

System.Object  
    [C1.Data.ClientCacheBase](#)  
        **C1.Silverlight.Data.RiaServices.RiaClientCache**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientCache Members](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)


## Members

[Properties](#) [Methods](#)

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaClientCache Class](#)


The following tables list the members exposed by [RiaClientCache](#).

## Public Constructors

	Name	Description
	<a href="#">RiaClientCache Constructor</a>	<a href="#">RiaClientCache</a> クラスの新しいインスタンスを初期化します。

[Top](#)

## Public Properties

	Name	Description
	<a href="#">DomainContext</a>	この <a href="#">RiaClientCache</a> がデータにアクセスするために使用する

t	<b>System.ServiceModel.DomainServices.Client.DomainContext。</b>
 IsLoading	この <a href="#">RiaClientCache</a> が現在サーバーからデータを取得しているかどうかを示す値を取得します。 。

[Top](#)

## Public Methods

	Name	Description
⇒💎	<a href="#">BulkChanges</a>	エンティティに対する大規模な変更をグループ化するため、およびエンティティの状態を手作業で明示的に変更できるようにするために使用されます。 (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
⇒💎	<a href="#">CleanupCache</a>	未使用のメモリを強制的に解放し、未使用のエンティティをコンテキストから強制的にデタッチします。 通常は自動的に行われるため、プログラマがコード内でこのメソッドを呼び出す必要はほとんどありません。 (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
⇒💎	<a href="#">Clear</a>	クライアント側のキャッシュを完全にクリアします。現時点から後のクエリーがサーバーから必ず新しいデータを取得するようにしたい場合に、このメソッドを呼び出してください。 (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
⇒💎	<a href="#">CreateScope</a>	データアクセスの範囲を定義する <a href="#">RiaClientScope</a> を作成します。
⇒💎	<a href="#">CreateTransaction</a>	トランザクションスコープ内で行われた変更を簡単にキャンセルできる <a href="#">C1.Data.Transactions.ClientTransaction</a> を作成します。 (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
⇒💎 S	<a href="#">GetDefault</a>	指定された <a href="#">contextType</a> のデフォルトの <a href="#">RiaClientCache</a> を返します。
⇒💎	<a href="#">Refresh</a>	この ClientCacheBase に接続されているすべての C1DataSource コントロールのデータを更新します。 (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
⇒💎	<a href="#">Register</a>	Overloaded. 特定の <b>コンテキストタイプ</b> の C1DataSource

<b>S</b>	<b>Context</b>	コントロールのデフォルトとして、 <a href="#">DomainContext</a> を登録します。
⇒💎	<b>RejectChanges</b>	この <a href="#">C1.Data.ClientCacheBase</a> に対する保留中のすべての変更を元に戻します。 <b>System.ServiceModel.DomainServices.Client.DomainContext.RejectChanges</b> の代わりにこのメソッドを呼び出すことをお勧めします。(Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
⇒💎	<b>SaveChanges</b>	すべての変更をサーバーに永続化します。 <b>System.Data.Objects.ObjectContext.SaveChanges()</b> の代わりにこのメソッドを呼び出すことをお勧めします。(Inherited from <a href="#">C1.Data.ClientCacheBase</a> )

[Top](#)

## See Also

### Reference

[RiaClientCache Class](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

## RiaClientCache Constructor

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientCache Class](#) : RiaClientCache Constructor

データへのアクセスに使用される [DomainContext](#)。

[RiaClientCache](#) クラスの新しいインスタンスを初期化します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Function New( _     ByVal baseContext As System.ServiceModel.DomainServices.Client.DomainContext _ )</pre>	
C#	
<pre>public RiaClientCache(     System.ServiceModel.DomainServices.Client.DomainContext baseContext )</pre>	

### Parameters

*baseContext*

データへのアクセスに使用される [DomainContext](#)。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientCache Class](#)

[RiaClientCache Members](#)

### Methods

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaClientCache Class](#)

For a list of all members of this type, see [RiaClientCache members](#).

## Public Methods

	Name	Description
⇒💎	<a href="#">BulkChanges</a>	エンティティに対する大規模な変更をグループ化するため、およびエンティティの状態を手作業で明示的に変更できるようにするために使用されます。 (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
⇒💎	<a href="#">CleanupCache</a>	未使用のメモリを強制的に解放し、未使用のエンティティをコンテキストから強制的にデタッチします。 通常は自動的に行われるため、プログラマがコード内でこのメソッドを呼び出す必要はほとんどありません。 (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
⇒💎	<a href="#">Clear</a>	クライアント側のキャッシュを完全にクリアします。現時点から後のクエリーがサーバーから必ず新しいデータを取得するようにしたい場合に、このメソッドを呼び出してください。 (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
⇒💎	<a href="#">CreateScope</a>	データアクセスの範囲を定義する <a href="#">RiaClientScope</a> を作成します。
⇒💎	<a href="#">CreateTransaction</a>	トランザクションスコープ内で行われた変更を簡単にキャンセルできる <a href="#">C1.Data.Transactions.ClientTransaction</a> を作成します。 (Inherited from



	n	<a href="#">C1.Data.ClientCacheBase</a> )
⇒ S	<a href="#">GetDefault</a>	指定された <a href="#">contextType</a> のデフォルトの <a href="#">RiaClientCache</a> を返します。
⇒	<a href="#">Refresh</a>	この ClientCacheBase に接続されているすべての C1DataSource コントロールのデータを更新します。 (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
⇒ S	<a href="#">RegisterContext</a>	Overloaded. 特定のコンテキストタイプの C1DataSource コントロールのデフォルトとして、 <a href="#">DomainContext</a> を登録します。
⇒	<a href="#">RejectChanges</a>	この <a href="#">C1.Data.ClientCacheBase</a> に対する保留中のすべての変更を元に戻します。 <b>System.ServiceModel.DomainServices.Client.DomainContext.RejectChanges</b> の代わりにこのメソッドを呼び出すことをお勧めします。 (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )
⇒	<a href="#">SaveChanges</a>	すべての変更をサーバーに永続化します。 <b>System.Data.Objects.ObjectContext.SaveChanges()</b> の代わりにこのメソッドを呼び出すことをお勧めします。 (Inherited from <a href="#">C1.Data.ClientCacheBase</a> )

[Top](#)

## See Also

### Reference

[RiaClientCache Class](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

### CreateScope Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientCache Class](#) : CreateScope Method

データアクセスの範囲を定義する [RiaClientScope](#) を作成します。

## Syntax

Visual Basic (Declaration)	
<b>Public Shadows Function</b> CreateScope() <b>As</b> <a href="#">RiaClientScope</a>	
C#	

```
public new RiaClientScope CreateScope()
```

### Return Value

新しいクライアントスコープ。

## Remarks

通常、各ユーザーコントロールは、[RiaClientScope](#) を作成し、それを使用してエンティティにアクセスします。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientCache Class](#)

[RiaClientCache Members](#)

[RiaClientScope Class](#)

### GetDefault Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientCache Class](#) : GetDefault Method

デフォルトの [RiaClientCache](#) を取得するための [DomainContext](#) のサブクラス。

指定された [contextType](#) のデフォルトの [RiaClientCache](#) を返します。

## Syntax

Visual Basic (Declaration)

```
Public Shared Function GetDefault( _  
    ByVal contextType As System.Type _  
) As RiaClientCache
```

C#

```
public static RiaClientCache GetDefault(  
    System.Type contextType  
)
```

### Parameters

*contextType*

デフォルトの [RiaClientCache](#) を取得するための [DomainContext](#) のサブクラス。

## Return Value

指定された[contextType](#)のデフォルトの [RiaClientCache](#)。

## Remarks

指定された[contextType](#)の [RiaClientCache](#) がまだ存在していない場合は作成します。  
存在する場合は、既存のインスタンスを返します。

これは、指定された [C1DataSource.DomainContextTypeName](#) で [C1DataSource](#) によって使用されるクライアントキャッシュと同じデフォルトのクライアントキャッシュです。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientCache Class](#)

[RiaClientCache Members](#)

### RegisterContext Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientCache Class](#) : RegisterContext Method

特定のコンテキストタイプの [C1DataSource](#) コントロールのデフォルトとして、[DomainContext](#) を登録します。

## Overload List

Overload	Description
<a href="#">RegisterContext(DomainContext,Type)</a>	特定の <a href="#">contextType</a> の <a href="#">C1DataSource</a> コントロールのデフォルトとして、 <a href="#">DomainContext</a> を登録します。
<a href="#">RegisterContext(DomainContext)</a>	<a href="#">C1DataSource</a> コントロールのデフォルトとして

	<a href="#">DomainContext</a> を登録します。
--	---------------------------------------

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientCache Class](#)

[RiaClientCache Members](#)

RegisterContext(DomainContext,Type) Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientCache Class](#) > [RegisterContext Method](#) :

RegisterContext(DomainContext,Type) Method

デフォルトとして設定する [DomainContext](#)。

登録する [context](#) に対するタイプ ([DomainContext](#) から派生)。

特定の[contextType](#)の C1DataSource コントロールのデフォルトとして、[DomainContext](#) を登録します。

## Syntax

Visual Basic (Declaration)

```
Public Overloads Shared Function RegisterContext( _
    ByVal context As System.ServiceModel.DomainServices.Client.DomainContext,
    -
    ByVal contextType As System.Type _
) As System.IDisposable
```

C#

```
public static System.IDisposable RegisterContext(
    System.ServiceModel.DomainServices.Client.DomainContext context,
    System.Type contextType
)
```

### Parameters

*context*

デフォルトとして設定する [DomainContext](#)。

*contextType*

登録する [context](#) に対するタイプ ([DomainContext](#) から派生)。

## Return Value

[context](#) を登録解除するための **System.IDisposable**。

## Exceptions

Exception	Description
<b>System.InvalidOperationException</b>	指定された <a href="#">contextType</a> に対して別のコンテキストが既に登録されています。

## Remarks

このメソッドは、C1DataSource コントロールで使用するデフォルトの [DomainContext](#) をカスタマイズする場合に使用します。カスタム [DomainContext](#) は、起動時に、どの C1DataSource インスタンスが作成されるより前に登録します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientCache Class](#)  
[RiaClientCache Members](#)  
[Overload List](#)  
[GetDefault Method](#)

RegisterContext(DomainContext) Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientCache Class](#) > [RegisterContext Method](#) : RegisterContext(DomainContext) Method

デフォルトとして設定する [DomainContext](#)。

C1DataSource コントロールのデフォルトとして [DomainContext](#) を登録します。

# Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Shared Function RegisterContext( _     ByVal context As System.ServiceModel.DomainServices.Client.DomainContext _ ) As System.IDisposable</pre>	
C#	
<pre>public static System.IDisposable RegisterContext(     System.ServiceModel.DomainServices.Client.DomainContext context )</pre>	

## Parameters

*context*

デフォルトとして設定する [DomainContext](#)。

## Return Value

[context](#) を登録解除するための **System.IDisposable**。

# Exceptions

Exception	Description
<b>System.InvalidOperationException</b>	別のコンテキストが既に登録されています。

## Remarks

このメソッドは、C1DataSource コントロールで使用するデフォルトの [DomainContext](#) をカスタマイズする場合に使用します。カスタム [DomainContext](#) は、起動時に、どの C1DataSource インスタンスが作成されるより前に登録します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference



[RiaClientCache Class](#)  
[RiaClientCache Members](#)  
[Overload List](#)  
[GetDefault Method](#)

## Properties

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaClientCache Class](#)

For a list of all members of this type, see [RiaClientCache members](#).

## Public Properties

	Name	Description
	<a href="#">DomainContext</a> t	この <a href="#">RiaClientCache</a> がデータにアクセスするために使用する <b>System.ServiceModel.DomainServices.Client.DomainContext</b> 。
	<a href="#">IsLoading</a>	この <a href="#">RiaClientCache</a> が現在サーバーからデータを取得しているかどうかを示す値を取得します。 。

[Top](#)

## See Also

### Reference

[RiaClientCache Class](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)

### DomainContext Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientCache Class](#) : DomainContext Property

この [RiaClientCache](#) がデータにアクセスするために使用する **System.ServiceModel.DomainServices.Client.DomainContext**。

## Syntax

Visual Basic (Declaration)	
<pre>Public ReadOnly Property DomainContext As System.ServiceModel.DomainServices.Client.DomainContext</pre>	
C#	

```
public System.ServiceModel.DomainServices.Client.DomainContext DomainContext
{get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientCache Class](#)

[RiaClientCache Members](#)

### IsLoading Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientCache Class](#) : IsLoading Property

この [RiaClientCache](#)

が現在サーバーからデータを取得しているかどうかを示す値を取得します。

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property IsLoading As System.Boolean
```

C#

```
public System.bool IsLoading {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientCache Class](#)

[RiaClientCache Members](#)



# RiaClientScope

[C1.Silverlight.Data.RiaServices Namespace](#) : RiaClientScope Class

## Object Model

RiaClientScope

## Syntax

Visual Basic (Declaration)	
<pre>Public Class RiaClientScope     Inherits C1.Data.ClientScope</pre>	
C#	
<pre>public class RiaClientScope : C1.Data.ClientScope</pre>	

## Inheritance Hierarchy

System.Object

[C1.Data.ClientScope](#)

**C1.Silverlight.Data.RiaServices.RiaClientScope**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientScope Members](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

### Overview

[C1.Silverlight.Data.RiaServices Namespace](#) : RiaClientScope Class

## Object Model

RiaClientScope

## Syntax

Visual Basic (Declaration)	
<pre>Public Class RiaClientScope     Inherits C1.Data.ClientScope</pre>	
C#	
<pre>public class RiaClientScope : C1.Data.ClientScope</pre>	

## Inheritance Hierarchy

System.Object  
[C1.Data.ClientScope](#)  
**C1.Silverlight.Data.RiaServices.RiaClientScope**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientScope Members](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)


## Members

[Properties](#) [Methods](#)

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaClientScope Class](#)

The following tables list the members exposed by [RiaClientScope](#).

## Public Constructors

	Name	Description
	<a href="#">RiaClientScope Constructor</a>	

[Top](#)





## Public Properties

	Name	Description
--	------	-------------

	ClientCache	
---	-------------	--

[Top](#)

## Public Methods

	Name	Description
≡ 	<a href="#">AddRef</a>	Overloaded. (Inherited from <a href="#">C1.Data.ClientScope</a> )
≡ 	<a href="#">Dispose</a>	(Inherited from <a href="#">C1.Data.ClientScope</a> )
≡ 	<a href="#">GetItems</a>	Overloaded.
≡ 	<a href="#">Release</a>	Overloaded. (Inherited from <a href="#">C1.Data.ClientScope</a> )

[Top](#)

## See Also

### Reference

[RiaClientScope Class](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

## RiaClientScope Constructor

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientScope Class](#) : RiaClientScope Constructor

## Syntax

Visual Basic (Declaration)	
<pre>Public Function New( _     ByVal <i>clientCache</i> As RiaClientCache _ )</pre>	
C#	
<pre>public RiaClientScope(     RiaClientCache <i>clientCache</i> )</pre>	

### Parameters

*clientCache*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientScope Class](#)

[RiaClientScope Members](#)

## Methods

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaClientScope Class](#)

For a list of all members of this type, see [RiaClientScope members](#).

## Public Methods

	Name	Description
≡	<a href="#">AddRef</a>	Overloaded. (Inherited from <a href="#">C1.Data.ClientScope</a> )
≡	<a href="#">Dispose</a>	(Inherited from <a href="#">C1.Data.ClientScope</a> )
≡	<a href="#">GetItems</a>	Overloaded.
≡	<a href="#">Release</a>	Overloaded. (Inherited from <a href="#">C1.Data.ClientScope</a> )

[Top](#)

## See Also

### Reference

[RiaClientScope Class](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

### GetItems Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientScope Class](#) : [GetItems Method](#)

## Overload List

Overload	Description
<a href="#">GetItems&lt;T&gt;(String, IDictionary&lt;String, Object&gt;)</a>	
<a href="#">GetItems&lt;T&gt;(EntityQuery&lt;T&gt;)</a>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientScope Class](#)

[RiaClientScope Members](#)

[GetItems<T>\(String, IDictionary<String, Object>\) Method](#)

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientScope Class](#) > [GetItems Method](#) :

[GetItems<T>\(String, IDictionary<String, Object>\) Method](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function GetItems(Of T As System.ServiceModel.DomainServices.Client.Entity)( _     ByVal queryName As System.String, _     Optional ByVal parameters As System.Collections.Generic.IDictionary(Of String,Object) _ ) As ClientView(Of T)</pre>	
C#	
<pre>public ClientView&lt;T&gt; GetItems&lt;T&gt;(     System.string queryName,     System.Collections.Generic.IDictionary&lt;string,object&gt; parameters ) where T: System.ServiceModel.DomainServices.Client.Entity</pre>	

### Parameters

*queryName*

*parameters*

## Type Parameters

*T*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[RiaClientScope Class](#)  
[RiaClientScope Members](#)  
[Overload List](#)

GetItems<T>(EntityQuery<T>) Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientScope Class](#) > [GetItems Method](#) :

GetItems<T>(EntityQuery<T>) Method

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function GetItems(Of T As System.ServiceModel.DomainServices.Client.Entity)( _     ByVal query As System.ServiceModel.DomainServices.Client.EntityQuery(Of T)     _ ) As ClientView(Of T)</pre>	
C#	
<pre>public ClientView&lt;T&gt; GetItems&lt;T&gt;(     System.ServiceModel.DomainServices.Client.EntityQuery&lt;T&gt; query ) where T: System.ServiceModel.DomainServices.Client.Entity</pre>	

## Parameters

*query*

## Type Parameters

*T*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientScope Class](#)  
[RiaClientScope Members](#)  
[Overload List](#)

## Properties

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaClientScope Class](#)

For a list of all members of this type, see [RiaClientScope members](#).

## Public Properties

	Name	Description
	<a href="#">ClientCache</a>	

[Top](#)

## See Also

### Reference

[RiaClientScope Class](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)

### ClientCache Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaClientScope Class](#) : ClientCache Property

## Syntax

Visual Basic (Declaration)	
<b>Public Shadows ReadOnly Property</b> ClientCache <b>As</b> <a href="#">RiaClientCache</a>	
C#	
<b>public new</b> <a href="#">RiaClientCache</a> ClientCache { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaClientScope Class](#)

[RiaClientScope Members](#)

## RiaServicesExtensions

[C1.Silverlight.Data.RiaServices Namespace](#) : RiaServicesExtensions Class

## Object Model

RiaServicesExtensions

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public MustInherit NotInheritable Class RiaServicesExtensions
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static class RiaServicesExtensions
```

## Inheritance Hierarchy

System.Object

**C1.Silverlight.Data.RiaServices.RiaServicesExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



## Overview

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaServicesExtensions Class](#)

## Object Model

[RiaServicesExtensions](#)

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public MustInherit NotInheritable Class RiaServicesExtensions</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static class RiaServicesExtensions</pre>	

## Inheritance Hierarchy

System.Object  
    **C1.Silverlight.Data.RiaServices.RiaServicesExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaServicesExtensions Members](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)

## Members

[Methods](#)

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaServicesExtensions Class](#)

The following tables list the members exposed by [RiaServicesExtensions](#).

# Public Methods

	Name	Description
	AsLive<T>	

[Top](#)

## See Also

### Reference

[RiaServicesExtensions Class](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)

## Methods

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaServicesExtensions Class](#)

For a list of all members of this type, see [RiaServicesExtensions members](#).

# Public Methods

	Name	Description
	AsLive<T>	

[Top](#)

## See Also

### Reference

[RiaServicesExtensions Class](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)

### AsLive<T> Method

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaServicesExtensions Class](#) : AsLive<T> Method

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Function AsLive(Of T As System.ServiceModel.DomainServices.Client.Entity)( _</pre>	

```

    ByVal entities As
System.ServiceModel.DomainServices.Client.EntityCollection(Of T), _
    ByVal clientCache As RiaClientCache _
) As View(Of T)

```

C#

```

[System.Runtime.CompilerServices.Extension()]
public static View<T> AsLive<T>(
    System.ServiceModel.DomainServices.Client.EntityCollection<T> entities,
    RiaClientCache clientCache
)
where T: System.ServiceModel.DomainServices.Client.Entity

```

## Parameters

*entities*

*clientCache*

## Type Parameters

*T*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[RiaServicesExtensions Class](#)

[RiaServicesExtensions Members](#)

## RiaViewSource

[C1.Silverlight.Data.RiaServices Namespace](#) : RiaViewSource Class

[C1.Data.DataSource.ClientViewSource](#) クラスの RIA サービス固有バージョン。

## Object Model

RiaViewSource

## Syntax

Visual Basic (Declaration)	
<pre>Public Class RiaViewSource     Inherits C1.Data.DataSource.ClientViewSource</pre>	
C#	
<pre>public class RiaViewSource : C1.Data.DataSource.ClientViewSource</pre>	

## Remarks

データをロードするには、**System.ServiceModel.DomainServices.Client.EntityQuery{T}** を返す **System.ServiceModel.DomainServices.Client.DomainContext** 内のメソッドの名前を [QueryName](#) に設定します。 [Parameters](#) を使用して、クエリーパラメータを指定します。

## Inheritance Hierarchy

```
System.Object
  System.Windows.DependencyObject
    C1.Data.DataSource.ClientViewSource
      C1.Silverlight.Data.RiaServices.RiaViewSource
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaViewSource Members](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)

### Overview

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaViewSource Class](#)

[C1.Data.DataSource.ClientViewSource](#) クラスの RIA サービス固有バージョン。

## Object Model

RiaViewSource

## Syntax

Visual Basic (Declaration)	
<pre>Public Class RiaViewSource     Inherits C1.Data.DataSource.ClientViewSource</pre>	
C#	
<pre>public class RiaViewSource : C1.Data.DataSource.ClientViewSource</pre>	

## Remarks

データをロードするには、**System.ServiceModel.DomainServices.Client.EntityQuery{T}** を返す **System.ServiceModel.DomainServices.Client.DomainContext** 内のメソッドの名前を [QueryName](#) に設定します。 [Parameters](#) を使用して、クエリーパラメータを指定します。

## Inheritance Hierarchy

```
System.Object
    System.Windows.DependencyObject
        C1.Data.DataSource.ClientViewSource
            C1.Silverlight.Data.RiaServices.RiaViewSource
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaViewSource Members](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)

## Members

[Properties](#) [Methods](#) [Events](#)

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaViewSource Class](#)

The following tables list the members exposed by [RiaViewSource](#).

## Public Constructors

Name	Description
------	-------------

	<b>RiaViewSource Constructor</b>	<a href="#">RiaViewSource</a> クラスの新しいインスタンスを初期化します。
---	----------------------------------	---

[Top](#)

## Public Properties


	Name	Description
	<b>AutoLoad</b>	起動時や、 <a href="#">C1.Data.DataSource.ClientViewSource</a> によって作成されたクエリーが影響を受ける変更が発生したときに、 <a href="#">Load</a> が自動的に呼び出されるかどうかを示す値を取得または設定します。 デフォルトは True です。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<b>BaseView</b>	<a href="#">C1.Data.DataSource.ClientViewSource</a> がクエリーを作成するための基礎として使用する <a href="#">C1.Data.ClientView&lt;T&gt;</a> のインスタンスを取得または設定します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<b>CacheTimeout</b>	仮想モードでロードされたエンティティが、必要かどうかのチェックがされることなくキャッシュに維持される時間を取得または設定します。 <a href="#">CacheTimeout</a> よりも長い時間、エンティティが使用されることも必要と判断されることもなかった場合、 <a href="#">C1.Data.DataSource.ClientViewSource</a> はそのエンティティをキャッシュから削除できます。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<b>CurrentClientView</b>	エンティティのロードに使用される現在の <a href="#">クライアントビュー</a> 、または <a href="#">仮想モード</a> では null を取得します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<b>DataView</b>	最後のロード操作の結果であるエンティティの現在のビューを取得します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
	<b>Dispatcher</b>	(Inherited from System.Windows.DependencyObject)
	<b>FilterDescriptor</b>	ロード実行時に使用される <a href="#">C1.Data.DataSource.FilterDescriptor</a>

 <b>ptors</b>	オブジェクトのコレクションを取得します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>FilterOperator</b>	フィルタコレクション内の <a href="#">FilterDescriptors</a> どうしを結合するために使用される論理演算子を取得または設定します。デフォルト値は、 <a href="#">C1.Data.DataSource.FilterDescriptorLogicalOperator.And</a> です。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>GroupDescriptors</b>	読み込んだエンティティをいくつかのグループに組織化するために使用される <a href="#">C1.Data.DataSource.GroupDescriptor</a> オブジェクトのコレクションを取得します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>IsLoadingData</b>	<a href="#">C1.Data.DataSource.ClientViewSource</a> が現在データをロード中かどうかを示す値を取得します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>LoadCommand</b>	この <a href="#">C1.Data.DataSource.ClientViewSource</a> で <b>Load</b> を呼び出す <b>System.Windows.Input.ICommand</b> を取得します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>LoadDelay</b>	自動データロード操作が開始されるまでの遅延時間を取得または設定します。自動ロードを促す変更が発生した時点から 結果としての <a href="#">Load</a> が開始される時点までが遅延時間です。デフォルトの遅延時間は、25 ミリ秒です。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>LoadSize</b>	<a href="#">Load</a> が実行されるたびにロードされる項目の最大数を取得または設定します。0 の場合は、要求されたエンティティがすべてロードされます。デフォルトは0 です。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>MoveToFirstOnLoad</b>	<a href="#">Load</a> 操作完了後の最初の項目を現在の項目にすることを示す値を取得または設定します (現在の項目が他の方法で設定されていない場合)。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>Name</b>	Overridden. <a href="#">C1DataSource.ViewSources</a> コレクション内で参照される、この <a href="#">RiaViewSource</a> の名前を取得します。これは、 <a href="#">QueryName</a>

		によって決定されますが、 <a href="#">NameOverride</a> でオーバーライドできます。
 <a href="#">NameOverride</a>		<a href="#">Name</a> プロパティの値をオーバーライドする値を取得または設定します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <a href="#">PageSize</a>		<a href="#">DataView</a> の各ページに表示される項目数、 または <a href="#">仮想モード</a> 時に各クエリーで取得する項目数を 取得または設定します。0はページングが無効であることを示します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <a href="#">Parameters</a>		<a href="#">QueryName</a> で指定された <b>System.ServiceModel.DomainServices.Client.EntityQuery{T}</b> のパラメータのコレクションを取得します。
 <a href="#">QueryName</a>		ソースとして使用される <b>System.ServiceModel.DomainServices.Client.EntityQuery{T}</b> の名前を取得または設定します。
 <a href="#">SortDescriptors</a>		データのソートに使用される <a href="#">C1.Data.DataSource.SortDescriptor</a> オブジェクトのコレクションを取得します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <a href="#">VirtualMode</a>		<a href="#">C1.Data.DataSource.ClientViewSource</a> が仮想モードかどうかを示す値を取得または設定します。 仮想モードは、遅延やパフォーマンスの低下を招くことなく、またページング の手間をかける必要もなく、 GUI コントロールを大規模なデータセットに直接連結できる革新的な技術です。 デフォルトでは、仮想モードは無効です（デフォルト値は <a href="#">C1.Data.DataSource.VirtualModeKind.None</a> ）。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )

[Top](#)

## Public Methods

	Name	Description
 <a href="#">ClearValue</a>		(Inherited from System.Windows.DependencyObject)



⇒💖DeferLoad	ロードに関係する複数のプロパティに対する変更をグループ化して、結果として行われるロード操作を遅延させるために使用されます。 これによってロード操作は最後に、つまり、このメソッドから返されるオブジェクトが破棄されるときに、1回だけ実行されます。(Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
⇒💖GetAnimationBaseValue	(Inherited from System.Windows.DependencyObject)
⇒💖GetValue	(Inherited from System.Windows.DependencyObject)
⇒💖Load	ロード操作を開始します。保留中のすべてのロードは、暗黙的にキャンセルされます。(Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
⇒💖LoadRange	仮想モードの場合に、エンティティの特定範囲をロードします。(Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
⇒💖ReadLocalValue	(Inherited from System.Windows.DependencyObject)
⇒💖Refresh	クライアント側のキャッシュを無視して、ロード操作を開始します。保留中のすべてのロードは、暗黙的にキャンセルされます。(Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
⇒💖SetValue	(Inherited from System.Windows.DependencyObject)

[Top](#)

## Public Events

	Name	Description
⚡	LoadedData	ロード操作が完了したとき、またはロード操作中に例外が生成されたときに発生します。(Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
⚡	PropertyChanged	プロパティ値が変更されたときに発生します。(Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )

[Top](#)

## See Also

### Reference

[RiaViewSource Class](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

## RiaViewSource Constructor

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaViewSource Class](#) : RiaViewSource Constructor

[RiaViewSource](#) クラスの新しいインスタンスを初期化します。

## Syntax

Visual Basic (Declaration)

```
Public Function New()
```

C#

```
public RiaViewSource()
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaViewSource Class](#)

[RiaViewSource Members](#)

## Properties

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaViewSource Class](#)

For a list of all members of this type, see [RiaViewSource members](#).

## Public Properties

Name	Description
 <a href="#">AutoLoad</a>	起動時や、 <a href="#">C1.Data.DataSource.ClientViewSource</a> によって作成されたクエリーが影響を受ける変更が発生したときに、 <a href="#">Load</a>

		<p>が自動的に呼び出されるかどうかを示す値を取得または設定します。</p> <p>デフォルトは True です。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a>)</p>
 <b>BaseView</b>		<p><a href="#">C1.Data.DataSource.ClientViewSource</a></p> <p>がクエリーを作成するための基礎として使用する <a href="#">C1.Data.ClientView&lt;T&gt;</a> のインスタンスを取得または設定します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a>)</p>
 <b>CacheTimeout</b>		<p>仮想モードでロードされたエンティティが、必要かどうかのチェックがされることなくキャッシュに維持される時間を取得または設定します。</p> <p><a href="#">CacheTimeout</a></p> <p>よりも長い時間、エンティティが使用されることも必要と判断されることもなかった場合、 <a href="#">C1.Data.DataSource.ClientViewSource</a> はそのエンティティをキャッシュから削除できます。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a>)</p>
 <b>CurrentClientView</b>		<p>エンティティのロードに使用される現在の<a href="#">クライアントビュー</a>、または<a href="#">仮想モード</a>では null を取得します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a>)</p>
 <b>DataView</b>		<p>最後のロード操作の結果であるエンティティの現在のビューを取得します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a>)</p>
 <b>Dispatcher</b>		<p>(Inherited from System.Windows.DependencyObject)</p>
 <b>FilterDescriptors</b>		<p>ロード実行時に使用される <a href="#">C1.Data.DataSource.FilterDescriptor</a> オブジェクトのコレクションを取得します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a>)</p>
 <b>FilterOperator</b>		<p>フィルタコレクション内の <a href="#">FilterDescriptors</a> どうしを結合するために使用される論理演算子を取得または設定します。</p> <p>デフォルト値は、<a href="#">C1.Data.DataSource.FilterDescriptorLogicalOperator.And</a> です。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a>)</p>
 <b>GroupDescriptors</b>		<p>読み込んだエンティティをいくつかのグループに組織化するために使用される <a href="#">C1.Data.DataSource.GroupDescriptor</a></p>

	オブジェクトのコレクションを取得します。(Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>IsLoading</b> Data	<a href="#">C1.Data.DataSource.ClientViewSource</a> が現在データをロード中かどうかを示す値を取得します。(Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>LoadCommand</b>	この <a href="#">C1.Data.DataSource.ClientViewSource</a> で <b>Load</b> を呼び出す <b>System.Windows.Input.ICommand</b> を取得します。(Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>LoadDelay</b>	自動データロード操作が開始されるまでの遅延時間を取得または設定します。 自動ロードを促す変更が発生した時点から 結果としての <a href="#">Load</a> が開始される時点までが遅延時間です。 デフォルトの遅延時間は、25 ミリ秒です。(Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>LoadSize</b>	<a href="#">Load</a> が実行されるたびにロードされる項目の最大数を取得または設定します。 0 の場合は、要求されたエンティティがすべてロードされます。 デフォルトは0です。(Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>MoveToFirstOnLoad</b>	<a href="#">Load</a> 操作完了後の最初の項目を現在の項目にすることを示す値を取得または設定します (現在の項目が他の方法で設定されていない場合)。(Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>Name</b>	Overridden. <a href="#">C1DataSource.ViewSources</a> コレクション内で参照される、この <a href="#">RiaViewSource</a> の名前を取得します。 これは、 <a href="#">QueryName</a> によって決定されますが、 <a href="#">NameOverride</a> でオーバーライドできます。
 <b>NameOverride</b>	<a href="#">Name</a> プロパティの値をオーバーライドする値を取得または設定します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 <b>PageSize</b>	<a href="#">DataView</a> の各ページに表示される項目数、 または <b>仮想モード</b> 時に各クエリーで取得する項目数を 取得または設定します。0 はページングが無効であることを示します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )

 Parameter s	<a href="#">QueryName</a> で指定された <b>System.ServiceModel.DomainServices.Client.EntityQuery{T}</b> のパラメータのコレクションを取得します。
 QueryName e	ソースとして使用される <b>System.ServiceModel.DomainServices.Client.EntityQuery{T}</b> の名前を取得または設定します。
 SortDescriptors	データのソートに使用される <a href="#">C1.Data.DataSource.SortDescriptor</a> オブジェクトのコレクションを取得します。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )
 VirtualMode	<a href="#">C1.Data.DataSource.ClientViewSource</a> が仮想モードかどうかを示す値を取得または設定します。 仮想モードは、遅延やパフォーマンスの低下を招くことなく、またページング の手間をかける必要もなく、 GUI コントロールを大規模なデータセットに直接連結できる革新的な技術です。 デフォルトでは、仮想モードは無効です（デフォルト値は <a href="#">C1.Data.DataSource.VirtualModeKind.None</a> ）。 (Inherited from <a href="#">C1.Data.DataSource.ClientViewSource</a> )

[Top](#)

## See Also

### Reference

[RiaViewSource Class](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

### Name Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaViewSource Class](#) : Name Property

[C1DataSource.ViewSources](#) コレクション内で参照される、この [RiaViewSource](#) の名前を取得します。これは、[QueryName](#) によって決定されますが、[NameOverride](#) でオーバーライドできます。

## Syntax

Visual Basic (Declaration)

Public Overrides ReadOnly Property Name As System.String	
C#	
public override System.string Name {get;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaViewSource Class](#)

[RiaViewSource Members](#)

### Parameters Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaViewSource Class](#) : Parameters Property

[QueryName](#) で指定された **System.ServiceModel.DomainServices.Client.EntityQuery{T}** のパラメータのコレクションを取得します。

## Syntax

Visual Basic (Declaration)	
Public ReadOnly Property Parameters As System.Windows.Controls.ParameterCollection	
C#	
public System.Windows.Controls.ParameterCollection Parameters {get;}	

## Remarks

[AutoLoad](#) が true に設定されている場合は、このコレクションまたは このコレクション内の **System.Windows.Controls.Parameter** オブジェクトのプロパティを変更すると、[RiaViewSource](#) がデータを再ロードします。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaViewSource Class](#)

[RiaViewSource Members](#)

### QueryName Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaViewSource Class](#) : QueryName Property

ソースとして使用される **System.ServiceModel.DomainServices.Client.EntityQuery{T}** の名前を取得または設定します。

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.CategoryAttribute("Common")&gt; &lt;System.ComponentModel.DefaultValueAttribute()&gt; Public Property QueryName As System.String</pre>	
C#	
<pre>[System.ComponentModel.Category("Common")] [System.ComponentModel.DefaultValue()] public System.string QueryName {get; set;}</pre>	

## Remarks

[AutoLoad](#) が true に設定されている場合は、このプロパティの値を変更すると、[RiaViewSource](#) がデータを再ロードします。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaViewSource Class](#)

[RiaViewSource Members](#)

# RiaViewSourceCollection

[C1.Silverlight.Data.RiaServices Namespace](#) : RiaViewSourceCollection Class

[RiaViewSource](#) オブジェクトの監視可能なコレクション。

## Object Model

RiaViewSourceCollection

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>
Public Class RiaViewSourceCollection
    Inherits System.Collections.ObjectModel.ObservableCollection(Of
RiaViewSource)
```

C#

```
[System.Reflection.DefaultMember("Item")]
public class RiaViewSourceCollection :
System.Collections.ObjectModel.ObservableCollection<RiaViewSource>
```

## Inheritance Hierarchy

System.Object

System.Collections.ObjectModel.Collection<T>

System.Collections.ObjectModel.ObservableCollection<T>

**C1.Silverlight.Data.RiaServices.RiaViewSourceCollection**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaViewSourceCollection Members](#)

[C1.Silverlight.Data.RiaServices Namespace](#)



## Overview

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaViewSourceCollection Class](#)

[RiaViewSource](#) オブジェクトの監視可能なコレクション。

## Object Model

[RiaViewSourceCollection](#)

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Reflection.DefaultMemberAttribute("Item")&gt; Public Class RiaViewSourceCollection     Inherits System.Collections.ObjectModel.ObservableCollection(Of RiaViewSource)</pre>	
C#	
<pre>[System.Reflection.DefaultMember("Item")] public class RiaViewSourceCollection : System.Collections.ObjectModel.ObservableCollection&lt;RiaViewSource&gt;</pre>	

## Inheritance Hierarchy

System.Object  
    System.Collections.ObjectModel.Collection<T>  
        System.Collections.ObjectModel.ObservableCollection<T>  
            **C1.Silverlight.Data.RiaServices.RiaViewSourceCollection**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[RiaViewSourceCollection Members](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)

## Members

[Properties](#) [Methods](#) [Events](#)

C1.Silverlight.Data.RiaServices Namespace : [RiaViewSourceCollection](#) Class



The following tables list the members exposed by [RiaViewSourceCollection](#).

## Public Constructors

	Name	Description
	<a href="#">RiaViewSourceCollection Constructor</a>	


[Top](#)

## Public Properties

	Name	Description
	<a href="#">Count</a>	(Inherited from System.Collections.ObjectModel.Collection<RiaViewSource>)
	<a href="#">Item</a>	指定された <a href="#">name</a> の <a href="#">RiaViewSource</a> を取得します。



[Top](#)








## Protected Properties

	Name	Description
	<a href="#">Items</a>	(Inherited from System.Collections.ObjectModel.Collection<RiaViewSource>)

[Top](#)


## Public Methods




	Name	Description
	<a href="#">Add</a>	(Inherited from System.Collections.ObjectModel.Collection<RiaViewSource>)
	<a href="#">Clear</a>	(Inherited from System.Collections.ObjectModel.Collection<RiaViewSource>)

	<a href="#">Contains</a>	(Inherited from System.Collections.ObjectModel.Collection<RiaViewSource>)
	<a href="#">CopyTo</a>	(Inherited from System.Collections.ObjectModel.Collection<RiaViewSource>)
	<a href="#">GetEnumerator</a>	(Inherited from System.Collections.ObjectModel.Collection<RiaViewSource>)
	<a href="#">IndexOf</a>	(Inherited from System.Collections.ObjectModel.Collection<RiaViewSource>)
	<a href="#">Insert</a>	(Inherited from System.Collections.ObjectModel.Collection<RiaViewSource>)
	<a href="#">Remove</a>	(Inherited from System.Collections.ObjectModel.Collection<RiaViewSource>)
	<a href="#">RemoveAt</a>	(Inherited from System.Collections.ObjectModel.Collection<RiaViewSource>)

[Top](#)


## Protected Methods

	Name	Description
	<a href="#">ClearItems</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<RiaViewSource>)
	<a href="#">InsertItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<RiaViewSource>)
	<a href="#">OnCollectionChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<RiaViewSource>)

	<a href="#">OnPropertyChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<RiaViewSource>)
	<a href="#">RemoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<RiaViewSource>)
	<a href="#">SetItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<RiaViewSource>)


[Top](#)

## Public Events

	Name	Description
	<a href="#">CollectionChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<RiaViewSource>)

[Top](#)

## Protected Events

	Name	Description
	<a href="#">PropertyChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<RiaViewSource>)

[Top](#)

## See Also

### Reference

[RiaViewSourceCollection Class](#)  
[C1.Silverlight.Data.RiaServices Namespace](#)

## RiaViewSourceCollection Constructor

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaViewSourceCollection Class](#) : RiaViewSourceCollection

Constructor

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public RiaViewSourceCollection()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



[RiaViewSourceCollection Class](#)  
[RiaViewSourceCollection Members](#)

## Properties

[C1.Silverlight.Data.RiaServices Namespace](#) : [RiaViewSourceCollection Class](#)

For a list of all members of this type, see [RiaViewSourceCollection members](#).


## Public Properties

	Name	Description
	<a href="#">Count</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;RiaViewSource&gt;</a> )
	<a href="#">Item</a>	指定された <a href="#">name</a> の <a href="#">RiaViewSource</a> を取得します。

[Top](#)

## Protected Properties

	Name	Description
--	------	-------------

 <b>Items</b>	(Inherited from System.Collections.ObjectModel.Collection<RiaViewSource>)
--	--

[Top](#)

## See Also

### Reference

[RiaViewSourceCollection Class](#)

[C1.Silverlight.Data.RiaServices Namespace](#)

### Item Property

[C1.Silverlight.Data.RiaServices Namespace](#) > [RiaViewSourceCollection Class](#) : Item Property

コレクションから取得する [RiaViewSource](#) の名前。

指定された [name](#) の [RiaViewSource](#) を取得します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Shadows ReadOnly Default Property Item( _     ByVal name As System.String _ ) As RiaViewSource</pre>	
C#	
<pre>public new RiaViewSource this[     System.string name ]; {get;}</pre>	

### Parameters

*name*

コレクションから取得する [RiaViewSource](#) の名前。

### Property Value

指定された [name](#) の [RiaViewSource](#)。存在しない場合は null。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [RiaViewSourceCollection Class](#)
- [RiaViewSourceCollection Members](#)

# C1.Util.Licensing Namespace

## Overview

[Inheritance Hierarchy](#)

## Classes

	Class	Description
	<a href="#">LicenseMode</a>	

# See Also

## Reference

[C1.Silverlight.Data.Entity.5 Assembly](#)

## Classes

## LicenseMode

[C1.Util.Licensing Namespace](#) : LicenseMode Class

## Object Model

LicenseMode

## Syntax

Visual Basic (Declaration)	
<code>Public Class LicenseMode</code>	
C#	
<code>public class LicenseMode</code>	

## Inheritance Hierarchy

System.Object  
**C1.Util.Licensing.LicenseMode**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LicenseMode Members](#)  
[C1.Util.Licensing Namespace](#)

### Overview

[C1.Util.Licensing Namespace](#) : LicenseMode Class

## Object Model

LicenseMode

## Syntax

Visual Basic (Declaration)	
<b>Public Class</b> LicenseMode	
C#	
<b>public class</b> LicenseMode	

## Inheritance Hierarchy

System.Object  
**C1.Util.Licensing.LicenseMode**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



Reference

[LicenseMode Members](#)  
[C1.Util.Licensing Namespace](#)

Members

[Fields](#) [Methods](#)

[C1.Util.Licensing Namespace](#) : [LicenseMode](#) Class

The following tables list the members exposed by [LicenseMode](#).

Public Constructors

	Name	Description
	<a href="#">LicenseMode Constructor</a>	



[Top](#)

Public Fields

	Name	Description
 S	<a href="#">EvaluationProperty</a>	

[Top](#)

Public Methods

	Name	Description
 S	<a href="#">GetEvaluation</a>	
 S	<a href="#">SetEvaluation</a>	

[Top](#)

See Also

Reference

[LicenseMode Class](#)  
[C1.Util.Licensing Namespace](#)

# LicenseMode Constructor

[C1.Util.Licensing Namespace](#) > [LicenseMode Class](#) : LicenseMode Constructor

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public LicenseMode()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LicenseMode Class](#)  
[LicenseMode Members](#)

## Methods

>

Name	Description
 <a href="#">GetEvaluation</a>	
 <a href="#">SetEvaluation</a>	

[Top](#)

## See Also

### Reference

[LicenseMode Class](#)  
[C1.Util.Licensing Namespace](#)

## GetEvaluation Method

[C1.Util.Licensing Namespace](#) > [LicenseMode Class](#) : GetEvaluation Method

# Syntax

Visual Basic (Declaration)	
<pre>Public Shared Function GetEvaluation( _     ByVal obj As System.Windows.DependencyObject _ ) As System.Boolean</pre>	
C#	
<pre>public static System.bool GetEvaluation(     System.Windows.DependencyObject obj )</pre>	

## Parameters

*obj*

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [LicenseMode Class](#)
- [LicenseMode Members](#)

## SetEvaluation Method

[C1.Util.Licensing Namespace](#) > [LicenseMode Class](#) : SetEvaluation Method

# Syntax

Visual Basic (Declaration)	
<pre>Public Shared Sub SetEvaluation( _     ByVal obj As System.Windows.DependencyObject, _     ByVal value As System.Boolean _ )</pre>	
C#	
<pre>public static void SetEvaluation( </pre>	

```
System.Windows.DependencyObject obj,  
System.bool value  
)
```

Parameters

*obj*  
  
*value*

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference


- [LicenseMode Class](#)
- [LicenseMode Members](#)

Fields

[C1.Util.Licensing Namespace](#) : [LicenseMode Class](#)

For a list of all members of this type, see [LicenseMode members](#).

Public Fields

	Name	Description
 <b>S</b>	<a href="#">EvaluationProperty</a>	

[Top](#)

See Also

Reference

- [LicenseMode Class](#)
- [C1.Util.Licensing Namespace](#)

EvaluationProperty Field

[C1.Util.Licensing Namespace](#) > [LicenseMode Class](#) : [EvaluationProperty Field](#)

Syntax

Visual Basic (Declaration)	
<code>Public Shared ReadOnly EvaluationProperty As System.Windows.DependencyProperty</code>	
C#	
<code>public static readonly System.Windows.DependencyProperty EvaluationProperty</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LicenseMode Class](#)

[LicenseMode Members](#)


# C1.Silverlight.LiveLinq.5 Assembly

## Namespaces

### (Global) Namespace

## Overview

## Classes

	Class	Description
	<a href="#">C1LiveLinqInfo</a>	C1.LiveLinq.dll アセンブリに関連する情報を提供します。

## See Also

### Reference

[C1.Silverlight.LiveLinq.5 Assembly](#)

# Classes

## C1LiveLinqInfo

(Global) Namespace : C1LiveLinqInfo Class

C1.LiveLinq.dll アセンブリに関連する情報を提供します。

## Object Model

C1LiveLinqInfo

## Syntax

Visual Basic (Declaration)	
Public MustInherit NotInheritable Class C1LiveLinqInfo	
C#	
public static class C1LiveLinqInfo	

## Inheritance Hierarchy

System.Object  
    **C1LiveLinqInfo**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

C1LiveLinqInfo Members  
(Global) Namespace

## Overview

(Global) Namespace : C1LiveLinqInfo Class

C1.LiveLinq.dll アセンブリに関連する情報を提供します。

## Object Model

## Syntax

Visual Basic (Declaration)	
<code>Public MustInherit NotInheritable Class C1LiveLinqInfo</code>	
C#	
<code>public static class C1LiveLinqInfo</code>	

## Inheritance Hierarchy

System.Object

**C1LiveLinqInfo**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1LiveLinqInfo Members](#)  
(Global) Namespace



## Members

[Fields](#)

(Global) Namespace : C1LiveLinqInfo Class

The following tables list the members exposed by [C1LiveLinqInfo](#).

## Public Fields

	Name	Description
	<a href="#">PublicKey_C1_Silverlight_LiveLinq</a>	C1.Silverlight.LiveLinq.dll の公開鍵を含みます。
	<a href="#">PublicKey_System_Core</a>	System.Core.dll の公開鍵を含みます。

[Top](#)

# See Also

## Reference



C1LiveLinqInfo Class  
(Global) Namespace

## Fields

(Global) Namespace : C1LiveLinqInfo Class

For a list of all members of this type, see [C1LiveLinqInfo members](#).

# Public Fields

	Name	Description
	<a href="#">PublicKey_C1_Silverlight_LiveLinq</a>	C1.Silverlight.LiveLinq.dll の公開鍵を含みます。
	<a href="#">PublicKey_System_Core</a>	System.Core.dll の公開鍵を含みます。

[Top](#)

# See Also

## Reference

C1LiveLinqInfo Class  
(Global) Namespace

## PublicKey\_C1\_Silverlight\_LiveLinq Field

[Example](#)

(Global) Namespace > [C1LiveLinqInfo Class](#) : PublicKey\_C1\_Silverlight\_LiveLinq Field

C1.Silverlight.LiveLinq.dll の公開鍵を含みます。

# Syntax

Visual Basic (Declaration)	
<code>Public Const PublicKey_C1_Silverlight_LiveLinq As System.String</code>	
C#	
<code>public const System.string PublicKey_C1_Silverlight_LiveLinq</code>	

# Example



- [C#](#)

```
[assembly:
System.Runtime.CompilerServices.InternalsVisibleTo("C1.Silverlight.LiveLinq,
PublicKey=" + C1LiveLinqInfo.PublicKey_C1_Silverlight_LiveLinq)]
```

## Remarks

**System.MemberAccessException** 例外を避けるには、この定数を使用して 内部メンバを C1.Silverlight.LiveLinq.dll に公開します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1LiveLinqInfo Class](#)

[C1LiveLinqInfo Members](#)

PublicKey\_System\_Core Field

[Example](#)

(Global) Namespace > [C1LiveLinqInfo Class](#) : PublicKey\_System\_Core Field

System.Core.dll の公開鍵を含みます。

## Syntax

Visual Basic (Declaration)

```
Public Const PublicKey_System_Core As System.String
```

C#

```
public const System.string PublicKey_System_Core
```

## Example

- [C#](#)

```
[assembly:
System.Runtime.CompilerServices.InternalsVisibleToAttribute("System.Core,
PublicKey=" + C1LiveLinqInfo.PublicKey_System_Core)]
```

## Remarks

**System.MemberAccessException** 例外を避けるには、この定数を使用して 内部メンバを System.Core.dll に公開します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



### Reference

[C1LiveLinqInfo Class](#)  
[C1LiveLinqInfo Members](#)



# C1.LiveLinq Namespace

## Overview

### Classes




	Class	Description
	<a href="#">LiveViewExtensions</a>	Provides a set of static (extension) methods used in queries to define live views.
	<a href="#">SourceChangeEventArgs&lt;T&gt;</a>	Provides data for the <a href="#">IObservableSource&lt;T&gt;.Changed</a> event.

### Interfaces

	Interface	Description
	<a href="#">IObservableSource&lt;T&gt;</a>	LiveLinq 機能（ライブビュー）に必要なメソッドとイベントを提供します。 。
	<a href="#">ITransaction</a>	Represents a transaction with an explicit scope.

### Enumerations

	Enumeration	Description
--	-------------	-------------

	<a href="#">Order</a>	Indicates if a certain order is required in the result collection of an operation.
	<a href="#">SourceChangeType</a>	Describes a change occurring in a collection.
	<a href="#">TransactionState</a>	Enumeration of the possible states an <a href="#">ITransaction</a> can be in.

## See Also

### Reference

[C1.Silverlight.LiveLinq.5 Assembly](#)

## Classes

### LiveViewExtensions

[C1.LiveLinq Namespace](#) : LiveViewExtensions Class

Provides a set of static (extension) methods used in queries to define live views.

## Object Model

LiveViewExtensions

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public MustInherit NotInheritable Class LiveViewExtensions</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static class LiveViewExtensions</pre>	

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViewExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Members](#)  
[C1.LiveLinq Namespace](#)

## Overview

[C1.LiveLinq Namespace](#) : LiveViewExtensions Class

Provides a set of static (extension) methods used in queries to define live views.

## Object Model

LiveViewExtensions

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public MustInherit NotInheritable Class LiveViewExtensions</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static class LiveViewExtensions</pre>	

## Inheritance Hierarchy

System.Object  
    **C1.LiveLinq.LiveViewExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference










## Members

### Methods

C1.LiveLinq Namespace : LiveViewExtensions Class

The following tables list the members exposed by [LiveViewExtensions](#).

## Public Methods

	Name	Description
≡ 	<a href="#">AsLive</a>	Overloaded. Creates a view based on the specified <a href="#">IObservableSource&lt;T&gt;</a> collection.
≡ 	<a href="#">AsNonUpdatable&lt;T&gt;</a>	Specifies a view as read-only.
≡ 	<a href="#">AsUpdatable&lt;T&gt;</a>	Specifies a view as updatable.
≡ 	<a href="#">LiveAggregate</a>	Overloaded. Applies an accumulator function over a view.
≡ 	<a href="#">LiveAverage</a>	Overloaded. Computes the average of a view of <b>System.Int32</b> values.
≡ 	<a href="#">LiveCount</a>	Overloaded. A view representing the number of elements in a view.
≡ 	<a href="#">LiveMax</a>	Overloaded. Computes the maximum value of a view of <b>System.Double</b> values.
≡ 	<a href="#">LiveMin</a>	Overloaded. Computes the minimum value of a view of nullable <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.
≡ 	<a href="#">LiveSum</a>	Overloaded. Computes the sum of a view of <b>System.Int32</b> values.

[Top](#)

## See Also

## Reference

[LiveViewExtensions Class](#)










[C1.LiveLinq Namespace](#)

## Methods

[C1.LiveLinq Namespace](#) : [LiveViewExtensions Class](#)

For a list of all members of this type, see [LiveViewExtensions members](#).

## Public Methods

	Name	Description
≡  <a href="#">S</a>	<a href="#">AsLive</a>	Overloaded. Creates a view based on the specified <a href="#">IObservableSource&lt;T&gt;</a> collection.
≡  <a href="#">S</a>	<a href="#">AsNonUpdatable&lt;T&gt;</a>	Specifies a view as read-only.
≡  <a href="#">S</a>	<a href="#">AsUpdatable&lt;T&gt;</a>	Specifies a view as updatable.
≡  <a href="#">S</a>	<a href="#">LiveAggregate</a>	Overloaded. Applies an accumulator function over a view.
≡  <a href="#">S</a>	<a href="#">LiveAverage</a>	Overloaded. Computes the average of a view of <b>System.Int32</b> values.
≡  <a href="#">S</a>	<a href="#">LiveCount</a>	Overloaded. A view representing the number of elements in a view.
≡  <a href="#">S</a>	<a href="#">LiveMax</a>	Overloaded. Computes the maximum value of a view of <b>System.Double</b> values.
≡  <a href="#">S</a>	<a href="#">LiveMin</a>	Overloaded. Computes the minimum value of a view of nullable <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.
≡  <a href="#">S</a>	<a href="#">LiveSum</a>	Overloaded. Computes the sum of a view of <b>System.Int32</b> values.

[Top](#)

# See Also

## Reference

[LiveViewExtensions Class](#)  
[C1.LiveLinq Namespace](#)

## AsLive Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : AsLive Method

Creates a view based on the specified [IObservableSource<T>](#) collection.

# Overload List

Overload	Description
<a href="#">AsLive&lt;T&gt;(IObservableSource&lt;T&gt;)</a>	Creates a view based on the specified <a href="#">IObservableSource&lt;T&gt;</a> collection.
<a href="#">AsLive&lt;T&gt;(IObservableSource&lt;T&gt;,ViewOrder)</a>	Creates a view based on the specified <a href="#">IObservableSource&lt;T&gt;</a> collection.
<a href="#">AsLive&lt;T&gt;(INotifyCollectionChanged)</a>	指定された <b>System.Collections.Specialized.INotifyCollectionChanged</b> データソースに基づいてビューを作成します。
<a href="#">AsLive&lt;T&gt;(INotifyCollectionChanged,ViewOrder)</a>	指定された <b>System.Collections.Specialized.INotifyCollectionChanged</b> データソースに基づいてビューを作成します。
<a href="#">AsLive&lt;T&gt;(ObservableCollection&lt;T&gt;)</a>	<a href="#">AsLive&lt;T&gt;(INotifyCollectionChanged)</a> メソッドの型付き特殊化。
<a href="#">AsLive&lt;T&gt;(ObservableCollection&lt;T&gt;,ViewOrder)</a>	<a href="#">AsLive&lt;T&gt;(INotifyCollectionChanged,ViewOrder)</a> メソッドの型付き特殊化。
<a href="#">AsLive&lt;T&gt;(ReadOnlyObservableCollection&lt;T&gt;)</a>	<a href="#">AsLive&lt;T&gt;(INotifyCollectionChanged)</a>

>)	メソッドの型付き特殊化。
<a href="#">AsLive&lt;T&gt;(ReadOnlyObservableCollection&lt;T&gt;,ViewOrder)</a>	<a href="#">AsLive&lt;T&gt;(INotifyCollectionChanged,ViewOrder)</a> メソッドの型付き特殊化。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)

[AsLive<T>\(IObservableSource<T>\) Method](#)  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) : [AsLive<T>\(IObservableSource<T>\) Method](#)

The type of the elements in the collection.

The [IObservableSource<T>](#) collection to expose as a view.

Creates a view based on the specified [IObservableSource<T>](#) collection.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function AsLive(Of T)( _     ByVal source As IObservableSource(Of T) _ ) As View(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsLive&lt;T&gt;(     IObservableSource&lt;T&gt; source )</pre>	

### Parameters

*source*



The [IObservableSource<T>](#) collection to expose as a view.

## Type Parameters

*T*

The type of the elements in the collection.

## Return Value

A view that contains the same elements as the [IObservableSource<T>](#)

## Remarks

The resulting view may have its elements ordered differently than they are ordered in the *source* collection. Correspondingly, views built on this resulting view (for example, if you filter it with **Where**) will not preserve the source order either. If you need to preserve the source order, consider using the other **AsLive** overload where you can specify to what extent you need the order to be preserved.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

[AsLive<T>\(IObservableSource<T>,ViewOrder\) Method](#)

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) :

[AsLive<T>\(IObservableSource<T>,ViewOrder\) Method](#)

The type of the elements in the collection.

The [IObservableSource<T>](#) collection to expose as a view.

Specifies whether to preserve source item order.

Creates a view based on the specified [IObservableSource<T>](#) collection.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function AsLive(Of T)( _     ByVal source As IObservableSource(Of T), _     ByVal order As ViewOrder _ ) As View(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsLive&lt;T&gt;(     IObservableSource&lt;T&gt; source,     ViewOrder order )</pre>	

## Parameters

*source*

The [IObservableSource<T>](#) collection to expose as a view.

*order*

Specifies whether to preserve source item order.

## Type Parameters

*T*

The type of the elements in the collection.

## Return Value

A view that contains the same elements as the [IObservableSource<T>](#)

## Remarks

If the *order* parameter specifies preserving item order, the order of items in the source is preserved, at a certain performance cost, in the resulting view and in views based on it (for example, if you filter it with **Where**).

Note that **Join** does not preserve source order. If you need to order a join result, use **OrderBy** after **Join**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

AsLive<T>(INotifyCollectionChanged) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) :

AsLive<T>(INotifyCollectionChanged) Method

ビュー内の要素の型。

ビューとして公開する **System.Collections.Specialized.INotifyCollectionChanged** データソース。

指定された **System.Collections.Specialized.INotifyCollectionChanged** データソースに基づいてビューを作成します。

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function AsLive(Of T)( _  
    ByVal source As System.Collections.Specialized.INotifyCollectionChanged _  
) As View(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static View<T> AsLive<T>(  
    System.Collections.Specialized.INotifyCollectionChanged source  
)
```

### Parameters

*source*

ビューとして公開する **System.Collections.Specialized.INotifyCollectionChanged** データソース。

### Type Parameters

T

ビュー内の要素の型。

## Return Value

**System.Collections.Specialized.INotifyCollectionChanged**

データソースと同じ要素を含むビュー。

## Remarks

このメソッドを使用して、**System.Collections.Specialized.INotifyCollectionChanged** を実装する既存のデータソースからビューを構築します。

このデータソースの要素タイプは、**System.ComponentModel.INotifyPropertyChanged** を実装する必要があります。「組み込みのコレクションクラス `IndexedCollection(T)` の使用 (オブジェクトへの LiveLinq) |tag=Using\_the\_built\_in\_collection\_class\_IndexedCollectionT\_LiveLinq\_to\_Objects」を参照してください。

結果のビューは、要素の順序がソースの順序と異なる場合があります。

また、この結果のビューに基づいて構築されるビュー (**Where** でフィルタした場合など) でも、ソースの順序は維持されません。

ソースの順序を維持する必要がある場合は、もう 1 つの **AsLive** オーバーロードの使用を検討してください。

この場合は、どこまで順序を維持する必要があるかを指定できます。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

**AsLive<T>(INotifyCollectionChanged,ViewOrder) Method**

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) :

**AsLive<T>(INotifyCollectionChanged,ViewOrder) Method**

ビュー内の要素の型。

ビューとして公開する **System.Collections.Specialized.INotifyCollectionChanged** データソース。

ソースの項目順序を維持するかどうかを指定します。

指定された **System.Collections.Specialized.INotifyCollectionChanged** データソースに基づいてビューを作成します。

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function AsLive(Of T)( _  
    ByVal source As System.Collections.Specialized.INotifyCollectionChanged, _  
    ByVal order As ViewOrder _  
) As View(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static View<T> AsLive<T>(  
    System.Collections.Specialized.INotifyCollectionChanged source,  
    ViewOrder order  
)
```

### Parameters

*source*

ビューとして公開する **System.Collections.Specialized.INotifyCollectionChanged** データソース。

*order*

ソースの項目順序を維持するかどうかを指定します。

### Type Parameters

*T*

ビュー内の要素の型。

### Return Value

**System.Collections.Specialized.INotifyCollectionChanged** データソースと同じ要素を含むビュー。

## Remarks

このメソッドを使用して、**System.Collections.Specialized.INotifyCollectionChanged** を実装する既存のデータソースからビューを構築します。

このデータソースの要素タイプは、**System.ComponentModel.INotifyProperty**  
**tyChanged** を実装する必要があります。「組み込みのコレクションクラス  
IndexedCollection(T) の使用 (オブジェクトへの  
LiveLinq) |tag=Using\_the\_built\_in\_collection\_class\_IndexedCollectionT\_LiveLinq  
\_to\_Objects」を参照してください。

#### *order*

パラメータで項目順序の維持が指定されている場合、ある程度のパフォーマンスの低下はありますが、結果のビューおよびそれに基づくビュー (**Where** でフィルタした場合など) でソースの項目順序が維持されます。

#### **Join**

ではソースの順序が維持されません。結合結果を並べ替える必要がある場合は、**Join** の後に **OrderBy** を使用します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

AsLive<T>(ObservableCollection<T>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) :

AsLive<T>(ObservableCollection<T>) Method

ビュー内の要素の型。

ビューとして公開する **System.Collections.ObjectModel.ObservableCollection`1**。

[AsLive<T>\(INotifyCollectionChanged\)](#) メソッドの型付き特殊化。

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function AsLive(Of T)( _
    ByVal source As System.Collections.ObjectModel.ObservableCollection(Of T)
    _
```

```
) As View(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static View<T> AsLive<T>(  
    System.Collections.ObjectModel.ObservableCollection<T> source  
)
```

## Parameters

*source*

ビューとして公開する **System.Collections.ObjectModel.ObservableCollection`1**。

## Type Parameters

*T*

ビュー内の要素の型。

## Return Value

**System.Collections.ObjectModel.ObservableCollection`1**  
と同じ要素を含むビュー。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

AsLive<T>(ObservableCollection<T>,ViewOrder) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) :

AsLive<T>(ObservableCollection<T>,ViewOrder) Method

ビュー内の要素の型。

ビューとして公開する **System.Collections.ObjectModel.ObservableCollection`1**。

ソースの項目順序を維持するかどうかを指定します。

AsLive<T>(INotifyCollectionChanged,ViewOrder) メソッドの型付き特殊化。

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function AsLive(Of T)( _     ByVal source As System.Collections.ObjectModel.ObservableCollection(Of T),     _     ByVal order As ViewOrder _ ) As View(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsLive&lt;T&gt;(     System.Collections.ObjectModel.ObservableCollection&lt;T&gt; source,     ViewOrder order )</pre>	

### Parameters

*source*

ビューとして公開する **System.Collections.ObjectModel.ObservableCollection`1**。

*order*

ソースの項目順序を維持するかどうかを指定します。

### Type Parameters

*T*

ビュー内の要素の型。

### Return Value

**System.Collections.ObjectModel.ObservableCollection`1**  
と同じ要素を含むビュー。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2



## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

[AsLive<T>\(ReadOnlyObservableCollection<T>\) Method](#)

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) :

[AsLive<T>\(ReadOnlyObservableCollection<T>\) Method](#)

ビュー内の要素の型。

ビューとして公開する **System.Collections.ObjectModel.ReadOnlyObservableCollection`1**。

[AsLive<T>\(INotifyCollectionChanged\)](#) メソッドの型付き特殊化。

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function AsLive(Of T)( _  
    ByVal source As  
System.Collections.ObjectModel.ReadOnlyObservableCollection(Of T) _  
) As View(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static View<T> AsLive<T>(  
    System.Collections.ObjectModel.ReadOnlyObservableCollection<T> source  
)
```

### Parameters

*source*

ビューとして公開する

**System.Collections.ObjectModel.ReadOnlyObservableCollection`1**。

### Type Parameters

*T*

ビュー内の要素の型。

### Return Value

**System.Collections.ObjectModel.ReadOnlyObservableCollection<T>**  
と同じ要素を含むビュー。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

[AsLive<T>\(ReadOnlyObservableCollection<T>,ViewOrder\) Method](#)

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [AsLive Method](#) :

[AsLive<T>\(ReadOnlyObservableCollection<T>,ViewOrder\) Method](#)

ビュー内の要素の型。

ビューとして公開する **System.Collections.ObjectModel.ReadOnlyObservableCollection<T>**。

ソースの項目順序を維持するかどうかを指定します。

[AsLive<T>\(INotifyCollectionChanged,ViewOrder\)](#) メソッドの型付き特殊化。

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function AsLive(Of T)( _  
    ByVal source As  
System.Collections.ObjectModel.ReadOnlyObservableCollection(Of T), _  
    ByVal order As ViewOrder _  
) As View(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static View<T> AsLive<T>(  
    System.Collections.ObjectModel.ReadOnlyObservableCollection<T> source,  
    ViewOrder order  
)
```

## Parameters

*source*

ビューとして公開する

**System.Collections.ObjectModel.ReadOnlyObservableCollection`1**。

*order*

ソースの項目順序を維持するかどうかを指定します。

## Type Parameters

*T*

ビュー内の要素の型。

## Return Value

**System.Collections.ObjectModel.ReadOnlyObservableCollection`1**

と同じ要素を含むビュー。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

## AsNonUpdatable<T> Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : AsNonUpdatable<T> Method

The type of the elements in the view.

The view to specify as read-only.

Specifies a view as read-only.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Shared Function AsNonUpdatable(Of T)( _
    ByVal view As View(Of T) _
) As View(Of T)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<T> AsNonUpdatable<T>(
    View<T> view
)
```

## Parameters

*view*

The view to specify as read-only.

## Type Parameters

*T*

The type of the elements in the view.

## Return Value

The same *view*.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[AsUpdatable<T> Method](#)

## AsUpdatable<T> Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : AsUpdatable<T> Method

The type of the elements in the view.

The view to specify as updatable.

Specifies a view as updatable.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Function AsUpdatable(Of T)( _     ByVal view As View(Of T) _ ) As View(Of T)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;T&gt; AsUpdatable&lt;T&gt;(     View&lt;T&gt; view )</pre>	

### Parameters

*view*

The view to specify as updatable.

### Type Parameters

*T*

The type of the elements in the view.

### Return Value

The same *view*.

## Remarks

This method is used with **Join** operation to specify which of the two parts of the join you want to be updatable.

Properties exposed by a view can be updatable or read-only. Updatable properties of a view can be modified directly in the view. Read-only properties of a view cannot be modified directly in the view, but they still reflect up-to-date values of the source, so the difference is often not critical, you can always modify corresponding property in the source, that will automatically change the property in the view.

Only one of the two arguments of a **Join** can be updatable, the one you qualified with **AsUpdatable()**. In absence of this qualification, both parts of the join are read-only. For example, in `from c in customers.AsLive() join o in orders.AsLive().AsUpdatable() on o.CustomerID equals c.CustomerID select new { c.CustomerName, o.OrderDate,`

`o.OrderAmount }` **CustomerName** is read-only, and **OrderDate** and **OrderAmount** are updatable.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

- [LiveViewExtensions Class](#)
- [LiveViewExtensions Members](#)
- [IsReadOnly Property](#)
- [ViewRow Class](#)
- [ViewRowState Enumeration](#)

### LiveAggregate Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : LiveAggregate Method

Applies an accumulator function over a view.

## Overload List

Overload	Description
<a href="#">LiveAggregate&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,TSource,TSource&gt;&gt;,Expression&lt;Func&lt;TSource,TSource,TSource&gt;&gt;,Expression&lt;Func&lt;TSource,TSource,Boolean&gt;&gt;&gt;)</a>	Applies an accumulator function over

	ra vie w.
<code>LiveAggregate&lt;TSource,TAccumulate&gt;(View&lt;TSource&gt;,TAccumulate,Expression&lt;Func&lt;TA ccumulate,TSource,TAccumulate&gt;&gt;,Expression&lt;Func&lt;TAccumulate,TSource,TAccumulate&gt; &gt;,Expression&lt;Func&lt;TAccumulate,TSource,Boolean&gt;&gt;)</code>	Ap plie s an acc um ulat or fun ctio n ove ra vie w. The spe cifi ed see d val ue is use d as the initi al acc um ulat

	or val ue.
<code>LiveAggregate&lt;TSource,TAccumulate,TResult&gt;(View&lt;TSource&gt;,TAccumulate,Expression&lt;Func&lt;TAccumulate,TSource,TAccumulate&gt;&gt;,Expression&lt;Func&lt;TAccumulate,TSource,TAccumulate&gt;&gt;,Expression&lt;Func&lt;TAccumulate,TSource,Boolean&gt;&gt;,Expression&lt;Func&lt;TAccumulate,TResult&gt;&gt;)</code>	Ap plie s an acc um ulat or fun ctio n ove r a vie w. The spe cifi ed see d val ue is use d as the initi al acc um



	ulat or val ue, and the spe cifi ed fun ctio n is use d to sele ct the res ult val ue.
--	---

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)

LiveAggregate<TSource>(View<TSource>,Expression<Func<TSource,TSource,TSource>>,Expres  
sion<Func<TSource,TSource,TSource>>,Expression<Func<TSource,TSource,Boolean>>)

Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAggregate Method](#) :

LiveAggregate<TSource>(View<TSource>,Expression<Func<TSource,TSource,TSource>>,Expression<Func<TSource,TSource,TSource>>,Expression<Func<TSource,TSource,Boolean>>) Method

The type of the elements of *source*.

A view to aggregate over.

An accumulator function to be invoked on each element that is added to the source view.

A function to be applied to the accumulated value and to an element to obtain the changed accumulated value, when an element is removed from the source view.

A function used to determine whether *funcRemove* must be applied when an element is removed from the source view, or the accumulated value is not affected by its removal.

Applies an accumulator function over a view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAggregate(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal funcAdd As System.Linq.Expressions.Expression(Of Func(Of TSource,TSource,TSource))), _     ByVal funcRemove As System.Linq.Expressions.Expression(Of Func(Of TSource,TSource,TSource))), _     ByVal funcRemoveDefined As System.Linq.Expressions.Expression(Of Func(Of TSource,TSource,Boolean))) _ ) As AggregationView(Of TSource,TSource)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,TSource&gt; LiveAggregate&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,TSource,TSource&gt;&gt; funcAdd,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,TSource,TSource&gt;&gt; funcRemove,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,TSource,bool&gt;&gt; funcRemoveDefined )</pre>	

### Parameters

*source*

A view to aggregate over.

*funcAdd*

An accumulator function to be invoked on each element that is added to the source view.

*funcRemove*

A function to be applied to the accumulated value and to an element to obtain the changed accumulated value, when an element is removed from the source view.

*funcRemoveDefined*

A function used to determine whether *funcRemove* must be applied when an element is removed from the source view, or the accumulated value is not affected by its removal.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the final accumulator value.

## Remarks

It is possible to use standard LINQ query operator **Aggregate** instead of **LiveAggregate**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Aggregate** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAggregate** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

LiveAggregate<TSource,TAccumulate>(View<TSource>,TAccumulate,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,Boolean>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAggregate Method](#) :

LiveAggregate<TSource,TAccumulate>(View<TSource>,TAccumulate,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,Boolean>>) Method

The type of the elements of *source*.

The type of the accumulator value.

A view to aggregate over.

The initial accumulator value.

An accumulator function to be invoked on each element that is added to the source view.

A function to be applied to the accumulated value and to an element to obtain the changed accumulated value, when an element is removed from the source view.

A function used to determine whether *funcRemove* must be applied when an element is removed from the source view, or the accumulated value is not affected by its removal.

Applies an accumulator function over a view. The specified seed value is used as the initial accumulator value.

## Syntax

Visual Basic (Declaration)

```

<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAggregate
    (Of TSource,TAccumulate)( _
    ByVal source As View(Of TSource), _
    ByVal seed As TAccumulate, _
    ByVal funcAdd As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate,TSource,TAccumulate)), _
    ByVal funcRemove As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate,TSource,TAccumulate)), _
    ByVal funcRemoveDefined As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate,TSource,Boolean)) _
) As AggregationView(Of TSource,TAccumulate)

```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,TAccumulate>
LiveAggregate<TSource,TAccumulate>(
    View<TSource> source,
    TAccumulate seed,
    System.Linq.Expressions.Expression<Func<TAccumulate,TSource,TAccumulate>>
funcAdd,
    System.Linq.Expressions.Expression<Func<TAccumulate,TSource,TAccumulate>>
funcRemove,
    System.Linq.Expressions.Expression<Func<TAccumulate,TSource,bool>>
funcRemoveDefined
)
```

## Parameters

*source*

A view to aggregate over.

*seed*

The initial accumulator value.

*funcAdd*

An accumulator function to be invoked on each element that is added to the source view.

*funcRemove*

A function to be applied to the accumulated value and to an element to obtain the changed accumulated value, when an element is removed from the source view.

*funcRemoveDefined*

A function used to determine whether *funcRemove* must be applied when an element is removed from the source view, or the accumulated value is not affected by its removal.

## Type Parameters

*TSource*

The type of the elements of *source*.

*TAccumulate*

The type of the accumulator value.

## Return Value

A view representing the final accumulator value.

## Remarks

It is possible to use standard LINQ query operator **Aggregate** instead of **LiveAggregate**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Aggregate** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAggregate** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

`LiveAggregate<TSource,TAccumulate,TResult>(View<TSource>,TAccumulate,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,Boolean>>,Expression<Func<TAccumulate,TResult>>) Method`

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAggregate Method](#) :

`LiveAggregate<TSource,TAccumulate,TResult>(View<TSource>,TAccumulate,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,TAccumulate>>,Expression<Func<TAccumulate,TSource,Boolean>>,Expression<Func<TAccumulate,TResult>>) Method`

The type of the elements of *source*.

The type of the accumulator value.

The type of the resulting value.

A view to aggregate over.

The initial accumulator value.

An accumulator function to be invoked on each element that is added to the source view.

A function to be applied to the accumulated value and to an element to obtain the changed accumulated value, when an element is removed from the source view.

A function used to determine whether *funcRemove* must be applied when an element is removed from the source view, or the accumulated value is not affected by its removal.

A function to transform the final accumulator value into the result value.

Applies an accumulator function over a view. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAggregate
    (Of TSource, TAccumulate, TResult)( _
    ByVal source As View(Of TSource), _
    ByVal seed As TAccumulate, _
    ByVal funcAdd As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate, TSource, TAccumulate)), _
    ByVal funcRemove As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate, TSource, TAccumulate)), _
    ByVal funcRemoveDefined As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate, TSource, Boolean)), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
TAccumulate, TResult)) _
) As AggregationView(Of TSource, TResult)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource, TResult>
LiveAggregate<TSource, TAccumulate, TResult>(
    View<TSource> source,
    TAccumulate seed,
    System.Linq.Expressions.Expression<Func<TAccumulate, TSource, TAccumulate>>
funcAdd,
    System.Linq.Expressions.Expression<Func<TAccumulate, TSource, TAccumulate>>
funcRemove,
    System.Linq.Expressions.Expression<Func<TAccumulate, TSource, bool>>
funcRemoveDefined,
```

```
System.Linq.Expressions.Expression<Func<TAccumulate, TResult>>  
resultSelector  
)
```

## Parameters

*source*

A view to aggregate over.

*seed*

The initial accumulator value.

*funcAdd*

An accumulator function to be invoked on each element that is added to the source view.

*funcRemove*

A function to be applied to the accumulated value and to an element to obtain the changed accumulated value, when an element is removed from the source view.

*funcRemoveDefined*

A function used to determine whether *funcRemove* must be applied when an element is removed from the source view, or the accumulated value is not affected by its removal.

*resultSelector*

A function to transform the final accumulator value into the result value.

## Type Parameters

*TSource*

The type of the elements of *source*.

*TAccumulate*

The type of the accumulator value.

*TResult*

The type of the resulting value.

## Return Value

A view representing the final accumulator value.

## Remarks



It is possible to use standard LINQ query operator **Aggregate** instead of **LiveAggregate**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Aggregate** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAggregate** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

### LiveAverage Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : LiveAverage Method

Computes the average of a view of **System.Int32** values.

## Overload List

Overload	Description
<a href="#">LiveAverage(View&lt;Int32&gt;)</a>	Computes the average of a view of <b>System.Int32</b> values.

<code>LiveAverage(View&lt;Nullable&lt;Int32&gt;&gt;)</code>	Computes the average of a view of nullable <b>System.Int32</b> values.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int32&gt;&gt;)</code>	Computes the average of a view of <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int32&gt;&gt;&gt;)</code>	Computes the average of a view of nullable <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveAverage(View&lt;Int64&gt;)</code>	Computes the average of a view of

	<b>System.Int64</b> values.
<a href="#">LiveAverage(View&lt;Nullable&lt;Int64&gt;&gt;)</a>	Computes the average of a view of nullable <b>System.Int64</b> values.
<a href="#">LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int64&gt;&gt;)</a>	Computes the average of a view of <b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source view.
<a href="#">LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int64&gt;&gt;&gt;)</a>	Computes the average of a view of nullable <b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source view.

<code>LiveAverage(View&lt;Decimal&gt;)</code>	Computes the average of a view of <b>System.Decimal</b> values.
<code>LiveAverage(View&lt;Nullable&lt;Decimal&gt;&gt;)</code>	Computes the average of a view of nullable <b>System.Decimal</b> values.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Decimal&gt;&gt;)</code>	Computes the average of a view of <b>System.Decimal</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Decimal&gt;&gt;&gt;)</code>	Computes the average of a view of nullable <b>System.Decimal</b> values that are obtained by invoking a transform

	function on each element of the source view.
<code>LiveAverage(View&lt;Double&gt;)</code>	Computes the average of a view of <b>System.Double</b> values.
<code>LiveAverage(View&lt;Nullable&lt;Double&gt;&gt;)</code>	Computes the average of a view of nullable <b>System.Double</b> values.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Double&gt;&gt;)</code>	Computes the average of a view of <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Double&gt;&gt;&gt;)</code>	Computes the average of a view of nullable <b>System.Double</b> values that

	are obtained by invoking a transform function on each element of the source view.
<code>LiveAverage(View&lt;Single&gt;)</code>	Computes the average of a view of <b>System.Single</b> values.
<code>LiveAverage(View&lt;Nullable&lt;Single&gt;&gt;)</code>	Computes the average of a view of nullable <b>System.Single</b> values.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Single&gt;&gt;)</code>	Computes the average of a view of <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveAverage&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Single&gt;&gt;&gt;)</code>	Computes the average of a view of

	nullable <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.
--	---

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)

LiveAverage(View<Int32>) Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) : LiveAverage(View<Int32>) Method

A view containing the values to calculate the average of.

Computes the average of a view of **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Integer) _ ) As AggregationView(Of Integer,Double)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()]</pre>	

```
public static AggregationView<int,double> LiveAverage(  
    View<int> source  
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage(View<Nullable<Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage(View<Nullable<Int32>>) Method

A view containing the values to calculate the average of.

Computes the average of a view of nullable **System.Int32** values.

## Syntax



Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Nullable(Of Integer)) _ ) As AggregationView(Of Nullable(Of Integer),Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;int&gt;,Nullable&lt;double&gt;&gt; LiveAverage(     View&lt;Nullable&lt;int&gt;&gt; source )</pre>	

## Parameters

*source*

A view containing the values to calculate the average of.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAverage(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Integer)) _
) As AggregationView(Of TSource,Double)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,double> LiveAverage<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,int>> selector
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of nullable **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>>  
Public Overloads Shared Function LiveAverage(Of TSource)( _
```

```

ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Nullable(Of Integer))) _
) As AggregationView(Of TSource,Nullable(Of Double))

```

C#

```

[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,Nullable<double>> LiveAverage<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,Nullable<int>>> selector
)

```

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage(View<Int64>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) : LiveAverage(View<Int64>) Method

A view containing the values to calculate the average of.

Computes the average of a view of **System.Int64** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Long) _ ) As AggregationView(Of Long,Double)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;long,double&gt; LiveAverage(     View&lt;long&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the average of.

### Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage(View<Nullable<Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage(View<Nullable<Int64>>) Method

A view containing the values to calculate the average of.

Computes the average of a view of nullable **System.Int64** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Nullable(Of Long)) _ ) As AggregationView(Of Nullable(Of Long),Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;long&gt;,Nullable&lt;double&gt;&gt; LiveAverage(     View&lt;Nullable&lt;long&gt;&gt; source )</pre>	

## Parameters

*source*

A view containing the values to calculate the average of.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre> &lt;System.Runtime.CompilerServices.ExtensionAttribute()&gt; Public Overloads Shared Function LiveAverage(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Long)) _ ) As AggregationView(Of TSource,Double) </pre>	
C#	
<pre> [System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,double&gt; LiveAverage&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,long&gt;&gt; selector ) </pre>	

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.



## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of nullable **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveAverage(Of TSource)( _  
    ByVal source As View(Of TSource), _  
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of  
TSource,Nullable(Of Long))) _  
) As AggregationView(Of TSource,Nullable(Of Double))
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,Nullable<double>> LiveAverage<TSource>(   
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,Nullable<long>>> selector  
)
```

### Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage(View<Decimal>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) : LiveAverage(View<Decimal>) Method

A view containing the values to calculate the average of.

Computes the average of a view of **System.Decimal** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Decimal) _ ) As AggregationView(Of Decimal,Decimal)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;decimal,decimal&gt; LiveAverage(     View&lt;decimal&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the average of.

### Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage(View<Nullable<Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage(View<Nullable<Decimal>>) Method

A view containing the values to calculate the average of.

Computes the average of a view of nullable **System.Decimal** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveAverage( _  
    ByVal source As View(Of Nullable(Of Decimal)) _  
) As AggregationView(Of Nullable(Of Decimal),Nullable(Of Decimal))
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<Nullable<decimal>,Nullable<decimal>>  
LiveAverage(  
    View<Nullable<decimal>> source  
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant

algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

[LiveAverage<TSource>\(View<TSource>,Expression<Func<TSource,Decimal>>\) Method](#)

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

[LiveAverage<TSource>\(View<TSource>,Expression<Func<TSource,Decimal>>\) Method](#)

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAverage(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Decimal))) _
) As AggregationView(Of TSource,Decimal)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,decimal> LiveAverage<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,decimal>> selector
```

```
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>)

Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of nullable **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Decimal)))) _ ) As AggregationView(Of TSource,Nullable(Of Decimal))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;decimal&gt;&gt; LiveAverage&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;decimal&gt;&gt;&gt; selector )</pre>	

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage(View<Double>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) : LiveAverage(View<Double>) Method

A view containing the values to calculate the average of.

Computes the average of a view of **System.Double** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Double) _</pre>	



```
) As AggregationView(Of Double,Double)
```

```
C#
```

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<double,double> LiveAverage(  
    View<double> source  
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage(View<Nullable<Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage(View<Nullable<Double>>) Method

A view containing the values to calculate the average of.

Computes the average of a view of nullable **System.Double** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Nullable(Of Double)) _ ) As AggregationView(Of Nullable(Of Double),Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;double&gt;,Nullable&lt;double&gt;&gt; LiveAverage(     View&lt;Nullable&lt;double&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the average of.

### Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAverage(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Double)) _
) As AggregationView(Of TSource,Double)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,double> LiveAverage<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,double>> selector
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>)  
Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of nullable **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre> &lt;System.Runtime.CompilerServices.ExtensionAttribute()&gt; Public Overloads Shared Function LiveAverage(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Double))) _ ) As AggregationView(Of TSource,Nullable(Of Double)) </pre>	
C#	
<pre> [System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;double&gt;&gt; LiveAverage&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;double&gt;&gt;&gt; selector ) </pre>	

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveAverage(View<Single>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) : LiveAverage(View<Single>) Method

A view containing the values to calculate the average of.

Computes the average of a view of **System.Single** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;&gt; Public Overloads Shared Function LiveAverage( _     ByVal source As View(Of Single) _ ) As AggregationView(Of Single,Double)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;float,double&gt; LiveAverage(     View&lt;float&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the average of.

### Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage(View<Nullable<Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage(View<Nullable<Single>>) Method

A view containing the values to calculate the average of.

Computes the average of a view of nullable **System.Single** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAverage( _
    ByVal source As View(Of Nullable(Of Single)) _
) As AggregationView(Of Nullable(Of Single),Nullable(Of Double))
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<Nullable<float>,Nullable<double>> LiveAverage(
```

```
View<Nullable<float>>> source
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty or contains only nulls, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :

LiveAverage<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of **System.Single** values that are obtained by invoking a transform function on each element of the source view.



## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveAverage(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Single)) _
) As AggregationView(Of TSource,Double)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,double> LiveAverage<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,float>> selector
)
```

### Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

### Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant

algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

`LiveAverage<TSource>(View<TSource>, Expression<Func<TSource, Nullable<Single>>>)` Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveAverage Method](#) :  
`LiveAverage<TSource>(View<TSource>, Expression<Func<TSource, Nullable<Single>>>)` Method

The type of the elements of *source*.

A view containing the values to calculate the average of.

A transform function to apply to each element.

Computes the average of a view of nullable **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveAverage(Of TSource)( _  
    ByVal source As View(Of TSource), _  
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of  
TSource, Nullable(Of Single))) _  
) As AggregationView(Of TSource, Nullable(Of Double))
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource, Nullable<double>> LiveAverage<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource, Nullable<float>>> selector
```

```
)
```

## Parameters

*source*

A view containing the values to calculate the average of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the average of the values.

## Remarks

If the *source* is empty, the average value is null.

It is possible to use standard LINQ query operator **Average** instead of **LiveAverage**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Average** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveAverage** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

# LiveCount Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : LiveCount Method

A view representing the number of elements in a view.

## Overload List

Overload	Description
<a href="#">LiveCount&lt;T&gt;(View&lt;T&gt;)</a>	A view representing the number of elements in a view.
<a href="#">LiveCount&lt;T&gt;(View&lt;T&gt;,Expression&lt;Func&lt;T,Boolean&gt;&gt;)</a>	A view representing the number of elements of the specified view satisfying a condition.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)

[LiveCount<T>\(View<T>\) Method](#)  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveCount Method](#) : LiveCount<T>(View<T>) Method

The type of the elements of *source*.

A view that contains elements to be counted.

A view representing the number of elements in a view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; <a href="#">Public Overloads Shared Function</a> LiveCount(<a href="#">Of</a> T)( _</pre>	

```
ByVal source As View(Of T) _
) As AggregationView(Of T,Integer)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<T,int> LiveCount<T>(
    View<T> source
)
```

## Parameters

*source*

A view that contains elements to be counted.

## Type Parameters

*T*

The type of the elements of *source*.

## Remarks

It is possible to use standard LINQ query operator **Count** instead of **LiveCount**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Count** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveCount** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveCount<T>(View<T>,Expression<Func<T,Boolean>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveCount Method](#) :

LiveCount<T>(View<T>,Expression<Func<T,Boolean>>) Method

The type of the elements of *source*.

A view that contains elements to be tested and counted.

A function to test each element for a condition.

A view representing the number of elements of the specified view satisfying a condition.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveCount(Of T)( _     ByVal source As View(Of T), _     ByVal predicate As System.Linq.Expressions.Expression(Of Func(Of T,Boolean))) _ ) As AggregationView(Of T,Integer)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;T,int&gt; LiveCount&lt;T&gt;(     View&lt;T&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;T,bool&gt;&gt; predicate )</pre>	

## Parameters

*source*

A view that contains elements to be tested and counted.

*predicate*

A function to test each element for a condition.

## Type Parameters

*T*

The type of the elements of *source*.

## Remarks

It is possible to use standard LINQ query operator **Count** instead of **LiveCount**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Count** will every time loop through the entire source collection and aggregate it from scratch, whereas

**LiveCount** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

### LiveMax Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : LiveMax Method

Computes the maximum value of a view of **System.Double** values.

## Overload List

Overload	Description
<a href="#">LiveMax(View&lt;Double&gt;)</a>	Computes the maximum value of a view of <b>System.Double</b> values.
<a href="#">LiveMax(View&lt;Nullable&lt;Double&gt;&gt;)</a>	Computes the maximum value of a view of nullable <b>System.Double</b> values.
<a href="#">LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Double&gt;&gt;)</a>	Computes the maximum

	<p>value of a view of</p> <p><b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.</p>
<a href="#">LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Double&gt;&gt;&gt;)</a>	<p>Computes the maximum value of a view of nullable <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.</p>
<a href="#">LiveMax(View&lt;Single&gt;)</a>	<p>Computes the maximum value of a view of <b>System.Single</b> values.</p>
<a href="#">LiveMax(View&lt;Nullable&lt;Single&gt;&gt;)</a>	<p>Computes the maximum value of a view of nullable</p>



	<b>System.Single</b> values.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Single&gt;&gt;)</code>	Computes the maximum value of a view of <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Single&gt;&gt;&gt;)</code>	Computes the maximum value of a view of nullable <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMax&lt;TSource,TResult&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,TResult&gt;&gt;)</code>	Invokes a transform function on each element of a view of elements of a

	generic type and computes the maximum resulting value.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;)</code>	Computes the maximum value of a view of elements of a generic type.
<code>LiveMax(View&lt;Int32&gt;)</code>	Computes the maximum value of a view of <b>System.Int32</b> values.
<code>LiveMax(View&lt;Nullable&lt;Int32&gt;&gt;)</code>	Computes the maximum value of a view of nullable <b>System.Int32</b> values.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int32&gt;&gt;)</code>	Computes the maximum value of a view of <b>System.Int32</b> values that are obtained by invoking a transform function on

	each element of the source view.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int32&gt;&gt;&gt;)</code>	Computes the maximum value of a view of nullable <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMax(View&lt;Decimal&gt;)</code>	Computes the maximum value of a view of <b>System.Decimal</b> values.
<code>LiveMax(View&lt;Nullable&lt;Decimal&gt;&gt;)</code>	Computes the maximum value of a view of nullable <b>System.Decimal</b> values.
<code>LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Decimal&gt;&gt;)</code>	Computes the maximum value of a view of <b>System.Decimal</b>

	<p><b>al</b> values that are obtained by invoking a transform function on each element of the source view.</p>
<p><a href="#">LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Decimal&gt;&gt;&gt;)</a></p>	<p>Computes the maximum value of a view of nullable <b>System.Decimal</b> values that are obtained by invoking a transform function on each element of the source view.</p>
<p><a href="#">LiveMax(View&lt;Int64&gt;)</a></p>	<p>Computes the maximum value of a view of <b>System.Int64</b> values.</p>
<p><a href="#">LiveMax(View&lt;Nullable&lt;Int64&gt;&gt;)</a></p>	<p>Computes the maximum value of a view of nullable <b>System.Int64</b> values.</p>

<a href="#">LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int64&gt;&gt;)</a>	<p>Computes the maximum value of a view of <b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source view.</p>
<a href="#">LiveMax&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int64&gt;&gt;&gt;)</a>	<p>Computes the maximum value of a view of nullable <b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source view.</p>

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)

## LiveMax(View<Double>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Double>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of **System.Double** values.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMax( _  
    ByVal source As View(Of Double) _  
) As AggregationView(Of Double,Double)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<double,double> LiveMax(  
    View<double> source  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax(View<Nullable<Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax(View<Nullable<Double>>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of nullable **System.Double** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax( _     ByVal source As View(Of Nullable(Of Double)) _ ) As AggregationView(Of Nullable(Of Double),Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;double&gt;,Nullable&lt;double&gt;&gt; LiveMax(     View&lt;Nullable&lt;double&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

### Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMax(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Double))) _
) As AggregationView(Of TSource,Double)
```

C#



```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,double> LiveMax<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,double>> selector  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

[LiveMax<TSource>\(View<TSource>,Expression<Func<TSource,Nullable<Double>>>\)](#) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

[LiveMax<TSource>\(View<TSource>,Expression<Func<TSource,Nullable<Double>>>\)](#) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of nullable **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Double))) _ ) As AggregationView(Of TSource,Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;double&gt;&gt; LiveMax&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;double&gt;&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax(View<Single>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Single>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of **System.Single** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMax( _  
    ByVal source As View(Of Single) _
```

```
) As AggregationView(Of Single,Single)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<float,float> LiveMax(  
    View<float> source  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMax(View<Nullable<Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax(View<Nullable<Single>>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of nullable **System.Single** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax( _     ByVal source As View(Of Nullable(Of Single)) _ ) As AggregationView(Of Nullable(Of Single), Nullable(Of Single))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;float&gt;, Nullable&lt;float&gt;&gt; LiveMax(     View&lt;Nullable&lt;float&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

### Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

`LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Single>>)` Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

`LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Single>>)` Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMax(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Single)) _
) As AggregationView(Of TSource,Single)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,float> LiveMax<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,float>> selector
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of nullable **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre> &lt;System.Runtime.CompilerServices.ExtensionAttribute()&gt; Public Overloads Shared Function LiveMax(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Single)))) _ ) As AggregationView(Of TSource,Nullable(Of Single)) </pre>	
C#	
<pre> [System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;float&gt;&gt; LiveMax&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;float&gt;&gt;&gt; selector ) </pre>	

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax<TSource,TResult>(View<TSource>,Expression<Func<TSource,TResult>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource,TResult>(View<TSource>,Expression<Func<TSource,TResult>>) Method

The type of the elements of *source*.

The type of the value returned by *selector*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Invokes a transform function on each element of a view of elements of a generic type and computes the maximum resulting value.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax     (Of TSource,TResult)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,TResult)) _ ) As AggregationView(Of TSource,TResult)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,TResult&gt; LiveMax&lt;TSource,TResult&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,TResult&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

*TResult*

The type of the value returned by *selector*.

## Return Value

A view representing the maximum of the values.

## Remarks

If type *TResult* implements **System.IComparable<T>**, this method uses that implementation to compare values. Otherwise, if type *TResult* implements **System.IComparable**, that implementation is used to compare values.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveMax<TSource>(View<TSource>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of elements of a generic type.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMax(Of TSource)( _
    ByVal source As View(Of TSource) _
) As AggregationView(Of TSource,TSource)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,TSource> LiveMax<TSource>(
    View<TSource> source
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If type *TSource* implements **System.IComparable`1**, this method uses that implementation to compare values. Otherwise, if type *TSource* implements **System.IComparable**, that implementation is used to compare values.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax(View<Int32>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Int32>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax( _     ByVal source As View(Of Integer) _ ) As AggregationView(Of Integer,Integer)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;int,int&gt; LiveMax(     View&lt;int&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax(View<Nullable<Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Nullable<Int32>>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of nullable **System.Int32** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMax( _  
    ByVal source As View(Of Nullable(Of Integer)) _
```

```
) As AggregationView(Of Nullable(Of Integer),Nullable(Of Integer))
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<Nullable<int>,Nullable<int>> LiveMax(  
    View<Nullable<int>> source  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMax(Of TSource)( _  
    ByVal source As View(Of TSource), _  
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of  
TSource,Integer))) _  
) As AggregationView(Of TSource,Integer)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,int> LiveMax<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,int>> selector  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of nullable **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Integer)))) _ ) As AggregationView(Of TSource,Nullable(Of Integer))</pre>	
C#	



```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource, Nullable<int>> LiveMax<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource, Nullable<int>>> selector  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveMax(View<Decimal>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Decimal>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of **System.Decimal** values.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMax( _  
    ByVal source As View(Of Decimal) _  
) As AggregationView(Of Decimal,Decimal)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<decimal,decimal> LiveMax(  
    View<decimal> source  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax(View<Nullable<Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax(View<Nullable<Decimal>>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of nullable **System.Decimal** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;&gt; Public Overloads Shared Function LiveMax( _     ByVal source As View(Of Nullable(Of Decimal)) _ ) As AggregationView(Of Nullable(Of Decimal), Nullable(Of Decimal))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;decimal&gt;, Nullable&lt;decimal&gt;&gt; LiveMax(     View&lt;Nullable&lt;decimal&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

### Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMax(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Decimal))) _
```

```
) As AggregationView(Of TSource,Decimal)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,decimal> LiveMax<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,decimal>> selector  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

`LiveMax<TSource>(View<TSource>, Expression<Func<TSource, Nullable<Decimal>>>) Method`

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

`LiveMax<TSource>(View<TSource>, Expression<Func<TSource, Nullable<Decimal>>>) Method`

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of nullable **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMax(Of TSource)( _  
    ByVal source As View(Of TSource), _  
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of  
TSource, Nullable(Of Decimal))) _  
) As AggregationView(Of TSource, Nullable(Of Decimal))
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource, Nullable<decimal>> LiveMax<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource, Nullable<decimal>>>  
    selector  
)
```

### Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMax(View<Int64>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Int64>) Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of **System.Int64** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMax( _
```

```
ByVal source As View(Of Long) _  
) As AggregationView(Of Long,Long)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<long,long> LiveMax(  
    View<long> source  
)
```

## Parameters

*source*

A view containing the values to determine the maximum value of.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMax(View<Nullable<Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) : LiveMax(View<Nullable<Int64>>)



Method

A view containing the values to determine the maximum value of.

Computes the maximum value of a view of nullable **System.Int64** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax( _     ByVal source As View(Of Nullable(Of Long)) _ ) As AggregationView(Of Nullable(Of Long),Nullable(Of Long))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;long&gt;,Nullable&lt;long&gt;&gt; LiveMax(     View&lt;Nullable&lt;long&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the maximum value of.

### Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [LiveViewExtensions Class](#)
- [LiveViewExtensions Members](#)
- [Overload List](#)

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

# Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMax(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Long)) _ ) As AggregationView(Of TSource,Long)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,long&gt; LiveMax&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,long&gt;&gt; selector )</pre>	

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

**LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>)** Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMax Method](#) :

**LiveMax<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>)** Method

The type of the elements of *source*.

A view containing the values to determine the maximum value of.

A transform function to apply to each element.

Computes the maximum value of a view of nullable **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre> &lt;System.Runtime.CompilerServices.ExtensionAttribute()&gt; Public Overloads Shared Function LiveMax(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Long)))) _ ) As AggregationView(Of TSource,Nullable(Of Long)) </pre>	
C#	
<pre> [System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;long&gt;&gt; LiveMax&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;long&gt;&gt;&gt; selector ) </pre>	

## Parameters

*source*

A view containing the values to determine the maximum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the maximum of the values.

## Remarks

If the *source* is empty or contains only nulls, the maximum value is null.

It is possible to use standard LINQ query operator **Max** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Max** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

- [LiveViewExtensions Class](#)
- [LiveViewExtensions Members](#)
- [Overload List](#)

LiveMin Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : LiveMin Method

Computes the minimum value of a view of nullable **System.Double** values that are obtained by invoking a transform function on each element of the source view.

Overload List

Overload	Description
<a href="#">LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Double&gt;&gt;&gt;&gt;)</a>	Computes the minimum value of a view of nullable <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.
<a href="#">LiveMin(View&lt;Single&gt;)</a>	Computes the minimum value of a view of <b>System.Single</b> values.

LiveMin(View<Nullable<Single>>)	Computes the minimum value of a view of nullable <b>System.Single</b> values.
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Single>>)	Computes the minimum value of a view of <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>>)	Computes the minimum value of a view of nullable <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.
LiveMin<TSource,TResult>(View<TSource>,Expression<Func<TSource,TResult>>>)	Invokes a transform function on

	each element of a view of elements of a generic type and computes the minimum resulting value.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;)</code>	Computes the minimum value of a view of elements of a generic type.
<code>LiveMin(View&lt;Int32&gt;)</code>	Computes the minimum value of a view of <b>System.Int32</b> values.
<code>LiveMin(View&lt;Nullable&lt;Int32&gt;&gt;)</code>	Computes the minimum value of a view of nullable <b>System.Int32</b> values.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int32&gt;&gt;)</code>	Computes the minimum value of a view of <b>System.Int32</b> values that are obtained by invoking a transform function on

	each element of the source view.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int32&gt;&gt;&gt;)</code>	Computes the minimum value of a view of nullable <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMin(View&lt;Decimal&gt;)</code>	Computes the minimum value of a view of <b>System.Decimal</b> values.
<code>LiveMin(View&lt;Nullable&lt;Decimal&gt;&gt;)</code>	Computes the minimum value of a view of nullable <b>System.Decimal</b> values.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Decimal&gt;&gt;)</code>	Computes the minimum value of a view of <b>System.Decimal</b> values that are obtained



	by invoking a transform function on each element of the source view.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Decimal&gt;&gt;&gt;)</code>	Computes the minimum value of a view of nullable <b>System.Decimal</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveMin(View&lt;Int64&gt;)</code>	Computes the minimum value of a view of <b>System.Int64</b> values.
<code>LiveMin(View&lt;Nullable&lt;Int64&gt;&gt;)</code>	Computes the minimum value of a view of nullable <b>System.Int64</b> values.
<code>LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int64&gt;&gt;)</code>	Computes the minimum value of a view of

	<p><b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source view.</p>
<p><a href="#">LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int64&gt;&gt;&gt;)</a></p>	<p>Computes the minimum value of a view of nullable <b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source view.</p>
<p><a href="#">LiveMin(View&lt;Double&gt;)</a></p>	<p>Computes the minimum value of a view of <b>System.Double</b> values.</p>
<p><a href="#">LiveMin(View&lt;Nullable&lt;Double&gt;&gt;)</a></p>	<p>Computes the minimum value of a view of nullable <b>System.Double</b> values.</p>

<a href="#">LiveMin&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Double&gt;&gt;)</a>	Computes the minimum value of a view of <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.
--	--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Double>>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of nullable **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
```

```
Public Overloads Shared Function LiveMin(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Nullable(Of Double)))) _
) As AggregationView(Of TSource,Nullable(Of Double))
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,Nullable<double>> LiveMin<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,Nullable<double>>>
selector
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

- [LiveViewExtensions Class](#)
- [LiveViewExtensions Members](#)
- [Overload List](#)

LiveMin(View<Single>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Single>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of **System.Single** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Single) _ ) As AggregationView(Of Single,Single)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;float,float&gt; LiveMin(     View&lt;float&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

### Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin(View<Nullable<Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Nullable<Single>>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of nullable **System.Single** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Nullable(Of Single)) _ ) As AggregationView(Of Nullable(Of Single),Nullable(Of Single))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;float&gt;,Nullable&lt;float&gt;&gt; LiveMin(     View&lt;Nullable&lt;float&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Single>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMin(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Single)) _
) As AggregationView(Of TSource,Single)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,float> LiveMin<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,float>> selector
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of nullable **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;&gt; Public Overloads Shared Function LiveMin(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Single)))) _ ) As AggregationView(Of TSource,Nullable(Of Single))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;float&gt;&gt; LiveMin&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;float&gt;&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMin<TSource,TResult>(View<TSource>,Expression<Func<TSource,TResult>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource,TResult>(View<TSource>,Expression<Func<TSource,TResult>>) Method

The type of the elements of *source*.

The type of the value returned by *selector*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Invokes a transform function on each element of a view of elements of a generic type and computes the minimum resulting value.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin     (Of TSource,TResult)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,TResult)) _ ) As AggregationView(Of TSource,TResult)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,TResult&gt; LiveMin&lt;TSource,TResult&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,TResult&gt;&gt; selector )</pre>	

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

*TResult*

The type of the value returned by *selector*.

## Return Value

A view representing the minimum of the values.

## Remarks

If type *TResult* implements **System.IComparable`1**, this method uses that implementation to compare values. Otherwise, if type *TResult* implements **System.IComparable**, that implementation is used to compare values.

It is possible to use standard LINQ query operator **Min** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of elements of a generic type.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMin(Of TSource)( _  
    ByVal source As View(Of TSource) _  
) As AggregationView(Of TSource,TSource)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,TSource> LiveMin<TSource>(  
    View<TSource> source  
)
```

### Parameters

*source*

A view containing the values to determine the minimum value of.

### Type Parameters

*TSource*

The type of the elements of *source*.

### Return Value

A view representing the minimum of the values.

## Remarks

If type *TSource* implements **System.IComparable`1**, this method uses that implementation to compare values. Otherwise, if type *TSource* implements **System.IComparable**, that implementation is used to compare values.

It is possible to use standard LINQ query operator **Min** instead of **LiveMax**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMax** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveMin(View<Int32>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Int32>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Integer) _ ) As AggregationView(Of Integer,Integer)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;int,int&gt; LiveMin(     View&lt;int&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

### Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin(View<Nullable<Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Nullable<Int32>>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of nullable **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Nullable(Of Integer)) _ ) As AggregationView(Of Nullable(Of Integer),Nullable(Of Integer))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;int&gt;,Nullable&lt;int&gt;&gt; LiveMin(     View&lt;Nullable&lt;int&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

### Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Integer))) _ ) As AggregationView(Of TSource,Integer)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()]</pre>	



```
public static AggregationView<TSource,int> LiveMin<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,int>> selector
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :  
LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of nullable **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Integer))) _ ) As AggregationView(Of TSource,Nullable(Of Integer))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;int&gt;&gt; LiveMin&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;int&gt;&gt;&gt; selector )</pre>	

Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

Type Parameters

*TSource*

The type of the elements of *source*.

Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin(View<Decimal>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Decimal>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of **System.Decimal** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMin( _  
    ByVal source As View(Of Decimal) _  
) As AggregationView(Of Decimal,Decimal)
```

C#

```
[System.Runtime.CompilerServices.Extension()  
public static AggregationView<decimal,decimal> LiveMin(
```

```
View<decimal> source
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMin(View<Nullable<Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin(View<Nullable<Decimal>>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of nullable **System.Decimal** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Nullable(Of Decimal)) _ ) As AggregationView(Of Nullable(Of Decimal),Nullable(Of Decimal))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;decimal&gt;,Nullable&lt;decimal&gt;&gt; LiveMin(     View&lt;Nullable&lt;decimal&gt;&gt; source )</pre>	

## Parameters

*source*

A view containing the values to determine the minimum value of.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveMin(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Decimal))) _
) As AggregationView(Of TSource,Decimal)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,decimal> LiveMin<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,decimal>> selector
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of nullable **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>>  
Public Overloads Shared Function LiveMin(Of TSource)( _  
    ByVal source As View(Of TSource), _
```

```

    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource, Nullable(Of Decimal))) _
) As AggregationView(Of

```

C#

```

[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource, Nullable<decimal>> LiveMin<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource, Nullable<decimal>>>
selector
)

```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2



## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

#### LiveMin(View<Int64>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Int64>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of **System.Int64** values.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMin( _  
    ByVal source As View(Of Long) _  
) As AggregationView(Of Long,Long)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<long,long> LiveMin(  
    View<long> source  
)
```

### Parameters

*source*

A view containing the values to determine the minimum value of.

### Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the

source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin(View<Nullable<Int64>>) Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Nullable<Int64>>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of nullable **System.Int64** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Nullable(Of Long)) _ ) As AggregationView(Of Nullable(Of Long),Nullable(Of Long))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;long&gt;,Nullable&lt;long&gt;&gt; LiveMin(     View&lt;Nullable&lt;long&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>>
Public Overloads Shared Function LiveMin(Of TSource)( _
    ByVal source As View(Of TSource), _
```

```

    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource, Long)) _
) As AggregationView(Of TSource, Long)

```

C#

```

[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource, long> LiveMin<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource, long>> selector
)

```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [LiveViewExtensions Class](#)
- [LiveViewExtensions Members](#)
- [Overload List](#)

`LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>)` Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :  
`LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>)` Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of nullable **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Long))) _     ) As AggregationView(Of TSource,Nullable(Of Long))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;long&gt;&gt; LiveMin&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;long&gt;&gt;&gt; selector )</pre>	

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin(View<Double>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) : LiveMin(View<Double>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of **System.Double** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMin( _
```

```
ByVal source As View(Of Double) _  
) As AggregationView(Of Double,Double)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<double,double> LiveMin(  
    View<double> source  
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveMin(View<Nullable<Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin(View<Nullable<Double>>) Method

A view containing the values to determine the minimum value of.

Computes the minimum value of a view of nullable **System.Double** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveMin( _     ByVal source As View(Of Nullable(Of Double)) _ ) As AggregationView(Of Nullable(Of Double),Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;double&gt;,Nullable&lt;double&gt;&gt; LiveMin(     View&lt;Nullable&lt;double&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to determine the minimum value of.

### Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty or contains only nulls, the minimum value is null.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveMin Method](#) :

LiveMin<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

The type of the elements of *source*.

A view containing the values to determine the minimum value of.

A transform function to apply to each element.

Computes the minimum value of a view of **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveMin(Of TSource)( _  
    ByVal source As View(Of TSource), _  
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of  
TSource,Double)) _  
) As AggregationView(Of TSource,Double)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,double> LiveMin<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,double>> selector  
)
```

## Parameters

*source*

A view containing the values to determine the minimum value of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the minimum of the values.

## Remarks

If the *source* is empty, an **System.InvalidOperationException** is thrown.

It is possible to use standard LINQ query operator **Min** instead of **LiveMin**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Min** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveMin** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveSum Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) : LiveSum Method

Computes the sum of a view of **System.Int32** values.

## Overload List

Overload	Description
<a href="#">LiveSum(View&lt;Int32&gt;)</a>	Computes the sum of a view of <b>System.Int32</b>

	values.
<a href="#">LiveSum(View&lt;Nullable&lt;Int32&gt;&gt;)</a>	Computes the sum of a view of nullable <b>System.Int32</b> values.
<a href="#">LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int32&gt;&gt;)</a>	Computes the sum of a view of <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.
<a href="#">LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int32&gt;&gt;&gt;)</a>	Computes the sum of a view of nullable <b>System.Int32</b> values that are obtained by invoking a transform function on each element of the source view.
<a href="#">LiveSum(View&lt;Decimal&gt;)</a>	Computes the sum of a view of

	<b>System.Decimal</b> values.
<a href="#">LiveSum(View&lt;Nullable&lt;Decimal&gt;&gt;)</a>	Computes the sum of a view of nullable <b>System.Decimal</b> values.
<a href="#">LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Decimal&gt;&gt;)</a>	Computes the sum of a view of <b>System.Decimal</b> values that are obtained by invoking a transform function on each element of the source view.
<a href="#">LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Decimal&gt;&gt;&gt;)</a>	Computes the sum of a view of nullable <b>System.Decimal</b> values that are obtained by invoking a transform function on each element of the source view.
<a href="#">LiveSum(View&lt;Int64&gt;)</a>	Computes the sum of a view

	of <b>System.Int64</b> values.
<code>LiveSum(View&lt;Nullable&lt;Int64&gt;&gt;)</code>	Computes the sum of a view of nullable <b>System.Int64</b> values.
<code>LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Int64&gt;&gt;)</code>	Computes the sum of a view of <b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Int64&gt;&gt;&gt;)</code>	Computes the sum of a view of nullable <b>System.Int64</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveSum(View&lt;Double&gt;)</code>	Computes the

	sum of a view of <b>System.Double</b> values.
<code>LiveSum(View&lt;Nullable&lt;Double&gt;&gt;)</code>	Computes the sum of a view of nullable <b>System.Double</b> values.
<code>LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Double&gt;&gt;)</code>	Computes the sum of a view of <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.
<code>LiveSum&lt;TSource&gt;(View&lt;TSource&gt;,Expression&lt;Func&lt;TSource,Nullable&lt;Double&gt;&gt;&gt;)</code>	Computes the sum of a view of nullable <b>System.Double</b> values that are obtained by invoking a transform function on each element of the source view.

LiveSum(View<Single>)	Computes the sum of a view of <b>System.Single</b> values.
LiveSum(View<Nullable<Single>>)	Computes the sum of a view of nullable <b>System.Single</b> values.
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Single>>)	Computes the sum of a view of <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source view.
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>)	Computes the sum of a view of nullable <b>System.Single</b> values that are obtained by invoking a transform function on each element of the source

	view.
--	-------

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)

LiveSum(View<Int32>) Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Int32>) Method

A view containing the values to calculate the sum of.  
Computes the sum of a view of **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Integer) _ ) As AggregationView(Of Integer,Integer)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;int,int&gt; LiveSum(     View&lt;int&gt; source )</pre>	

### Parameters

*source*  
A view containing the values to calculate the sum of.

### Return Value

A view representing the sum of the values.



## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum(View<Nullable<Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Nullable<Int32>>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of nullable **System.Int32** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Nullable(Of Integer)) _ ) As AggregationView(Of Nullable(Of Integer),Nullable(Of Integer))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;int&gt;,Nullable&lt;int&gt;&gt; LiveSum(</pre>	

```
View<Nullable<int>> source
)
```

## Parameters

*source*

A view containing the values to calculate the sum of.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Int32>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveSum(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Integer))) _
) As AggregationView(Of TSource,Integer)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,int> LiveSum<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,int>> selector
)
```

### Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [LiveViewExtensions Class](#)
- [LiveViewExtensions Members](#)
- [Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int32>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of nullable **System.Int32** values that are obtained by invoking a transform function on each element of the source view.

# Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Integer)))) _ ) As AggregationView(Of TSource,Nullable(Of Integer))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;int&gt;&gt; LiveSum&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;int&gt;&gt;&gt; selector )</pre>	

## Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveSum(View<Decimal>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Decimal>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of **System.Decimal** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Decimal) _ ) As AggregationView(Of Decimal,Decimal)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;decimal,decimal&gt; LiveSum(     View&lt;decimal&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the sum of.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveSum(View<Nullable<Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum(View<Nullable<Decimal>>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of nullable **System.Decimal** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveSum( _
    ByVal source As View(Of Nullable(Of Decimal)) _
) As AggregationView(Of Nullable(Of Decimal),Nullable(Of Decimal))
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<Nullable<decimal>,Nullable<decimal>> LiveSum(
    View<Nullable<decimal>> source
)
```

## Parameters

*source*

A view containing the values to calculate the sum of.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [LiveViewExtensions Class](#)
- [LiveViewExtensions Members](#)
- [Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Decimal>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

# Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Decimal))) _ ) As AggregationView(Of TSource,Decimal)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,decimal&gt; LiveSum&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,decimal&gt;&gt; selector )</pre>	

## Parameters



*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Decimal>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of nullable **System.Decimal** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Decimal)))) _ ) As AggregationView(Of TSource,Nullable(Of Decimal))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;decimal&gt;&gt; LiveSum&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;decimal&gt;&gt;&gt; selector )</pre>	

## Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum(View<Int64>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Int64>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of **System.Int64** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Long) _ ) As AggregationView(Of Long,Long)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;long,long&gt; LiveSum(     View&lt;long&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the sum of.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum(View<Nullable<Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Nullable<Int64>>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of nullable **System.Int64** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Nullable(Of Long)) _ ) As AggregationView(Of Nullable(Of Long), Nullable(Of Long))</pre>	

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<Nullable<long>,Nullable<long>> LiveSum(  
    View<Nullable<long>> source  
)
```

### Parameters

*source*

A view containing the values to calculate the sum of.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Int64>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Long)) _ ) As AggregationView(Of TSource,Long)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,long&gt; LiveSum&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,long&gt;&gt; selector )</pre>	

### Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>) Method  
[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :  
LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Int64>>>) Method

- The type of the elements of *source*.
- A view containing the values to calculate the sum of.
- A transform function to apply to each element.
- Computes the sum of a view of nullable **System.Int64** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Long))) _ ) As AggregationView(Of TSource,Nullable(Of Long))</pre>	
C#	

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource, Nullable<long>> LiveSum<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource, Nullable<long>>> selector  
)
```

## Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference



[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## LiveSum(View<Double>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Double>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of **System.Double** values.

## Syntax

### Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveSum( _  
    ByVal source As View(Of Double) _  
) As AggregationView(Of Double,Double)
```

### C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<double,double> LiveSum(  
    View<double> source  
)
```

## Parameters

*source*

A view containing the values to calculate the sum of.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum(View<Nullable<Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum(View<Nullable<Double>>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of nullable **System.Double** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Nullable(Of Double)) _ ) As AggregationView(Of Nullable(Of Double),Nullable(Of Double))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;double&gt;,Nullable&lt;double&gt;&gt; LiveSum(     View&lt;Nullable&lt;double&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the sum of.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Double>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveSum(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Double))) _
```

```
) As AggregationView(Of TSource,Double)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource,double> LiveSum<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource,double>> selector  
)
```

### Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

### Type Parameters

*TSource*

The type of the elements of *source*.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

`LiveSum<TSource>(View<TSource>, Expression<Func<TSource, Nullable<Double>>>)` Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

`LiveSum<TSource>(View<TSource>, Expression<Func<TSource, Nullable<Double>>>)` Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of nullable **System.Double** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveSum(Of TSource)( _  
    ByVal source As View(Of TSource), _  
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of  
TSource, Nullable(Of Double))) _  
) As AggregationView(Of TSource, Nullable(Of Double))
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static AggregationView<TSource, Nullable<double>> LiveSum<TSource>(  
    View<TSource> source,  
    System.Linq.Expressions.Expression<Func<TSource, Nullable<double>>>  
    selector  
)
```

### Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum(View<Single>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) : LiveSum(View<Single>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of **System.Single** values.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function LiveSum( _
```

```

    ByVal source As View(Of Single) _
) As AggregationView(Of Single,Single)

C#

[System.Runtime.CompilerServices.Extension()]
public static AggregationView<float,float> LiveSum(
    View<float> source
)

```

## Parameters

*source*

A view containing the values to calculate the sum of.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum(View<Nullable<Single>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :  
 LiveSum(View<Nullable<Single>>) Method

A view containing the values to calculate the sum of.

Computes the sum of a view of nullable **System.Single** values.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function LiveSum( _     ByVal source As View(Of Nullable(Of Single)) _ ) As AggregationView(Of Nullable(Of Single),Nullable(Of Single))</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;Nullable&lt;float&gt;,Nullable&lt;float&gt;&gt; LiveSum(     View&lt;Nullable&lt;float&gt;&gt; source )</pre>	

### Parameters

*source*

A view containing the values to calculate the sum of.

### Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



## Reference

[LiveViewExtensions Class](#)

[LiveViewExtensions Members](#)

[Overload List](#)

`LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Single>>)` Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

`LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Single>>)` Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Overloads Shared Function LiveSum(Of TSource)( _
    ByVal source As View(Of TSource), _
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of
TSource,Single)) _
) As AggregationView(Of TSource,Single)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static AggregationView<TSource,float> LiveSum<TSource>(
    View<TSource> source,
    System.Linq.Expressions.Expression<Func<TSource,float>> selector
)
```

## Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method

[C1.LiveLinq Namespace](#) > [LiveViewExtensions Class](#) > [LiveSum Method](#) :

LiveSum<TSource>(View<TSource>,Expression<Func<TSource,Nullable<Single>>>) Method

The type of the elements of *source*.

A view containing the values to calculate the sum of.

A transform function to apply to each element.

Computes the sum of a view of nullable **System.Single** values that are obtained by invoking a transform function on each element of the source view.

## Syntax

Visual Basic (Declaration)	
<pre> &lt;System.Runtime.CompilerServices.ExtensionAttribute()&gt; Public Overloads Shared Function LiveSum(Of TSource)( _     ByVal source As View(Of TSource), _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of TSource,Nullable(Of Single))) _ ) As AggregationView(Of TSource,Nullable(Of Single)) </pre>	
C#	
<pre> [System.Runtime.CompilerServices.Extension()] public static AggregationView&lt;TSource,Nullable&lt;float&gt;&gt; LiveSum&lt;TSource&gt;(     View&lt;TSource&gt; source,     System.Linq.Expressions.Expression&lt;Func&lt;TSource,Nullable&lt;float&gt;&gt;&gt; selector ) </pre>	

## Parameters

*source*

A view containing the values to calculate the sum of.

*selector*

A transform function to apply to each element.

## Type Parameters

*TSource*

The type of the elements of *source*.

## Return Value

A view representing the sum of the values.

## Remarks

If the *source* is empty or contains only nulls, the sum value is zero.

It is possible to use standard LINQ query operator **Sum** instead of **LiveSum**. Both are "live" in the sense that they are recomputed automatically when any change occurs in the source. The difference is that **Sum** will every time loop through the entire source collection and aggregate it from scratch, whereas **LiveSum** will use a more performant algorithm, will maintain its value incrementally, processing only those source items that actually changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[LiveViewExtensions Class](#)  
[LiveViewExtensions Members](#)  
[Overload List](#)

## SourceChangeEventArgs<T>

[C1.LiveLinq Namespace](#) : SourceChangeEventArgs<T> Class

Type of the changed object.

Provides data for the [IObservableSource<T>.Changed](#) event.

## Object Model

SourceChangeEventArgs<T>

## Syntax

Visual Basic (Declaration)

```
Public Class SourceChangeEventArgs(Of T)  
    Inherits System.EventArgs
```

C#

```
public class SourceChangeEventArgs<T> : System.EventArgs
```

## Type Parameters

*T*

Type of the changed object.

## Inheritance Hierarchy

System.Object  
  System.EventArgs  
    **C1.LiveLinq.SourceChangeEventArgs<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Members](#)  
[C1.LiveLinq Namespace](#)

### Overview

[C1.LiveLinq Namespace](#) : [SourceChangeEventArgs<T>](#) Class

Type of the changed object.

Provides data for the [IObservableSource<T>.Changed](#) event.

## Object Model

[SourceChangeEventArgs<T>](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Class SourceChangeEventArgs(Of T)     Inherits System.EventArgs</pre>	
C#	
<pre>public class SourceChangeEventArgs&lt;T&gt; : System.EventArgs</pre>	

## Type Parameters

*T*

Type of the changed object.

## Inheritance Hierarchy

System.Object  
    System.EventArgs  
        **C1.LiveLinq.SourceChangeEventArgs<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Members](#)  
[C1.LiveLinq Namespace](#)


## Members

[Properties](#) [Methods](#)

[C1.LiveLinq Namespace](#) : [SourceChangeEventArgs<T>](#) Class




The following tables list the members exposed by [SourceChangeEventArgs<T>](#).

## Public Constructors

	Name	Description
	<a href="#">SourceChangeEventArgs&lt;T&gt; Constructor</a>	Initializes a new instance of the <a href="#">SourceChangeEventArgs&lt;T&gt;</a> class.


[Top](#)

## Public Properties

	Name	Description
	<a href="#">ChangeType</a>	Gets the type of change.
	<a href="#">Item</a>	Gets the object that is being changed.
	<a href="#">Ordinal</a>	Gets the ordinal position of the collection item that is being changed.

[Top](#)

## Public Methods

	Name	Description
	<a href="#">Equals</a>	Overloaded. Determines whether the specified object is equal to the

		current object.
⇒	<a href="#">GetHashCode</a>	Return a hash code for this instance.
⇒	<a href="#">ToString</a>	Returns a string that represents this instance of <a href="#">SourceChangeEventArgs&lt;T&gt;</a> .

[Top](#)

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[C1.LiveLinq Namespace](#)

## SourceChangeEventArgs<T> Constructor

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : [SourceChangeEventArgs<T> Constructor](#)

Changed object.

Type of change.

Ordinal position of the changed item.

Initializes a new instance of the [SourceChangeEventArgs<T>](#) class.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function New( _     ByVal item As T, _     ByVal changeType As SourceChangeType, _     ByVal ordinal As System.Integer _ )</pre>	
C#	
<pre>public SourceChangeEventArgs&lt;T&gt;(     T item,     SourceChangeType changeType,     System.int ordinal )</pre>	

### Parameters

*item*

Changed object.

*changeType*

Type of change.

*ordinal*

Ordinal position of the changed item.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[SourceChangeEventArgs<T> Members](#)

## Methods

[C1.LiveLinq Namespace](#) : [SourceChangeEventArgs<T> Class](#)

For a list of all members of this type, see [SourceChangeEventArgs<T> members](#).

## Public Methods

	Name	Description
≡	<a href="#">Equals</a>	Overloaded. Determines whether the specified object is equal to the current object.
≡	<a href="#">GetHashCode</a>	Return a hash code for this instance.
≡	<a href="#">ToString</a>	Returns a string that represents this instance of <a href="#">SourceChangeEventArgs&lt;T&gt;</a> .

[Top](#)

## See Also

### Reference



[SourceChangeEventArgs<T> Class](#)  
[C1.LiveLinq Namespace](#)

**Equals Method**

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : Equals Method

Determines whether the specified object is equal to the current object.

**Overload List**

Overload	Description
<a href="#">Equals(Object)</a>	Determines whether the specified object is equal to the current object.
<a href="#">Equals(SourceChangeEventArgs&lt;T&gt;)</a>	Determines whether the specified object is equal to the current object.

**Requirements**

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

**See Also**

**Reference**

[SourceChangeEventArgs<T> Class](#)  
[SourceChangeEventArgs<T> Members](#)

**Equals(Object) Method**

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) > [Equals Method](#) : Equals(Object) Method

The object to compare with the current object.

Determines whether the specified object is equal to the current object.

**Syntax**

Visual Basic (Declaration)	
<pre>Public Overloads Overrides Function Equals( _     ByVal obj As System.Object _ ) As System.Boolean</pre>	

C#	
<pre>public override System.bool Equals(     System.object obj )</pre>	

## Parameters

*obj*

The object to compare with the current object.

## Return Value

**true** if the specified object is equal to the current one; otherwise, false.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[SourceChangeEventArgs<T> Class](#)  
[SourceChangeEventArgs<T> Members](#)  
[Overload List](#)

Equals(SourceChangeEventArgs<T>) Method

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) > [Equals Method](#) :

Equals(SourceChangeEventArgs<T>) Method

The object to compare with the current object.

Determines whether the specified object is equal to the current object.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Equals( _     ByVal args As SourceChangeEventArgs(Of T) _ ) As System.Boolean</pre>	
C#	

```
public System.bool Equals(  
    SourceChangeEventArgs<T> args  
)
```

## Parameters

*args*

The object to compare with the current object.

## Return Value

**true** if the specified object is equal to the current one; otherwise, false.

## Remarks

Two [SourceChangeEventArgs<T>](#) are considered equal if the values of their [Item](#) and [ChangeType](#) properties are equal.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[SourceChangeEventArgs<T> Class](#)  
[SourceChangeEventArgs<T> Members](#)  
[Overload List](#)

## GetHashCode Method

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : GetHashCode Method

Return a hash code for this instance.

## Syntax

Visual Basic (Declaration)

```
Public Overrides Function GetHashCode() As System.Integer
```

C#

```
public override System.int GetHashCode()
```

## Return Value

A 32-bit signed integer hash code.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[SourceChangeEventArgs<T> Members](#)

### ToString Method

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : ToString Method

Returns a string that represents this instance of [SourceChangeEventArgs<T>](#).

## Syntax

Visual Basic (Declaration)	
<b>Public Overrides Function</b> ToString() <b>As</b> System.String	
C#	
<b>public override</b> System.string ToString()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)




[SourceChangeEventArgs<T> Members](#)

## Properties

[C1.LiveLinq Namespace](#) : [SourceChangeEventArgs<T> Class](#)

For a list of all members of this type, see [SourceChangeEventArgs<T> members](#).

## Public Properties

	Name	Description
	<a href="#">ChangeType</a>	Gets the type of change.
	<a href="#">Item</a>	Gets the object that is being changed.
	<a href="#">Ordinal</a>	Gets the ordinal position of the collection item that is being changed.

[Top](#)

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[C1.LiveLinq Namespace](#)

### ChangeType Property

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : ChangeType Property

Gets the type of change.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property ChangeType As SourceChangeType</code>	
C#	
<code>public SourceChangeType ChangeType {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[SourceChangeEventArgs<T> Members](#)

## Item Property

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : Item Property

Gets the object that is being changed.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Item As T</code>	
C#	
<code>public T Item {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[SourceChangeEventArgs<T> Members](#)

## Ordinal Property

[C1.LiveLinq Namespace](#) > [SourceChangeEventArgs<T> Class](#) : Ordinal Property

Gets the ordinal position of the collection item that is being changed.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Ordinal As System.Integer</code>	
C#	
<code>public System.int Ordinal {get;}</code>	

## Remarks

This property can return -1 (ordinal unknown) if the collection cannot provide this information (if [IObservableSource<T>.SupportsItemOrdinals](#) returns **false**).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[SourceChangeEventArgs<T> Class](#)

[SourceChangeEventArgs<T> Members](#)

## Enumerations

### Order

[C1.LiveLinq Namespace](#) : Order Enumeration

Indicates if a certain order is required in the result collection of an operation.

## Syntax

Visual Basic (Declaration)	
<pre>Public Enum Order     Inherits System.Enum</pre>	
C#	
<pre>public enum Order : System.Enum</pre>	

## Members

Member	Description
<b>Ascending</b>	The resulting collection must be ordered in ascending key order.
<b>Descending</b>	The resulting collection must be ordered in descending key order.
<b>Unordered</b>	No particular order is required in the resulting collection.

## Inheritance Hierarchy

System.Object  
  System.ValueType

System.Enum

**C1.LiveLinq.Order**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.LiveLinq Namespace](#)

## SourceChangeType

[C1.LiveLinq Namespace](#) : SourceChangeType Enumeration

Describes a change occurring in a collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public Enum SourceChangeType     Inherits System.Enum</pre>	
C#	
<pre>public enum SourceChangeType : System.Enum</pre>	

## Members

Member	Description
<b>Add</b>	An item is added to the collection.
<b>Modify</b>	At least one of the properties of an item has changed its value.
<b>Remove</b>	An item is removed from the collection.
<b>Reset</b>	Multiple items have changed in the collection. Indexes must be rebuilt.

## Inheritance Hierarchy



System.Object  
System.ValueType  
System.Enum  
**C1.LiveLinq.SourceChangeType**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.LiveLinq Namespace](#)

## TransactionState

[C1.LiveLinq Namespace](#) : TransactionState Enumeration

Enumeration of the possible states an [ITransaction](#) can be in.

## Syntax

Visual Basic (Declaration)	
<b>Public Enum</b> TransactionState <b>Inherits</b> System.Enum	
C#	
<b>public enum</b> TransactionState : System.Enum	

## Members

Member	Description
<b>Committed</b>	A transaction is committed.
<b>Committing</b>	A transaction is in the process of being committed.
<b>Open</b>	A transaction is open.
<b>RolledBack</b>	A transaction is rolled back.

<b>RollingBack</b>	A transaction is in the process of being rolled back.
--------------------	---

## Inheritance Hierarchy

System.Object  
  System.ValueType  
    System.Enum  
      **C1.LiveLinq.TransactionState**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.LiveLinq Namespace](#)

## Interfaces

### IObservableSource<T>

[C1.LiveLinq Namespace](#) : IObservableSource<T> Interface

The type of the elements in the collection.

LiveLinq 機能（ライブビュー）に必要なメソッドとイベントを提供します。

## Object Model

IObservableSource<T>

## Syntax

Visual Basic (Declaration)

```
Public Interface IObservableSource(Of T)
```

C#

```
public interface IObservableSource<T>
```

## Type Parameters

*T*

The type of the elements in the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IObservableSource<T> Members](#)  
[C1.LiveLinq Namespace](#)

### Overview

[C1.LiveLinq Namespace](#) : IObservableSource<T> Interface

The type of the elements in the collection.

LiveLinq 機能（ライブビュー）に必要なメソッドとイベントを提供します。

## Object Model

IObservableSource<T>

## Syntax

Visual Basic (Declaration)

```
Public Interface IObservableSource(Of T)
```

C#

```
public interface IObservableSource<T>
```

## Type Parameters

*T*

The type of the elements in the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IObservableSource<T> Members](#)  
[C1.LiveLinq Namespace](#)




## Members

[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq Namespace](#) : [IObservableSource<T>](#) Interface


The following tables list the members exposed by [IObservableSource<T>](#).

## Public Properties

	Name	Description
	<a href="#">CreateNew</a>	This delegate is used to create new items. If it is null, a public parameterless constructor of type <b>T</b> is used.
	<a href="#">IsDeletedStateAvailable</a>	Gets a value indicating whether an item of this collection can still return correct property values after it has been deleted from the collection.
	<a href="#">SupportsItemOrdinals</a>	Gets a value that indicates whether this collection is capable of providing the ordinal position of the changed item when it notifies of an item change.


[Top](#)

## Public Methods

	Name	Description
	<a href="#">EnableItemOrdinals</a>	After this method is called, the collection is required to provide <a href="#">SourceChangeEventArgs&lt;T&gt;.Ordinal</a> in event data.

[Top](#)

# Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after an item of the collection or the entire collection has changed.

[Top](#)

# See Also

## Reference


[IObservableSource<T> Interface](#)  
[C1.LiveLinq Namespace](#)

# Methods

[C1.LiveLinq Namespace](#) : [IObservableSource<T> Interface](#)

For a list of all members of this type, see [IObservableSource<T> members](#).

# Public Methods

	Name	Description
	<a href="#">EnableItemOrdinals</a>	After this method is called, the collection is required to provide <a href="#">SourceChangeEventArgs&lt;T&gt;.Ordinal</a> in event data.

[Top](#)

# See Also

## Reference

[IObservableSource<T> Interface](#)  
[C1.LiveLinq Namespace](#)

## EnableItemOrdinals Method

[C1.LiveLinq Namespace](#) > [IObservableSource<T> Interface](#) : [EnableItemOrdinals Method](#)

After this method is called, the collection is required to provide [SourceChangeEventArgs<T>.Ordinal](#) in event data.

# Syntax

Visual Basic (Declaration)	
<b>Sub</b> EnableItemOrdinals()	
C#	
<b>void</b> EnableItemOrdinals()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IObservableSource<T> Interface](#)




[IObservableSource<T> Members](#)

## Properties

[C1.LiveLinq Namespace](#) : IObservableSource<T> Interface

For a list of all members of this type, see [IObservableSource<T> members](#).

## Public Properties

	Name	Description
	<a href="#">CreateNew</a>	This delegate is used to create new items. If it is null, a public parameterless constructor of type <b>T</b> is used.
	<a href="#">IsDeletedStateAvailable</a>	Gets a value indicating whether an item of this collection can still return correct property values after it has been deleted from the collection.
	<a href="#">SupportsItemOrdinals</a>	Gets a value that indicates whether this collection is capable of providing the ordinal position of the changed item when it notifies of an item change.

[Top](#)

## See Also

## Reference

[IObservableSource<T> Interface](#)

[C1.LiveLinq Namespace](#)

## CreateNew Property

[C1.LiveLinq Namespace](#) > [IObservableSource<T> Interface](#) : CreateNew Property

This delegate is used to create new items. If it is null, a public parameterless constructor of type **T** is used.

## Syntax

Visual Basic (Declaration)	
<b>ReadOnly Property</b> CreateNew <b>As</b> System.Func(Of T)	
C#	
System.Func<T> CreateNew { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IObservableSource<T> Interface](#)

[IObservableSource<T> Members](#)

## IsDeletedStateAvailable Property

[C1.LiveLinq Namespace](#) > [IObservableSource<T> Interface](#) : IsDeletedStateAvailable Property

Gets a value indicating whether an item of this collection can still return correct property values after it has been deleted from the collection.

## Syntax

Visual Basic (Declaration)	
<b>ReadOnly Property</b> IsDeletedStateAvailable <b>As</b> System.Boolean	
C#	

```
System.bool IsDeletedStateAvailable {get;}
```

### Property Value

**true** if deleted items can still be used to get property values.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IObservableSource<T> Interface](#)

[IObservableSource<T> Members](#)

### SupportsItemOrdinals Property

[C1.LiveLinq Namespace](#) > [IObservableSource<T> Interface](#) : SupportsItemOrdinals Property

Gets a value that indicates whether this collection is capable of providing the ordinal position of the changed item when it notifies of an item change.

## Syntax

Visual Basic (Declaration)	
<b>ReadOnly Property</b> SupportsItemOrdinals <b>As</b> System.Boolean	
C#	
System. <b>bool</b> SupportsItemOrdinals { <b>get</b> ;}	

### Property Value

**true** if the collection can provide item ordinal positions.

## Remarks

If this property returns **true**, LiveLinq can call the [EnableItemOrdinals](#) method to require providing ordinals.

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2


See Also

Reference

[IObservableSource<T> Interface](#)  
[IObservableSource<T> Members](#)

Events

>

Name	Description
 <a href="#">Changed</a>	Occurs after an item of the collection or the entire collection has changed.

[Top](#)

See Also

Reference

[IObservableSource<T> Interface](#)  
[C1.LiveLinq Namespace](#)

Changed Event

[C1.LiveLinq Namespace](#) > [IObservableSource<T> Interface](#) : Changed Event

Occurs after an item of the collection or the entire collection has changed.

Syntax

Visual Basic (Declaration)	
<code>Event Changed As System.EventHandler(Of SourceChangeEventArgs(Of T))</code>	
C#	
<code>event System.EventHandler&lt;SourceChangeEventArgs&lt;T&gt;&gt; Changed</code>	

Event Data

The event handler receives an argument of type [SourceChangeEventArgs<T>](#) containing data related to this event. The following **SourceChangeEventArgs<T>** properties provide information specific to this event.

Property	Description
<a href="#">ChangeType</a>	Gets the type of change.
<a href="#">Item</a>	Gets the object that is being changed.
<a href="#">Ordinal</a>	Gets the ordinal position of the collection item that is being changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[IObservableSource<T> Interface](#)

[IObservableSource<T> Members](#)

## ITransaction

[C1.LiveLinq Namespace](#) : ITransaction Interface

Represents a transaction with an explicit scope.

## Object Model

ITransaction

## Syntax

Visual Basic (Declaration)	
<b>Public Interface</b> ITransaction	
C#	
<b>public interface</b> ITransaction	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ITransaction Members](#)  
[C1.LiveLinq Namespace](#)

## Overview

[C1.LiveLinq Namespace](#) : ITransaction Interface

Represents a transaction with an explicit scope.

# Object Model

ITransaction

## Syntax

Visual Basic (Declaration)	
<b>Public Interface</b> ITransaction	
C#	
<b>public interface</b> ITransaction	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ITransaction Members](#)  
[C1.LiveLinq Namespace](#)



## Members

[Properties](#) [Methods](#)

[C1.LiveLinq Namespace](#) : ITransaction Interface




The following tables list the members exposed by [ITransaction](#).

## Public Properties

	Name	Description
	<a href="#">HasChanges</a>	Gets a value indicating whether any changes were made in the scope of this transaction.
	<a href="#">State</a>	Gets the current state of the transaction.

[Top](#)

## Public Methods

	Name	Description
	<a href="#">Commit</a>	Commits changes that were made while this transaction's scope was open.
	<a href="#">Rollback</a>	Rolls back changes that were made while this transaction's scope was open.
	<a href="#">Scope</a>	Opens a transaction scope.

[Top](#)

## See Also

### Reference

[ITransaction Interface](#)


[C1.LiveLinq Namespace](#)



## Methods

[C1.LiveLinq Namespace](#) : [ITransaction Interface](#)

For a list of all members of this type, see [ITransaction members](#).

## Public Methods

	Name	Description
	<a href="#">Commit</a>	Commits changes that were made while this transaction's scope was open.

	<a href="#">Rollback</a>	Rolls back changes that were made while this transaction's scope was open.
	<a href="#">Scope</a>	Opens a transaction scope.

[Top](#)

## See Also

### Reference

[ITransaction Interface](#)

[C1.LiveLinq Namespace](#)

### Commit Method

[C1.LiveLinq Namespace](#) > [ITransaction Interface](#) : Commit Method

Commits changes that were made while this transaction's scope was open.

## Syntax

Visual Basic (Declaration)	
<b>Sub</b> Commit()	
C#	
<b>void</b> Commit()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ITransaction Interface](#)

[ITransaction Members](#)

### Rollback Method

[C1.LiveLinq Namespace](#) > [ITransaction Interface](#) : Rollback Method

Rolls back changes that were made while this transaction's scope was open.

# Syntax

Visual Basic (Declaration)	
<code>Sub Rollback()</code>	
C#	
<code>void Rollback()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ITransaction Interface](#)

[ITransaction Members](#)

### Scope Method

[C1.LiveLinq Namespace](#) > [ITransaction Interface](#) : Scope Method

Opens a transaction scope.

# Syntax

Visual Basic (Declaration)	
<code>Function Scope() As System.IDisposable</code>	
C#	
<code>System.IDisposable Scope()</code>	

### Return Value

An instance of **System.IDisposable** that will close the scope when its **System.IDisposable.Dispose** method is called.

## Remarks

The transaction tracks changes only when they are made inside an open scope.

Calling **System.IDisposable.Dispose** on the return value closes the scope.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ITransaction Interface](#)



[ITransaction Members](#)

## Properties

[C1.LiveLinq Namespace](#) : [ITransaction Interface](#)

For a list of all members of this type, see [ITransaction members](#).

## Public Properties

	Name	Description
	<a href="#">HasChanges</a>	Gets a value indicating whether any changes were made in the scope of this transaction.
	<a href="#">State</a>	Gets the current state of the transaction.

[Top](#)

## See Also

### Reference

[ITransaction Interface](#)

[C1.LiveLinq Namespace](#)

## HasChanges Property

[C1.LiveLinq Namespace](#) > [ITransaction Interface](#) : HasChanges Property

Gets a value indicating whether any changes were made in the scope of this transaction.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

<b>ReadOnly Property</b> HasChanges <b>As</b> System.Boolean	
C#	
System.bool HasChanges {get;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ITransaction Interface](#)

[ITransaction Members](#)

### State Property

[C1.LiveLinq Namespace](#) > [ITransaction Interface](#) : State Property

Gets the current state of the transaction.

## Syntax

Visual Basic (Declaration)	
<b>ReadOnly Property</b> State <b>As</b> TransactionState	
C#	
TransactionState State {get;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ITransaction Interface](#)











[ITransaction Members](#)






# C1.LiveLinq.LiveViews Namespace

## Overview

### Classes

	Class	Description
	<a href="#">AggregationView&lt;TSource,TResult&gt;</a>	Represents a view having a single element calculated by aggregating a source view.
	<a href="#">GroupView&lt;TKey,TElement&gt;</a>	A group in a grouping view.
	<a href="#">OrderedView&lt;T&gt;</a>	Represents a sorted view.
	<a href="#">PropertyIsNotVirtualException</a>	Represents an exception that indicates that a property used in a result selector of a live view is not virtual.
	<a href="#">View</a>	Base class for the <a href="#">View&lt;T&gt;</a> class. Contains members that don't depend on the element type <i>T</i> .
	<a href="#">View&lt;T&gt;</a>	Represents a <i>live view</i> : a LINQ query result that supports two-way data binding and is kept up-to-date with base data.
	<a href="#">ViewRow</a>	Represents a view element (item) for the purposes of dynamic, programmatic access to its properties and data binding.
	<a href="#">ViewRowAddingEventArgs</a>	Provides data for the <a href="#">ViewRowCollection.ViewRowAdding</a> event.
	<a href="#">ViewRowCollection</a>	Represents a collection of <a href="#">ViewRow</a> objects used for programmatic access to view elements (items) and for data binding.
	<a href="#">ViewRowPropertyInfo</a>	Allows to control certain behavior of a property of the element type of a <a href="#">View</a> .

# Enumerations

	Enumeration	Description
	<a href="#">ViewMaintenanceMode</a>	Specifies how a view is synchronized with changes in its base data.
	<a href="#">ViewOrder</a>	Specifies whether and how a view must preserve item order if it exists in the source.
	<a href="#">ViewRowState</a>	The state of a view row with regard to edit, add and delete operations if they are performed directly on the view.

## See Also

### Reference

[C1.Silverlight.LiveLinq.5 Assembly](#)

## Classes

### AggregationView<TSource,TResult>

[C1.LiveLinq.LiveViews Namespace](#) : [AggregationView<TSource,TResult>](#) Class

The type of the elements of the source view.

The type of the single element of the aggregation view.

Represents a view having a single element calculated by aggregating a source view.

## Object Model

**AggregationView<TSource,TResult>**

## Syntax

Visual Basic (Declaration)	
<pre>Public Class AggregationView     (Of TSource,TResult)     Inherits View(Of TResult)     Implements C1.LiveLinq.IObservableSource(Of TResult)</pre>	
C#	

```
public class AggregationView<TSource,TResult> : View<TResult>,
C1.LiveLinq.IObservableSource<TResult>
```

## Type Parameters

*TSource*

The type of the elements of the source view.

*TResult*

The type of the single element of the aggregation view.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

**C1.LiveLinq.LiveViews.AggregationView<TSource,TResult>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AggregationView<TSource,TResult> Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

### Overview

[C1.LiveLinq.LiveViews Namespace](#) : [AggregationView<TSource,TResult> Class](#)

The type of the elements of the source view.

The type of the single element of the aggregation view.

Represents a view having a single element calculated by aggregating a source view.

## Object Model

AggregationView<TSource,TResult>

## Syntax

Visual Basic (Declaration)	
<pre>Public Class AggregationView     (Of TSource,TResult)     Inherits View(Of TResult)     Implements C1.LiveLinq.IObservableSource(Of TResult)</pre>	
C#	
<pre>public class AggregationView&lt;TSource,TResult&gt; : View&lt;TResult&gt;, C1.LiveLinq.IObservableSource&lt;TResult&gt;</pre>	

## Type Parameters

*TSource*

The type of the elements of the source view.

*TResult*

The type of the single element of the aggregation view.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

**C1.LiveLinq.LiveViews.AggregationView<TSource,TResult>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AggregationView<TSource,TResult> Members](#)

[C1.LiveLinq.LiveViews Namespace](#)










## Members


[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.LiveViews Namespace](#) : [AggregationView<TSource,TResult>](#) Class

The following tables list the members exposed by [AggregationView<TSource,TResult>](#).









## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )




	Value	Gets the value of the single element of the aggregation view.
---	-------	---

[Top](#)

## Public Methods


	Name	Description
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">Concat</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )

⇒	<a href="#">GroupBy</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">GroupJoin</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">Join</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">SelectMany</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
⇒	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

	<a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">Union</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )

[Top](#)

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )

[Top](#)

## See Also

### Reference



[AggregationView<TSource,TResult> Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Properties








[C1.LiveLinq.LiveViews Namespace](#) : [AggregationView<TSource,TResult> Class](#)

For a list of all members of this type, see [AggregationView<TSource,TResult> members](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )



	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TResult&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Value</a>	Gets the value of the single element of the aggregation view.

[Top](#)

## See Also

### Reference

[AggregationView<TSource,TResult> Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Value Property

[C1.LiveLinq.LiveViews Namespace](#) > [AggregationView<TSource,TResult> Class](#) : Value Property

Gets the value of the single element of the aggregation view.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Value As TResult</code>	
C#	
<code>public TResult Value {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[AggregationView<TSource,TResult> Class](#)

[AggregationView<TSource,TResult> Members](#)

## GroupView<TKey,TElement>

[C1.LiveLinq.LiveViews Namespace](#) : [GroupView<TKey,TElement> Class](#)

The type of the key used for grouping.

The type of the elements in the group view.

A group in a grouping view.

## Object Model

**GroupView<TKey,TElement>**

## Syntax

Visual Basic (Declaration)	
<code>&lt;System.Diagnostics.DebuggerTypeProxyAttribute(ProxyTypeName="C1.LiveLinq.Seq</code>	

```

uenceDebugView, C1.Silverlight.LiveLinq.5, Version=5.0.20151.455,
Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",
    Target=,
    TargetTypeName="")>
<System.Diagnostics.DebuggerDisplayAttribute(Value="\{ Count= {Count}, Key=
{Key} \}",
    Name="",
    Type="",
    Target=,
    TargetTypeName="")>
Public NotInheritable Class GroupView
    (Of TKey,TElement)
    Inherits View(Of TElement)
    Implements C1.LiveLinq.IObservableSource(Of TElement)

```

C#

```

[System.Diagnostics.DebuggerTypeProxy(ProxyTypeName="C1.LiveLinq.SequenceDebu
gView, C1.Silverlight.LiveLinq.5, Version=5.0.20151.455, Culture=neutral,
PublicKeyToken=2aa4ec5576d6c3ce",
    Target=,
    TargetTypeName="")]
[System.Diagnostics.DebuggerDisplay(Value="\{ Count= {Count}, Key= {Key} \}",
    Name="",
    Type="",
    Target=,
    TargetTypeName="")]
public sealed class GroupView<TKey,TElement> : View<TElement>,
C1.LiveLinq.IObservableSource<TElement>

```

## Type Parameters

*TKey*

The type of the key used for grouping.

*TElement*

The type of the elements in the group view.

## Remarks

A grouping view is a result of a **GroupBy** operation on a live view. It is a collection of groups. Each group contains elements with the same key. That collection is a live

view, and every group in itself is a live view, an object of type **GroupView<TKey, TElement>**.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

**C1.LiveLinq.LiveViews.GroupView<TKey, TElement>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey, TElement> Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

## Overview

[C1.LiveLinq.LiveViews Namespace](#) : GroupView<TKey, TElement> Class

The type of the key used for grouping.

The type of the elements in the group view.

A group in a grouping view.

## Object Model

**GroupView<TKey, TElement>**

## Syntax

Visual Basic (Declaration)

```
<System.Diagnostics.DebuggerTypeProxyAttribute(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.Silverlight.LiveLinq.5, Version=5.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",  
    Target=,  
    TargetTypeName="")>  
<System.Diagnostics.DebuggerDisplayAttribute(Value="\{ Count= {Count}, Key=
```

```

{Key} \}",
    Name="",
    Type="",
    Target=,
    TargetTypeName="")>
Public NotInheritable Class GroupView
    (Of TKey,TElement)
    Inherits View(Of TElement)
    Implements C1.LiveLinq.IObservableSource(Of TElement)

```

C#

```

[System.Diagnostics.DebuggerTypeProxy(ProxyTypeName="C1.LiveLinq.SequenceDebu
gView, C1.Silverlight.LiveLinq.5, Version=5.0.20151.455, Culture=neutral,
PublicKeyToken=2aa4ec5576d6c3ce",
    Target=,
    TargetTypeName="")]
[System.Diagnostics.DebuggerDisplay(Value="\{ Count= {Count}, Key= {Key} \}",
    Name="",
    Type="",
    Target=,
    TargetTypeName="")]
public sealed class GroupView<TKey,TElement> : View<TElement>,
C1.LiveLinq.IObservableSource<TElement>

```

## Type Parameters

*TKey*

The type of the key used for grouping.

*TElement*

The type of the elements in the group view.

## Remarks

A grouping view is a result of a **GroupBy** operation on a live view. It is a collection of groups. Each group contains elements with the same key. That collection is a live view, and every group in itself is a live view, an object of type **GroupView<TKey, TElement>**.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

**C1.LiveLinq.LiveViews.GroupView<TKey,TElement>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Members](#)

[C1.LiveLinq.LiveViews Namespace](#)







### Members






[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.LiveViews Namespace](#) : [GroupView<TKey,TElement>](#) Class

The following tables list the members exposed by [GroupView<TKey,TElement>](#).



### Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Overridden. This property overrides <a href="#">DeferredMaintenance</a> .
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
	<a href="#">Key</a>	Gets the key value of the group.

	<a href="#">MaintenanceMode</a>	Overridden. This property overrides <a href="#">MaintenanceMode</a> .
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Parent</a>	Gets the grouping view (the result of a <b>GroupBy</b> operation) to which this group belongs.
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## Public Methods

	Name	Description
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

≡	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">Concat</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">GroupBy</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">GroupJoin</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">Join</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
≡	<a href="#">Maintain</a>	Overridden. This method overrides <a href="#">View.Maintain</a> .
≡	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )



⇒💎	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒💎	<a href="#">PurgeEmptyGroups</a>	Overridden. This method overrides <a href="#">View.PurgeEmptyGroups</a> .
⇒💎	<a href="#">Rebuild</a>	Overridden. This method overrides <a href="#">View.Rebuild</a> .
⇒💎	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒💎	<a href="#">SelectMany</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒💎	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒💎	<a href="#">ToString</a>	Returns a string that represents this instance of <a href="#">GroupView&lt;TKey,TElement&gt;</a>
⇒💎	<a href="#">Union</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒💎	<a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )

[Top](#)

## Public Events

	Name	Description
⚡	<a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )

[Top](#)












## See Also













### Reference

GroupView<TKey,TElement> Class  
C1.LiveLinq.LiveViews Namespace

## Methods

>

Name	Description
 <a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
 <a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
 <a href="#">Concat</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
 <a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
 <a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
 <a href="#">GroupBy</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
 <a href="#">GroupJoin</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
 <a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from

	<a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒  <a href="#">Join</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒  <a href="#">Maintain</a>	Overridden. This method overrides <a href="#">View.Maintain</a> .
⇒  <a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒  <a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒  <a href="#">PurgeEmptyGroups</a>	Overridden. This method overrides <a href="#">View.PurgeEmptyGroups</a> .
⇒  <a href="#">Rebuild</a>	Overridden. This method overrides <a href="#">View.Rebuild</a> .
⇒  <a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒  <a href="#">SelectMany</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒  <a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒  <a href="#">ToString</a>	Returns a string that represents this instance of <a href="#">GroupView&lt;TKey,TElement&gt;</a>
⇒  <a href="#">Union</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
⇒  <a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )

[Top](#)

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

### Maintain Method

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : Maintain Method

This method overrides [View.Maintain](#).

## Syntax

Visual Basic (Declaration)	
<code>Public Overrides NotOverridable Sub Maintain()</code>	
C#	
<code>public override void Maintain()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)  
[GroupView<TKey,TElement> Members](#)

### PurgeEmptyGroups Method

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : PurgeEmptyGroups Method

This method overrides [View.PurgeEmptyGroups](#).

## Syntax

Visual Basic (Declaration)	
<code>Public Overrides NotOverridable Sub PurgeEmptyGroups()</code>	
C#	
<code>public override void PurgeEmptyGroups()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)  
[GroupView<TKey,TElement> Members](#)

Rebuild Method

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : Rebuild Method

This method overrides [View.Rebuild](#).

Syntax

Visual Basic (Declaration)	
<code>Public Overrides NotOverridable Sub Rebuild()</code>	
C#	
<code>public override void Rebuild()</code>	

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[GroupView<TKey,TElement> Class](#)  
[GroupView<TKey,TElement> Members](#)

ToString Method

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : ToString Method

Returns a string that represents this instance of [GroupView<TKey,TElement>](#)

Syntax

Visual Basic (Declaration)	
<code>Public Overrides Function ToString() As System.String</code>	
C#	
<code>public override System.string ToString()</code>	

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)









[GroupView<TKey,TElement> Members](#)




## Properties

[C1.LiveLinq.LiveViews Namespace](#) : [GroupView<TKey,TElement> Class](#)

For a list of all members of this type, see [GroupView<TKey,TElement> members](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Overridden. This property overrides <a href="#">DeferredMaintenance</a> .
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;TElement&gt;</a> )
	<a href="#">Key</a>	Gets the key value of the group.
	<a href="#">MaintenanceMode</a>	Overridden. This property overrides <a href="#">MaintenanceMode</a> .
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item

		was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Parent</a>	Gets the grouping view (the result of a <b>GroupBy</b> operation) to which this group belongs.
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

### DeferredMaintenance Property

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : DeferredMaintenance Property

This property overrides [DeferredMaintenance](#).

## Syntax

Visual Basic (Declaration)	
<b>Public Overrides NotOverridable ReadOnly Property</b> DeferredMaintenance <b>As</b> System.Boolean	
C#	
<b>public override</b> System. <b>bool</b> DeferredMaintenance { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)  
[GroupView<TKey,TElement> Members](#)

### Key Property

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : Key Property

Gets the key value of the group.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Key As TKey</code>	
C#	
<code>public TKey Key {get;}</code>	

## Remarks

The key value is common to the elements of this group.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)  
[GroupView<TKey,TElement> Members](#)

### MaintenanceMode Property

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : MaintenanceMode Property

This property overrides [MaintenanceMode](#).



## Syntax

Visual Basic (Declaration)	
<code>Public Overrides NotOverridable Property MaintenanceMode As ViewMaintenanceMode</code>	
C#	
<code>public override ViewMaintenanceMode MaintenanceMode {get; set;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[GroupView<TKey,TElement> Class](#)

[GroupView<TKey,TElement> Members](#)

### Parent Property

[C1.LiveLinq.LiveViews Namespace](#) > [GroupView<TKey,TElement> Class](#) : Parent Property

Gets the grouping view (the result of a **GroupBy** operation) to which this group belongs.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Parent As View</code>	
C#	
<code>public View Parent {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[GroupView<TKey,TElement> Class](#)

[GroupView<TKey,TElement> Members](#)

## OrderedView<T>

[C1.LiveLinq.LiveViews Namespace](#) : [OrderedView<T> Class](#)

The type of the elements in the view.

Represents a sorted view.

## Object Model

OrderedView<T>

## Syntax

Visual Basic (Declaration)

```
Public Class OrderedView(Of T)
    Inherits View(Of T)
    Implements C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class OrderedView<T> : View<T>, C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

The type of the elements in the view.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

[C1.LiveLinq.LiveViews.View<T>](#)

**C1.LiveLinq.LiveViews.OrderedView<T>**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[OrderedView<T> Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)  
[OrderBy<TKey>\(Expression<Func<T,TKey>>\) Method](#)  
[OrderByDescending<TKey> Method](#)

### Overview

[C1.LiveLinq.LiveViews Namespace](#) : [OrderedView<T> Class](#)

The type of the elements in the view.

Represents a sorted view.

## Object Model

**OrderedView<T>**

### Syntax

Visual Basic (Declaration)

```
Public Class OrderedView(Of T)
    Inherits View(Of T)
    Implements C1.LiveLinq.IObservableSource(Of T)
```

C#

```
public class OrderedView<T> : View<T>, C1.LiveLinq.IObservableSource<T>
```

### Type Parameters

*T*

The type of the elements in the view.

### Inheritance Hierarchy

System.Object  
[C1.LiveLinq.LiveViews.View](#)  
[C1.LiveLinq.LiveViews.View<T>](#)  
**C1.LiveLinq.LiveViews.OrderedView<T>**

### Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[OrderedView<T> Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)  
[OrderBy<TKey>\(Expression<Func<T,TKey>>\) Method](#)  
[OrderByDescending<TKey> Method](#)







## Members




[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.LiveViews Namespace](#) : [OrderedView<T>](#) Class

The following tables list the members exposed by [OrderedView<T>](#).






## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## Public Methods

	Name	Description
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">Concat</a>	Overloaded. Concatenation of two views. (Inherited from


		<a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">GroupJoin</a>	Overloaded. Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Join</a>	Overloaded. Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

⇒	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
⇒	<a href="#">ThenBy&lt;TKey&gt;</a>	Performs a subsequent ordering of view elements in ascending order according to a key.
⇒	<a href="#">ThenByDescending&lt;TKey&gt;</a>	Performs a subsequent ordering of view elements in descending order according to a key.
⇒	<a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Union</a>	Overloaded. Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
⇒	<a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## Public Events

	Name	Description
--	------	-------------

	Changed	Occurs after an item of the view or the entire view has changed. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
---	---------	---

[Top](#)

## See Also

### Reference

[OrderedView<T> Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

[OrderBy<TKey>\(Expression<Func<T,TKey>>\) Method](#)






[OrderByDescending<TKey> Method](#)

## Methods

[C1.LiveLinq.LiveViews Namespace](#) : [OrderedView<T> Class](#)










For a list of all members of this type, see [OrderedView<T> members](#).

## Public Methods

	Name	Description
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AttachAggregationView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">AttachView</a>	Overloaded. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
	<a href="#">Concat</a>	Overloaded. Concatenation of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )



≡	<a href="#">Contains</a>	Determines whether the view contains a specified item. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">GroupJoin</a>	Overloaded. Correlates the elements of two views based on equality of keys and groups the results. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">Join</a>	Overloaded. Correlates the elements of two views based on matching keys. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
≡	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

 <a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
 <a href="#">ThenBy&lt;TKey&gt;</a>	Performs a subsequent ordering of view elements in ascending order according to a key.
 <a href="#">ThenByDescending&lt;TKey&gt;</a>	Performs a subsequent ordering of view elements in descending order according to a key.
 <a href="#">ToString</a>	Returns a string representing this view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">Union</a>	Overloaded. Set union of two views. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )
 <a href="#">Where</a>	Filters the source view based on a predicate. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View&lt;T&gt;</a> )

[Top](#)

## See Also

### Reference

[OrderedView<T> Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

[OrderBy<TKey>\(Expression<Func<T,TKey>>\) Method](#)

[OrderByDescending<TKey> Method](#)

## ThenBy<TKey> Method

[C1.LiveLinq.LiveViews Namespace](#) > [OrderedView<T> Class](#) : ThenBy<TKey>(Expression<Func<T,TKey>>)  
Method

The type of the key returned by *keySelector*.

A function to extract a key from each element.

Performs a subsequent ordering of view elements in ascending order according to a key.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function ThenBy(Of TKey)( _     ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of T, TKey)) _     ) As OrderedView(Of T)</pre>	
C#	
<pre>public OrderedView&lt;T&gt; ThenBy&lt;TKey&gt;(     System.Linq.Expressions.Expression&lt;Func&lt;T,TKey&gt;&gt; keySelector )</pre>	

## Parameters

*keySelector*

A function to extract a key from each element.

## Type Parameters

*TKey*

The type of the key returned by *keySelector*.

## Return Value

A view whose elements are sorted according to a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[OrderedView<T> Class](#)

[OrderedView<T> Members](#)

## ThenByDescending<TKey> Method

[C1.LiveLinq.LiveViews Namespace](#) > [OrderedView<T> Class](#) : ThenByDescending<TKey> Method

The type of the key returned by *keySelector*.

A function to extract a key from each element.

Performs a subsequent ordering of view elements in descending order according to a key.

## Syntax

Visual Basic (Declaration)

```
Public Function ThenByDescending(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
    ) As OrderedView(Of T)
```

C#

```
public OrderedView<T> ThenByDescending<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector  
)
```

## Parameters

*keySelector*

A function to extract a key from each element.

## Type Parameters

*TKey*

The type of the key returned by *keySelector*.

## Return Value

A view whose elements are sorted in descending order according to a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[OrderedView<T> Class](#)

[OrderedView<T> Members](#)

## PropertyIsNotVirtualException

[C1.LiveLinq.LiveViews Namespace](#) : [PropertyIsNotVirtualException Class](#)

Represents an exception that indicates that a property used in a result selector of a live view is not virtual.

## Object Model

PropertyIsNotVirtualException

## Syntax

Visual Basic (Declaration)

```
Public Class PropertyIsNotVirtualException
    Inherits System.Exception
```

C#

```
public class PropertyIsNotVirtualException : System.Exception
```

## Inheritance Hierarchy

System.Object

System.Exception

**C1.LiveLinq.LiveViews.PropertyIsNotVirtualException**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyIsNotVirtualException Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

# Overview

[C1.LiveLinq.LiveViews Namespace](#) : PropertyIsNotVirtualException Class

Represents an exception that indicates that a property used in a result selector of a live view is not virtual.

## Object Model

PropertyIsNotVirtualException

## Syntax

Visual Basic (Declaration)	
<pre>Public Class PropertyIsNotVirtualException     Inherits System.Exception</pre>	
C#	
<pre>public class PropertyIsNotVirtualException : System.Exception</pre>	

## Inheritance Hierarchy

System.Object  
System.Exception  
**C1.LiveLinq.LiveViews.PropertyIsNotVirtualException**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyIsNotVirtualException Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)












## Members

[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.LiveViews Namespace](#) : PropertyIsNotVirtualException Class





The following tables list the members exposed by [PropertyIsNotVirtualException](#).

## Public Properties

	Name	Description
	<a href="#">Context</a>	Lambda expression where a type with the non-virtual property is used.
	<a href="#">Data</a>	(Inherited from System.Exception)
	<a href="#">HelpLink</a>	(Inherited from System.Exception)
	<a href="#">HResult</a>	(Inherited from System.Exception)
	<a href="#">InnerException</a>	(Inherited from System.Exception)
	<a href="#">Message</a>	Overridden. Gets a message that describes the current exception.
	<a href="#">Property</a>	The non-virtual property.
	<a href="#">ResultType</a>	The result type of the lambda expression.
	<a href="#">Source</a>	(Inherited from System.Exception)
	<a href="#">StackTrace</a>	(Inherited from System.Exception)
	<a href="#">TargetSite</a>	(Inherited from System.Exception)


[Top](#)

## Public Methods

	Name	Description
	<a href="#">GetBaseException</a>	(Inherited from System.Exception)
	<a href="#">GetObjectData</a>	(Inherited from System.Exception)
	<a href="#">GetType</a>	(Inherited from System.Exception)
	<a href="#">ToString</a>	(Inherited from System.Exception)

[Top](#)

## Protected Events

	Name	Description
	<a href="#">SerializeObjectState</a>	(Inherited from System.Exception)

[Top](#)












## See Also

### Reference

[PropertyIsNotVirtualException Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Properties

>

Name	Description
 <a href="#">Context</a>	Lambda expression where a type with the non-virtual property is used.
 <a href="#">Data</a>	(Inherited from System.Exception)
 <a href="#">HelpLink</a>	(Inherited from System.Exception)
 <a href="#">HResult</a>	(Inherited from System.Exception)
 <a href="#">InnerException</a>	(Inherited from System.Exception)
 <a href="#">Message</a>	Overridden. Gets a message that describes the current exception.
 <a href="#">Property</a>	The non-virtual property.
 <a href="#">ResultType</a>	The result type of the lambda expression.
 <a href="#">Source</a>	(Inherited from System.Exception)
 <a href="#">StackTrace</a>	(Inherited from System.Exception)
 <a href="#">TargetSite</a>	(Inherited from System.Exception)

[Top](#)

## See Also



## Reference

[PropertyIsNotVirtualException Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### Context Property

[C1.LiveLinq.LiveViews Namespace](#) > [PropertyIsNotVirtualException Class](#) : Context Property

Lambda expression where a type with the non-virtual property is used.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Context As System.Linq.Expressions.LambdaExpression</code>	
C#	
<code>public System.Linq.Expressions.LambdaExpression Context {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyIsNotVirtualException Class](#)

[PropertyIsNotVirtualException Members](#)

### Message Property

[C1.LiveLinq.LiveViews Namespace](#) > [PropertyIsNotVirtualException Class](#) : Message Property

Gets a message that describes the current exception.

## Syntax

Visual Basic (Declaration)	
<code>Public Overrides ReadOnly Property Message As System.String</code>	
C#	
<code>public override System.string Message {get;}</code>	

## Property Value

The error message that explains the reason for the exception.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyIsNotVirtualException Class](#)

[PropertyIsNotVirtualException Members](#)

### Property Property

[C1.LiveLinq.LiveViews Namespace](#) > [PropertyIsNotVirtualException Class](#) : Property Property

The non-virtual property.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property Property As</b> System.Reflection.PropertyInfo	
C#	
<b>public</b> System.Reflection.PropertyInfo Property { <b>get</b> ;}	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[PropertyIsNotVirtualException Class](#)

[PropertyIsNotVirtualException Members](#)

### ResultType Property

[C1.LiveLinq.LiveViews Namespace](#) > [PropertyIsNotVirtualException Class](#) : ResultType Property

The result type of the lambda expression.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property ResultType As System.Type</code>	
C#	
<code>public System.Type ResultType {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

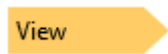
[PropertyIsNotVirtualException Class](#)  
[PropertyIsNotVirtualException Members](#)

## View

[C1.LiveLinq.LiveViews Namespace](#) : View Class

Base class for the [View<T>](#) class. Contains members that don't depend on the element type *T*.

## Object Model



## Syntax

Visual Basic (Declaration)	
<code>Public MustInherit Class View</code>	
C#	
<code>public abstract class View</code>	

## Remarks

Use this class to type variables that can accept views with different element types or a view with anonymous element type.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.View**

[C1.LiveLinq.LiveViews.View<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

## Overview

[C1.LiveLinq.LiveViews Namespace](#) : View Class

Base class for the [View<T>](#) class. Contains members that don't depend on the element type *T*.

## Object Model



## Syntax

Visual Basic (Declaration)	
<code>Public MustInherit Class View</code>	
C#	
<code>public abstract class View</code>	

## Remarks

Use this class to type variables that can accept views with different element types or a view with anonymous element type.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.View**

[C1.LiveLinq.LiveViews.View<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Members](#)

[C1.LiveLinq.LiveViews Namespace](#)







## Members




[Properties](#) [Methods](#)

[C1.LiveLinq.LiveViews Namespace](#) : View Class

The following tables list the members exposed by [View](#).







## Public Properties



	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view.
	<a href="#">CurrentItem</a>	Gets the current item in the view.
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode.
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable.
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data.
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item

		was not set by other means. The default is True.
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source.
	<a href="#">Rows</a>	Gets the collection of <a href="#">ViewRow</a> objects used for programmatic access to view elements (items) and for data binding.
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code.

[Top](#)

## Public Methods

	Name	Description
	<a href="#">AllowInResult</a>	Specifies that a type with non-virtual properties can be used in a result selector.
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> .
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access.
	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance.
	<a href="#">Maintain</a>	Brings the view up to date with its source data.
	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view.

	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query.
	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property.

[Top](#)

## See Also

### Reference

[View Class](#)








[C1.LiveLinq.LiveViews Namespace](#)



## Methods

[C1.LiveLinq.LiveViews Namespace](#) : [View Class](#)

For a list of all members of this type, see [View members](#).

## Public Methods

	Name	Description
 	<a href="#">AllowInResult</a>	Specifies that a type with non-virtual properties can be used in a result selector.
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> .
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access.
	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance.
	<a href="#">Maintain</a>	Brings the view up to date with its source data.
	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view.

	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query.
	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property.

[Top](#)

## See Also

### Reference

[View Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### AllowInResult Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : AllowInResult Method

The type that is allowed to be used.

Specifies that a type with non-virtual properties can be used in a result selector.

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared Sub AllowInResult( _     ByVal type As System.Type _ )</pre>	
C#	
<pre>public static void AllowInResult(     System.Type type )</pre>	

### Parameters

*type*

The type that is allowed to be used.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



## Reference

[View Class](#)

[View Members](#)

## AsCollectionViewFactory Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : AsCollectionViewFactory Method

Returns an instance of **System.ComponentModel.ICollectionViewFactory** that can be used as a source of a **System.Windows.Data.CollectionViewSource**.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function AsCollectionViewFactory() As System.ComponentModel.ICollectionViewFactory</pre>	
C#	
<pre>public System.ComponentModel.ICollectionViewFactory AsCollectionViewFactory()</pre>	

### Return Value

A factory that returns the same View as a **System.ComponentModel.ICollectionView**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

## AsDynamic Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : AsDynamic Method

Used for views with anonymous type constructor as the result selector, converts the [View](#) to a View<dynamic> so it can be used for data binding and programmatic access.

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> AsDynamic() <b>As</b> View(Of Object)	
C#	
<b>public</b> View<object> AsDynamic()	

## Return Value

A dynamic view.

## Remarks

A view with anonymous type constructor as the result selector cannot be used for data binding or programmatic access without applying [AsDynamic](#) to it. An attempt to do so results in an exception. After applying [AsDynamic](#), such view can be used for data binding and programmatic access without limitations.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

### DeferMaintenance Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : DeferMaintenance Method

Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance.

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> DeferMaintenance() <b>As</b> System.IDisposable	
C#	
<b>public</b> System.IDisposable DeferMaintenance()	

## Return Value

An **System.IDisposable** object that you can use to dispose of the calling object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

### Maintain Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : Maintain Method

Brings the view up to date with its source data.

## Syntax

Visual Basic (Declaration)	
<b>Public Overridable Sub</b> Maintain()	
C#	
<b>public virtual void</b> Maintain()	

## Remarks

If source data have not changed since the last time the view was maintained (updated), this method does nothing. It also does nothing if the view's [MaintenanceMode](#) is **Immediate**, because in that case the view is guaranteed to be in synch with its base data at all times. If the view is in deferred mode (its [MaintenanceMode](#) property returns **true**), the programmer can use the **Maintain** method to force updating the view.

Note that it is not necessary to call **Maintain** to make sure you get updated data from the view. The view is automatically updated every time you request data from it, if base data changed since the last request, regardless of the view's [MaintenanceMode](#).

LiveLinq maintains views using optimized incremental algorithms, not simply re-populates them from scratch. It calculating the delta in the view from the delta in the base data. In most cases, it allows to propagate the change from base data to the view very fast.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

### PurgeEmptyGroups Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : PurgeEmptyGroups Method

Remove empty groups from a grouping view.

## Syntax

Visual Basic (Declaration)	
<b>Public Overridable Sub</b> PurgeEmptyGroups()	
C#	
<b>public virtual void</b> PurgeEmptyGroups()	

## Remarks

This method is used only for **GroupBy** (grouping) views, does nothing for views of other kinds.

When a grouping view is populated, it does not contain empty groups. But later, as a result of maintaining the grouping view, keeping it in synch with changes in its base data, some of the groups can become empty. If it is undesirable to have empty groups, you can call this method to delete them.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)  
[View Members](#)

## Rebuild Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : Rebuild Method

Re-populates the view by re-executing the view's query.

## Syntax

Visual Basic (Declaration)	
<code>Public Overridable Sub Rebuild()</code>	
C#	
<code>public virtual void Rebuild()</code>	

## Remarks

This method is rarely needed, because normally automatic incremental [maintenance](#) is faster than re-executing the query over the entire base data collection. However, if for some reason you need to re-populate it from scratch, that can be done with the **Rebuild** method.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)  
[View Members](#)

## SetTransaction Method

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : SetTransaction Method

The new value for the the [Transaction](#) property.

Set this parameter to True to prevent exception if you have writable property paths and want to ignore them (but be aware that updates through property paths are not tracked by the transaction). The default is False.

Sets the value of the [Transaction](#) property.

## Syntax

Visual Basic (Declaration)

```
Public Sub SetTransaction( _  
    ByVal transaction As ITransaction, _  
    Optional ByVal allowPropertyPaths As System.Boolean _  
)
```

C#

```
public void SetTransaction(  
    ITransaction transaction,  
    System.bool allowPropertyPaths  
)
```

### Parameters

*transaction*

The new value for the the [Transaction](#) property.

*allowPropertyPaths*

Set this parameter to True to prevent exception if you have writable property paths and want to ignore them (but be aware that updates through property paths are not tracked by the transaction). The default is False.

## Remarks

If a writable property path exists in the view element type (for example, `Order.Customer.City`), then an exception is thrown unless suppressed by setting [allowPropertyPaths](#) to True, because modifying properties using such paths cannot be supported by transactions. To prevent the exception, set the [allowPropertyPaths](#) parameter to True, but make sure you do not use such property paths in your code and don't have a two-way data binding to such property path in your controls. If you modify a property using such path in your code or through data binding, that modification happens outside the transaction scope.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View Class](#)










[View Members](#)

## Properties

[C1.LiveLinq.LiveViews Namespace](#) : View Class

For a list of all members of this type, see [View members](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view.
	<a href="#">CurrentItem</a>	Gets the current item in the view.
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode.
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable.
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data.
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True.
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source.
	<a href="#">Rows</a>	Gets the collection of <a href="#">ViewRow</a> objects used for programmatic access to view elements (items) and for data binding.
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so

		the programmer does not need to do it manually in code.
--	--	---

[Top](#)

## See Also

### Reference

[View Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### Count Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : Count Property

Gets the total number of elements in the view.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Count As System.Integer</code>	
C#	
<code>public System.int Count {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

### CurrentItem Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : CurrentItem Property

Gets the current item in the view.

## Syntax



Visual Basic (Declaration)	
<code>Public ReadOnly Property CurrentItem As System.Object</code>	
C#	
<code>public System.object CurrentItem {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

### DeferredMaintenance Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : DeferredMaintenance Property

Gets the effective value of MaintenanceMode.

## Syntax

Visual Basic (Declaration)	
<code>Public Overridable ReadOnly Property DeferredMaintenance As System.Boolean</code>	
C#	
<code>public virtual System.bool DeferredMaintenance {get;}</code>	

### Property Value

**true** if [MaintenanceMode](#) is set to **Deferred**, or if it is set to **Default** and no listeners are registered with this view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View Class](#)

[View Members](#)

### IsReadOnly Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : IsReadOnly Property

Gets a value indicating whether this view is read-only, not updatable.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> IsReadOnly <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool IsReadOnly { <b>get</b> ;}	

## Remarks

Properties exposed by a view can be updatable or read-only. Updatable properties of a view can be modified directly in the view.

All properties of a read-only (not updatable) view are read-only. In an updatable view, properties directly corresponding to base data (source) properties are updatable, calculated properties are read-only. For example, in a view `from x in X select new { x.P, Q = x.Q + 1 }` P is updatable and Q is read-only.

Read-only properties of a view cannot be modified directly in the view, but they still reflect up-to-date values of the source, so the difference is often not critical, you can always modify corresponding property in the source, that will automatically change the property in the view.

Also, a read-only view cannot be cleared or its rows deleted or new rows added directly in the view (but all these actions can be performed on the source data collection and they will normally result in the corresponding changes of the view).

A **Join** view is read-only by default, but one of its two parts can be made updatable using the [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#) method.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View Class](#)

[View Members](#)

[AsUpdatable<T> Method](#)

[ViewRow Class](#)

[ViewRowState Enumeration](#)

## MaintenanceMode Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : MaintenanceMode Property

Gets or sets a value controlling how the view is synchronized with changes in its base data.

## Syntax

Visual Basic (Declaration)

```
Public Overridable Property MaintenanceMode As ViewMaintenanceMode
```

C#

```
public virtual ViewMaintenanceMode MaintenanceMode {get; set;}
```

## Remarks

A view in **Default** mode (which is the default value for this property) is effectively in **Immediate** mode if it has a listener (for example, if a GUI control is bound to it); otherwise it is in **Deferred** mode. To find out whether the view is effectively in **Deferred** or in **Immediate** mode, you can use the [DeferredMaintenance](#) property.

If you set this property to **Deferred**, no listeners are allowed to register with this view. An attempt to register a listener will result in an exception.

**See Also:** View Maintenance Mode.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

[DeferredMaintenance Property](#)

## MoveToFirstOnReset Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : MoveToFirstOnReset Property

Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any [C1.LiveLinq.SourceChangeType.Reset](#) notification) if current item was not set by other means. The default is True.

## Syntax

Visual Basic (Declaration)

```
Public Property MoveToFirstOnReset As System.Boolean
```

C#

```
public System.bool MoveToFirstOnReset {get; set;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)

[View Members](#)

## Order Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : Order Property

Gets a value indicating whether and how this view preserves item order if it exists in its base data source.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property Order As ViewOrder
```

C#

```
public ViewOrder Order {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[View Class](#)  
[View Members](#)

Rows Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : Rows Property

Gets the collection of [ViewRow](#) objects used for programmatic access to view elements (items) and for data binding.

Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Rows As ViewRowCollection</code>	
C#	
<code>public ViewRowCollection Rows {get;}</code>	

Remarks

There can be a view with elements of any type *T*, as represented by the generic class [View<T>](#). That type is not always known beforehand, so it is not always possible to access properties of view elements with strong-typed (early binding) code. Dynamic (untyped, late binding) access to view elements is provided by the [ViewRowCollection](#) owned by the view.

The collection of *view rows* ([ViewRow](#) objects) is always synchronized with the collection of view elements. [ViewRow](#) objects provide programmatic access to view elements and their properties.

Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

See Also

Reference

[View Class](#)  
[View Members](#)

## Transaction Property

[C1.LiveLinq.LiveViews Namespace](#) > [View Class](#) : Transaction Property

Gets an instance of [C1.LiveLinq.ITransaction](#) associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Transaction As ITransaction</code>	
C#	
<code>public ITransaction Transaction {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View Class](#)  
[View Members](#)

## View<T>

[C1.LiveLinq.LiveViews Namespace](#) : View<T> Class

The type of the elements in the view.

Represents a *live view*: a LINQ query result that supports two-way data binding and is kept up-to-date with base data.

## Object Model

View<T>

## Syntax

Visual Basic (Declaration)	
<pre> &lt;System.Diagnostics.DebuggerTypeProxyAttribute(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.Silverlight.LiveLinq.5, Version=5.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",     Target=,     TargetTypeName="")&gt; &lt;System.Diagnostics.DebuggerDisplayAttribute(Value="Count={Count}",     Name="",     Type="",     Target=,     TargetTypeName="")&gt; &lt;System.Reflection.DefaultMemberAttribute("Item")&gt; Public Class View(Of T)     Inherits View     Implements C1.LiveLinq.IObservableSource(Of T) </pre>	
C#	
<pre> [System.Diagnostics.DebuggerTypeProxy(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.Silverlight.LiveLinq.5, Version=5.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",     Target=,     TargetTypeName="")] [System.Diagnostics.DebuggerDisplay(Value="Count={Count}",     Name="",     Type="",     Target=,     TargetTypeName="")] [System.Reflection.DefaultMember("Item")] public class View&lt;T&gt; : View, C1.LiveLinq.IObservableSource&lt;T&gt; </pre>	

## Type Parameters

*T*

The type of the elements in the view.

## Inheritance Hierarchy

System.Object

C1.LiveLinq.LiveViews.View

**C1.LiveLinq.LiveViews.View<T>**

C1.Data.ClientView<T>

C1.LiveLinq.LiveViews.AggregationView<TSource,TResult>

[C1.LiveLinq.LiveViews.GroupView<TKey,TElement>](#)  
[C1.LiveLinq.LiveViews.OrderedView<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)

### Overview

[C1.LiveLinq.LiveViews Namespace](#) : View<T> Class

The type of the elements in the view.

Represents a *live view*: a LINQ query result that supports two-way data binding and is kept up-to-date with base data.

## Object Model

View<T>

## Syntax

Visual Basic (Declaration)

```
<System.Diagnostics.DebuggerTypeProxyAttribute(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.Silverlight.LiveLinq.5, Version=5.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",  
    Target=,  
    TargetTypeName="")>  
<System.Diagnostics.DebuggerDisplayAttribute(Value="Count={Count}",  
    Name="",  
    Type="",  
    Target=,  
    TargetTypeName="")>  
<System.Reflection.DefaultMemberAttribute("Item")>  
Public Class View(Of T)  
    Inherits View
```



Implements [C1.LiveLinq.IObservableSource\(Of T\)](#)

C#

```
[System.Diagnostics.DebuggerTypeProxy(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.Silverlight.LiveLinq.5, Version=5.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce", Target=, TargetTypeName="")]
[System.Diagnostics.DebuggerDisplay(Value="Count={Count}", Name="", Type="", Target=, TargetTypeName="")]
[System.Reflection.DefaultMember("Item")]
public class View<T> : View, C1.LiveLinq.IObservableSource<T>
```

## Type Parameters

*T*

The type of the elements in the view.

## Inheritance Hierarchy

System.Object

[C1.LiveLinq.LiveViews.View](#)

**C1.LiveLinq.LiveViews.View<T>**

[C1.Data.ClientView<T>](#)

[C1.LiveLinq.LiveViews.AggregationView<TSource,TResult>](#)

[C1.LiveLinq.LiveViews.GroupView<TKey,TElement>](#)

[C1.LiveLinq.LiveViews.OrderedView<T>](#)

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Members](#)

[C1.LiveLinq.LiveViews Namespace](#)










## Members

[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.LiveViews Namespace](#) : [View<T>](#) Class

The following tables list the members exposed by [View<T>](#).

### Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position.
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's


		scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
--	--	---

[Top](#)

## Public Methods


	Name	Description
≡	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">AttachAggregationView</a>	Overloaded.
≡	<a href="#">AttachView</a>	Overloaded.
≡	<a href="#">Concat</a>	Overloaded. Concatenation of two views.
≡	<a href="#">Contains</a>	Determines whether the view contains a specified item.
≡	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items.
≡	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function.

≡	<a href="#">GroupJoin</a>	Overloaded. Correlates the elements of two views based on equality of keys and groups the results.
≡	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence.
≡	<a href="#">Join</a>	Overloaded. Correlates the elements of two views based on matching keys.
≡	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order.
≡	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order.
≡	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form.
≡	<a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.
≡	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">ToString</a>	Returns a string representing this view.
≡	<a href="#">Union</a>	Overloaded. Set union of two views.

	<a href="#">Where</a>	Filters the source view based on a predicate.
---	-----------------------	---

[Top](#)

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed.

[Top](#)

## See Also

### Reference

[View<T> Class](#)




[C1.LiveLinq.LiveViews Namespace](#)

## Methods

[C1.LiveLinq.LiveViews Namespace](#) : [View<T> Class](#)

For a list of all members of this type, see [View<T> members](#).

## Public Methods

	Name	Description
	<a href="#">AsCollectionViewFactory</a>	Returns an instance of <b>System.ComponentModel.ICollectionViewFactory</b> that can be used as a source of a <b>System.Windows.Data.CollectionViewSource</b> . (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AsDynamic</a>	Used for views with anonymous type constructor as the result selector, converts the <a href="#">View</a> to a View<dynamic> so it can be used for data binding and programmatic access. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">AttachAggregationView</a>	Overloaded.

≡	<a href="#">AttachView</a>	Overloaded.
≡	<a href="#">Concat</a>	Overloaded. Concatenation of two views.
≡	<a href="#">Contains</a>	Determines whether the view contains a specified item.
≡	<a href="#">DeferMaintenance</a>	Enters a defer cycle that you can use to make bulk changes to the view sources and delay automatic view maintenance. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the view items.
≡	<a href="#">GroupBy</a>	Overloaded. Groups the elements of a view according to a specified key selector function.
≡	<a href="#">GroupJoin</a>	Overloaded. Correlates the elements of two views based on equality of keys and groups the results.
≡	<a href="#">IndexOf</a>	Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence.
≡	<a href="#">Join</a>	Overloaded. Correlates the elements of two views based on matching keys.
≡	<a href="#">Maintain</a>	Brings the view up to date with its source data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">OrderBy&lt;TKey&gt;</a>	Sorts the elements of a view in ascending order.
≡	<a href="#">OrderByDescending&lt;TKey&gt;</a>	Sorts the elements of a view in descending order.
≡	<a href="#">PurgeEmptyGroups</a>	Remove empty groups from a grouping view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">Rebuild</a>	Re-populates the view by re-executing the view's query. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

≡	<a href="#">Select&lt;TResult&gt;</a>	Projects each element of a view into a new form.
≡	<a href="#">SelectMany</a>	Overloaded. Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.
≡	<a href="#">SetTransaction</a>	Sets the value of the <a href="#">Transaction</a> property. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
≡	<a href="#">ToString</a>	Returns a string representing this view.
≡	<a href="#">Union</a>	Overloaded. Set union of two views.
≡	<a href="#">Where</a>	Filters the source view based on a predicate.

[Top](#)

## See Also

### Reference

[View<T> Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### AttachAggregationView Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : AttachAggregationView Method

## Overload List

Overload	Description
<a href="#">AttachAggregationView&lt;TResult&gt;(Object,Func&lt;View,AggregationView&lt;T,TResult&gt;&gt;)</a>	
<a href="#">AttachAggregationView&lt;TResult&gt;(Func&lt;View,AggregationView&lt;T,TResult&gt;&gt;)</a>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

[AttachAggregationView<TResult>\(Object,Func<View,AggregationView<T,TResult>>\) Method](#)

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [AttachAggregationView Method](#) :

[AttachAggregationView<TResult>\(Object,Func<View,AggregationView<T,TResult>>\) Method](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function AttachAggregationView(Of TResult)( _     ByVal subqueryId As System.Object, _     ByVal selector As System.Func(Of View(Of T),AggregationView(Of T,TResult)) _ ) As AggregationView(Of T,TResult)</pre>	
C#	
<pre>public AggregationView&lt;T,TResult&gt; AttachAggregationView&lt;TResult&gt;(     System.object subqueryId,     System.Func&lt;View&lt;T&gt;,AggregationView&lt;T,TResult&gt;&gt; selector )</pre>	

### Parameters

*subqueryId*

*selector*

### Type Parameters

*TResult*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2



# See Also

## Reference

- [View<T> Class](#)
- [View<T> Members](#)
- [Overload List](#)

[AttachAggregationView<TResult>\(Func<View,AggregationView<T,TResult>>\) Method](#)  
[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [AttachAggregationView Method](#) :  
[AttachAggregationView<TResult>\(Func<View,AggregationView<T,TResult>>\) Method](#)

# Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function AttachAggregationView(Of TResult)( _     ByVal selector As System.Func(Of View(Of T),AggregationView(Of T,TResult)) - ) As AggregationView(Of T,TResult)</pre>	
C#	
<pre>public AggregationView&lt;T,TResult&gt; AttachAggregationView&lt;TResult&gt;(     System.Func&lt;View&lt;T&gt;,AggregationView&lt;T,TResult&gt;&gt; selector )</pre>	

## Parameters

*selector*

## Type Parameters

*TResult*

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [View<T> Class](#)
- [View<T> Members](#)
- [Overload List](#)

# AttachView Method

C1.LiveLinq.LiveViews Namespace > View<T> Class : AttachView Method

## Overload List

Overload	Description
<a href="#">AttachView&lt;TResult&gt;(Func&lt;View,View&lt;TResult&gt;&gt;)</a>	
<a href="#">AttachView&lt;TResult&gt;(Object,Func&lt;View,View&lt;TResult&gt;&gt;)</a>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)

[AttachView<TResult>\(Func<View,View<TResult>>\) Method](#)  
[C1.LiveLinq.LiveViews Namespace > View<T> Class > AttachView Method](#) :  
[AttachView<TResult>\(Func<View,View<TResult>>\) Method](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function AttachView(Of TResult)( _     ByVal selector As System.Func(Of View(Of T),View(Of TResult)) _ ) As View(Of TResult)</pre>	
C#	
<pre>public View&lt;TResult&gt; AttachView&lt;TResult&gt;(      System.Func&lt;View&lt;T&gt;,View&lt;TResult&gt;&gt; selector )</pre>	

### Parameters

*selector*

## Type Parameters

*TResult*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)  
[Overload List](#)

[AttachView<TResult>\(Object,Func<View,View<TResult>>\) Method](#)  
[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [AttachView Method](#) :  
[AttachView<TResult>\(Object,Func<View,View<TResult>>\) Method](#)

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function AttachView(Of TResult)( _     ByVal subqueryId As System.Object, _     ByVal selector As System.Func(Of View(Of T),View(Of TResult)) _ ) As View(Of TResult)</pre>	
C#	
<pre>public View&lt;TResult&gt; AttachView&lt;TResult&gt;(      System.object subqueryId,      System.Func&lt;View&lt;T&gt;,View&lt;TResult&gt;&gt; selector  )</pre>	

### Parameters

*subqueryId*  
  
*selector*

### Type Parameters

*TResult*

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)  
[Overload List](#)

### Concat Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Concat Method

Concatenation of two views.

## Overload List

Overload	Description
<a href="#">Concat(IObservableSource&lt;T&gt;)</a>	Concatenation of two views.
<a href="#">Concat(ObservableCollection&lt;T&gt;)</a>	このビューとコレクションを結合します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)

### Concat(IObservableSource<T>) Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [Concat Method](#) : Concat(IObservableSource<T>) Method

A collection (usually, a view) to concatenate to this view's collection of elements.

Concatenation of two views.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Concat( _  
    ByVal second As IObservableSource(Of T) _  
) As View(Of T)
```

C#

```
public View<T> Concat(  
    IObservableSource<T> second  
)
```

### Parameters

*second*

A collection (usually, a view) to concatenate to this view's collection of elements.

### Return Value

The view that contains first the elements of this view and then the elements of the parameter collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

Concat(ObservableCollection<T>) Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [Concat Method](#) : Concat(ObservableCollection<T>) Method

このビューの要素のコレクションに結合するコレクション。

このビューとコレクションを結合します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Concat( _     ByVal second As System.Collections.ObjectModel.ObservableCollection(Of T)     _ ) As View(Of T)</pre>	
C#	
<pre>public View&lt;T&gt; Concat(     System.Collections.ObjectModel.ObservableCollection&lt;T&gt; second )</pre>	

### Parameters

*second*

このビューの要素のコレクションに結合するコレクション。

### Return Value

最初にこのビューの要素を含み、次にパラメータコレクションの要素を含むビュー。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)  
[Overload List](#)

### Contains Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Contains Method

The item to locate in the view.

Determines whether the view contains a specified item.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

```
Public Function Contains( _
    ByVal item As T _
) As System.Boolean
```

C#

```
public System.bool Contains(
    T item
)
```

## Parameters

*item*

The item to locate in the view.

## Return Value

**true** if the view contains the specified item; otherwise, **false**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

## GetEnumerator Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : GetEnumerator Method

Returns an enumerator that iterates through the view items.

## Syntax

Visual Basic (Declaration)

```
Public Function GetEnumerator() As System.Collections.Generic.IEnumerator(Of
T)
```

C#

```
public System.Collections.Generic.IEnumerator<T> GetEnumerator()
```

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)

### GroupBy Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : GroupBy Method

Groups the elements of a view according to a specified key selector function.

## Overload List

Overload	Descri ption
<a href="#">GroupBy&lt;TKey&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;)</a>	Group s the eleme nts of a view accor ding to a specifi ed key select or functi on.
<a href="#">GroupBy&lt;TKey,TElement&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Expression&lt;Func&lt;T,TElement&gt;&gt; )</a>	Group s the



	<p>elements of a view according to a specified key selector or function and projects the elements for each group by using a specified function.</p>
<p><code>GroupBy&lt;TKey,TElement,TResult&gt;(Expression&lt;Func&lt;T,TKey&gt;&gt;,Expression&lt;Func&lt;T,TElement&gt;&gt;,Expression&lt;Func&lt;TKey,IEnumerable&lt;TElement&gt;,TResult&gt;&gt;)</code></p>	<p>Groups the elements of a view according to a</p>

	<p>specified  key  select  or  function  and  creates a  result  value  from  each  group  and  its  key.  The  elements of  each  group  are  projected by  using  a  specified  function.</p>
--	---

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

[GroupBy<TKey>\(Expression<Func<T,TKey>>\) Method](#)

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [GroupBy Method](#) :

[GroupBy<TKey>\(Expression<Func<T,TKey>>\) Method](#)

The type of the key returned by *keySelector*.

A function to extract the key for each element.

Groups the elements of a view according to a specified key selector function.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function GroupBy(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
) As View(Of GroupView(Of TKey, T))
```

C#

```
public View<GroupView<TKey, T>> GroupBy<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector  
)
```

### Parameters

*keySelector*

A function to extract the key for each element.

### Type Parameters

*TKey*

The type of the key returned by *keySelector*.

### Return Value

A view containing elements of type [GroupView<TKey, TElement>](#) each containing a key value and a view of the elements having that key value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

`GroupBy<TKey,TElement>(Expression<Func<T,TKey>>,Expression<Func<T,TElement>>)`  
Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [GroupBy Method](#) :

`GroupBy<TKey,TElement>(Expression<Func<T,TKey>>,Expression<Func<T,TElement>>)` Method

The type of the key returned by *keySelector*.

The type of the element to which elements of each group are projected.

A function to extract the key for each element.

A function to map each source element to a *TElement*.

Groups the elements of a view according to a specified key selector function and projects the elements for each group by using a specified function.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function GroupBy
    (Of TKey,TElement)( _
        ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of
T,TKey)), _
        ByVal elementSelector As System.Linq.Expressions.Expression(Of Func(Of
T,TElement)) _
    ) As View(Of GroupView(Of TKey,TElement))
```

C#

```
public View<GroupView<TKey,TElement>> GroupBy<TKey,TElement>(
    System.Linq.Expressions.Expression<Func<T,TKey>> keySelector,
    System.Linq.Expressions.Expression<Func<T,TElement>> elementSelector
```

```
)
```

## Parameters

*keySelector*

A function to extract the key for each element.

*elementSelector*

A function to map each source element to a *TElement*.

## Type Parameters

*TKey*

The type of the key returned by *keySelector*.

*TElement*

The type of the element to which elements of each group are projected.

## Return Value

A view containing elements of type [GroupView<TKey,TElement>](#) each containing a key value and a view of the elements projected (mapped) from the elements having that key value.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

`GroupBy<TKey,TElement,TResult>(Expression<Func<T,TKey>>,Expression<Func<T,TElement>>,Expression<Func<TKey,IEnumerable<TElement>,TResult>>) Method`

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [GroupBy Method](#) :

`GroupBy<TKey,TElement,TResult>(Expression<Func<T,TKey>>,Expression<Func<T,TElement>>,Expression<Func<TKey,IEnumerable<TElement>,TResult>>) Method`

The type of the key returned by *keySelector*.

The type of the elements in groups.

The type of the result value returned by *resultSelector*.

A function to extract the key for each element.

A function to map each source element to an element in the **System.Linq.IGrouping`2**.

A function to create a result value from each group.

Groups the elements of a view according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function GroupBy
    (Of TKey,TElement,TResult)( _
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of
T,TKey)), _
    ByVal elementSelector As System.Linq.Expressions.Expression(Of Func(Of
T,TElement)), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
TKey,IEnumerable(Of TElement),TResult)) _
) As View(Of TResult)
```

C#

```
public View<TResult> GroupBy<TKey,TElement,TResult>(
    System.Linq.Expressions.Expression<Func<T,TKey>> keySelector,
    System.Linq.Expressions.Expression<Func<T,TElement>> elementSelector,

System.Linq.Expressions.Expression<Func<TKey,IEnumerable<TElement>,TResult>>
resultSelector
)
```

### Parameters

*keySelector*

A function to extract the key for each element.

*elementSelector*

A function to map each source element to an element in the **System.Linq.IGrouping`2**.

*resultSelector*

A function to create a result value from each group.

## Type Parameters

*TKey*

The type of the key returned by *keySelector*.

*TElement*

The type of the elements in groups.

*TResult*

The type of the result value returned by *resultSelector*.

## Return Value

A view of elements of type *TResult* where each element represents a projection over a group and its key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)  
[View<T> Members](#)  
[Overload List](#)

## GroupJoin Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : GroupJoin Method

Correlates the elements of two views based on equality of keys and groups the results.

## Overload List

Overload	Description
<a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;(IObservableSource&lt;TInner&gt;,Expression&lt;Func&lt;T,TKey&gt;&gt;,Expression&lt;Func&lt;TInner,TKey&gt;</a>	Correlates the elements of two views based on equality of keys

<a href="#">&gt;,Expression&lt;Func&lt;T,GroupView&lt;TKey,TInner&gt;,TResult&gt;&gt;)</a>	and groups the results.
<a href="#">GroupJoin&lt;TInner,TKey,TResult&gt;(ObservableCollection&lt;TInner&gt;,Expression&lt;Func&lt;T,TKey&gt;&gt;,Expression&lt;Func&lt;TInner,TKey&gt;&gt;,Expression&lt;Func&lt;T,GroupView&lt;TKey,TInner&gt;,TResult&gt;&gt;)</a>	キーの一致に基づいてこのビューの要素とコレクションを関連付け、結果をグループ化します。 。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

[GroupJoin<TInner,TKey,TResult>\(IObservableSource<TInner>,Expression<Func<T,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<T,GroupView<TKey,TInner>,TResult>>\)](#) Method  
[C1.LiveInq.LiveViews Namespace](#) > [View<T> Class](#) > [GroupJoin Method](#) :

[GroupJoin<TInner,TKey,TResult>\(IObservableSource<TInner>,Expression<Func<T,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<T,GroupView<TKey,TInner>,TResult>>\)](#) Method

The type of the elements of the view to join with this view.

The type of the keys returned by the key selector functions.

The type of the result elements.

The collection (usually, a view) to join to this view.

A function to extract the join key from each element of this view.

A function to extract the join key from each element of the second view.

A function to create a result view from an element from this view and a collection of matching elements from the second view.

Correlates the elements of two views based on equality of keys and groups the results.

## Syntax

Visual Basic (Declaration)



```
Public Overloads Function GroupJoin
    (Of TInner,TKey,TResult)( _
    ByVal inner As IObservableSource(Of TInner), _
    ByVal outerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
T,TKey)), _
    ByVal innerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
TInner,TKey)), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
T,GroupView(Of TKey,TInner),TResult)) _
) As View(Of TResult)
```

C#

```
public View<TResult> GroupJoin<TInner,TKey,TResult>(
    IObservableSource<TInner> inner,
    System.Linq.Expressions.Expression<Func<T,TKey>> outerKeySelector,
    System.Linq.Expressions.Expression<Func<TInner,TKey>> innerKeySelector,
    System.Linq.Expressions.Expression<Func<T,GroupView<TKey,TInner>,TResult>>
resultSelector
)
```

## Parameters

*inner*

The collection (usually, a view) to join to this view.

*outerKeySelector*

A function to extract the join key from each element of this view.

*innerKeySelector*

A function to extract the join key from each element of the second view.

*resultSelector*

A function to create a result view from an element from this view and a collection of matching elements from the second view.

## Type Parameters

*TInner*

The type of the elements of the view to join with this view.

*TKey*

The type of the keys returned by the key selector functions.

*TResult*

The type of the result elements.

## Return Value

A view containing elements of type *TResult* that are obtained by performing a grouped join on two views.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

`GroupJoin<TInner,TKey,TResult>(ObservableCollection<TInner>,Expression<Func<T,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<T,GroupView<TKey,TInner>,TResult>>)`  
Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [GroupJoin Method](#) :

`GroupJoin<TInner,TKey,TResult>(ObservableCollection<TInner>,Expression<Func<T,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<T,GroupView<TKey,TInner>,TResult>>)` Method

このビューと結合するコレクションの要素のタイプ。

キーセレクタ関数から返されるキーのタイプ。

結果要素のタイプ。

このビューと結合するコレクション。

このビューの各要素から結合キーを抽出する関数。

コレクションの各要素から結合キーを抽出する関数。

このビューの要素と、第2のビューの一致する要素のコレクションから結果ビューを作成する関数。

キーの一致に基づいてこのビューの要素とコレクションを関連付け、結果をグループ化します。

## Syntax

## Visual Basic (Declaration)

```
Public Overloads Function GroupJoin
    (Of TInner,TKey,TResult)( _
    ByVal inner As System.Collections.ObjectModel.ObservableCollection(Of
TInner), _
    ByVal outerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
T,TKey)), _
    ByVal innerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
TInner,TKey)), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
T,GroupView(Of TKey,TInner),TResult)) _
) As View(Of TResult)
```

## C#

```
public View<TResult> GroupJoin<TInner,TKey,TResult>(
    System.Collections.ObjectModel.ObservableCollection<TInner> inner,
    System.Linq.Expressions.Expression<Func<T,TKey>> outerKeySelector,
    System.Linq.Expressions.Expression<Func<TInner,TKey>> innerKeySelector,
    System.Linq.Expressions.Expression<Func<T,GroupView<TKey,TInner>,TResult>>
resultSelector
)
```

## Parameters

*inner*

このビューと結合するコレクション。

*outerKeySelector*

このビューの各要素から結合キーを抽出する関数。

*innerKeySelector*

コレクションの各要素から結合キーを抽出する関数。

*resultSelector*

このビューの要素と、第2のビューの一致する要素のコレクションから  
結果ビューを作成する関数。

## Type Parameters

*TInner*

このビューと結合するコレクションの要素のタイプ。

*TKey*

キーセレクタ関数から返されるキーのタイプ。

*TResult*

結果要素のタイプ。

## Return Value

このビューとコレクションに対してグループ化された結合を実行することで 取得されるタイプ *TResult* の要素を含むビュー。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

## IndexOf Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : IndexOf Method

The object to locate in the view.

Searches for the specified object among the elements of the view and returns the zero-based ordinal position of its first occurrence.

## Syntax

Visual Basic (Declaration)

```
Public Function IndexOf( _  
    ByVal item As T _  
) As System.Integer
```

C#

```
public System.int IndexOf(
```

```
    T item
)
```

### Parameters

*item*

The object to locate in the view.

### Return Value

The zero-based ordinal position of the first occurrence of *item* in the view, if found; otherwise, -1.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)

### Join Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Join Method

Correlates the elements of two views based on matching keys.

## Overload List

Overload	Description
<a href="#">Join&lt;TInner,TKey,TResult&gt;(IObservableSource&lt;TInner&gt;,Expression&lt;Func&lt;T,TKey&gt;&gt;,Expression&lt;Func&lt;TInner,TKey&gt;&gt;,Expression&lt;Func&lt;T,TInner,TResult&gt;&gt;)</a>	Correlates the elements of two views based on matching keys.
<a href="#">Join&lt;TInner,TKey,TResult&gt;(ObservableCollection&lt;TInner&gt;,Expression&lt;Func&lt;T,TKey&gt;&gt;,Expression&lt;Func&lt;TInner,TKey&gt;&gt;,Expression&lt;Func&lt;T,TInner,TResult&gt;&gt;)</a>	一致キーに基づいて、このビューの要素とコレクションを関連付けます。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

[Join<TInner,TKey,TResult>\(IObservableSource<TInner>,Expression<Func<T,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<T,TInner,TResult>>\)](#) Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [Join Method](#) :

[Join<TInner,TKey,TResult>\(IObservableSource<TInner>,Expression<Func<T,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<T,TInner,TResult>>\)](#) Method

The type of the elements of the view to join with this view.

The type of the keys returned by the key selector functions.

The type of the result elements.

The collection (usually, a view) to join to this view.

A function to extract the join key from each element of this view.

A function to extract the join key from each element of the second view.

A function to create a result element from two matching elements.

Correlates the elements of two views based on matching keys.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Join
    (Of TInner,TKey,TResult)( _
    ByVal inner As IObservableSource(Of TInner), _
    ByVal outerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
T,TKey)), _
    ByVal innerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
TInner,TKey)), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
T,TInner,TResult)) _
) As View(Of TResult)
```

C#

```
public View<TResult> Join<TInner,TKey,TResult>(
    IObservableSource<TInner> inner,
    System.Linq.Expressions.Expression<Func<T,TKey>> outerKeySelector,
    System.Linq.Expressions.Expression<Func<TInner,TKey>> innerKeySelector,
    System.Linq.Expressions.Expression<Func<T,TInner,TResult>> resultSelector
)
```

## Parameters

*inner*

The collection (usually, a view) to join to this view.

*outerKeySelector*

A function to extract the join key from each element of this view.

*innerKeySelector*

A function to extract the join key from each element of the second view.

*resultSelector*

A function to create a result element from two matching elements.

## Type Parameters

*TInner*

The type of the elements of the view to join with this view.

*TKey*

The type of the keys returned by the key selector functions.

*TResult*

The type of the result elements.

## Return Value

A view containing elements of type *TResult* that are obtained by performing an inner join on two views.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core

not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

[Join<TInner,TKey,TResult>\(ObservableCollection<TInner>,Expression<Func<T,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<T,TInner,TResult>>\)](#) Method

[C1.LiveLinq.LiveViews Namespace > View<T> Class > Join Method :](#)

[Join<TInner,TKey,TResult>\(ObservableCollection<TInner>,Expression<Func<T,TKey>>,Expression<Func<TInner,TKey>>,Expression<Func<T,TInner,TResult>>\)](#) Method

このビューと結合するコレクションの要素のタイプ。

キーセレクタ関数から返されるキーのタイプ。

結果要素のタイプ。

このビューと結合するコレクション。

このビューの各要素から結合キーを抽出する関数。

コレクションの各要素から結合キーを抽出する関数。

2つの一致する要素から結果要素を作成する関数。

一致キーに基づいて、このビューの要素とコレクションを関連付けます。

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Join
    (Of TInner,TKey,TResult)( _
        ByVal inner As System.Collections.ObjectModel.ObservableCollection(Of
TInner), _
        ByVal outerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
T,TKey)), _
        ByVal innerKeySelector As System.Linq.Expressions.Expression(Of Func(Of
TInner,TKey)), _
        ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
T,TInner,TResult)) _
    ) As View(Of TResult)
```



C#

```
public View<TResult> Join<TInner,TKey,TResult>(
    System.Collections.ObjectModel.ObservableCollection<TInner> inner,
    System.Linq.Expressions.Expression<Func<T,TKey>> outerKeySelector,
    System.Linq.Expressions.Expression<Func<TInner,TKey>> innerKeySelector,
    System.Linq.Expressions.Expression<Func<T,TInner,TResult>> resultSelector
)
```

## Parameters

*inner*

このビューと結合するコレクション。

*outerKeySelector*

このビューの各要素から結合キーを抽出する関数。

*innerKeySelector*

コレクションの各要素から結合キーを抽出する関数。

*resultSelector*

2つの一致する要素から結果要素を作成する関数。

## Type Parameters

*TInner*

このビューと結合するコレクションの要素のタイプ。

*TKey*

キーセレクタ関数から返されるキーのタイプ。

*TResult*

結果要素のタイプ。

## Return Value

このビューとコレクションで内部結合を実行することで取得されるタイプ *TResult* の要素を含むビュー。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core

not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)  
[Overload List](#)

### OrderBy<TKey> Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : OrderBy<TKey>(Expression<Func<T,TKey>>)  
Method

The type of the key returned by *keySelector*.

A function to extract a key from an element.

Sorts the elements of a view in ascending order.

## Syntax

Visual Basic (Declaration)

```
Public Function OrderBy(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
) As OrderedView(Of T)
```

C#

```
public OrderedView<T> OrderBy<TKey>(  
    System.Linq.Expressions.Expression<Func<T,TKey>> keySelector  
)
```

### Parameters

*keySelector*

A function to extract a key from an element.

### Type Parameters

*TKey*

The type of the key returned by *keySelector*.

### Return Value

A view whose elements are sorted according to a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

### OrderByDescending<TKey> Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : OrderByDescending<TKey> Method

The type of the key returned by *keySelector*.

A function to extract a key from an element.

Sorts the elements of a view in descending order.

## Syntax

Visual Basic (Declaration)

```
Public Function OrderByDescending(Of TKey)( _  
    ByVal keySelector As System.Linq.Expressions.Expression(Of Func(Of  
T, TKey)) _  
    ) As OrderedView(Of T)
```

C#

```
public OrderedView<T> OrderByDescending<TKey>(  
    System.Linq.Expressions.Expression<Func<T, TKey>> keySelector  
)
```

### Parameters

*keySelector*

A function to extract a key from an element.

### Type Parameters

*TKey*

The type of the key returned by *keySelector*.

## Return Value

A view whose elements are sorted in descending order according to a key.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

## Select<TResult> Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Select<TResult> Method

The type of the value returned by *selector*.

A transform function to apply to each element.

Projects each element of a view into a new form.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function Select(Of TResult)( _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of T, TResult))) _ ) As View(Of TResult)</pre>	
C#	
<pre>public View&lt;TResult&gt; Select&lt;TResult&gt;(     System.Linq.Expressions.Expression&lt;Func&lt;T, TResult&gt;&gt; selector )</pre>	

## Parameters

*selector*

A transform function to apply to each element.

## Type Parameters

*TResult*

The type of the value returned by *selector*.

## Return Value

A view whose elements are the result of invoking the transform function on each element of this view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

### SelectMany Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : SelectMany Method

Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.

## Overload List

Overload	Description
<a href="#">SelectMany&lt;TCollection,TResult&gt;(Expression&lt;Func&lt;T,IEnumerableSource&lt;TCollection&gt;&gt;&gt;,Expression&lt;Func&lt;T,TCollection,TResult&gt;&gt;)</a>	Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.
<a href="#">SelectMany&lt;TResult&gt;(Expression&lt;Func&lt;T,IEnumerableSource&lt;TResult&gt;&gt;&gt;)</a>	Projects each element of this view to a collection of <i>TResult</i> and flattens the resulting collections into one view.
<a href="#">SelectMany&lt;TCollection,TResult&gt;(Expression&lt;Fu</a>	このビューの各要素をコレクションのコレ

<a href="#">nc&lt;T,ObservableCollection&lt;TCollection&gt;&gt;&gt;,Expression&lt;Func&lt;T,TCollection,TResult&gt;&gt;&gt;)</a>	クシヨンに投影し、結果のコレクションを1つのコレクションにフラット化し、コレクション内の要素ごとに結果セクタ関数を呼び出します。
<a href="#">SelectMany&lt;TResult&gt;(Expression&lt;Func&lt;T,ObservableCollection&lt;TResult&gt;&gt;&gt;)</a>	このビューの各要素を <i>TResult</i> のコレクションに投影し、結果のコレクションを1つのビューにフラット化します。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)

[SelectMany<TCollection,TResult>\(Expression<Func<T,IObservableSource<TCollection>>>,Expression<Func<T,TCollection,TResult>>>\) Method](#)  
[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [SelectMany Method](#) :  
[SelectMany<TCollection,TResult>\(Expression<Func<T,IObservableSource<TCollection>>>,Expression<Func<T,TCollection,TResult>>>\) Method](#)

The type of the intermediate elements collected by *collectionSelector*.

The type of the elements of the resulting view.

A transform function to apply to each element of this view.

A transform function to apply to each element of the intermediate collection.

Projects each element of this view to a collection of collections, flattens the resulting collections into one collection, and invokes a result selector function on each element therein.

## Syntax

Visual Basic (Declaration)	
<b>Public Overloads Function</b> <a href="#">SelectMany</a> ( <b>Of</b> <a href="#">TCollection</a> , <a href="#">TResult</a> )( <a href="#">_</a>	

```

    ByVal collectionSelector As System.Linq.Expressions.Expression(Of Func(Of
T,IEnumerable(Of TCollection))), _
    ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of
T,TCollection,TResult)) _
) As View(Of TResult)

```

C#

```

public View<TResult> SelectMany<TCollection,TResult>(
    System.Linq.Expressions.Expression<Func<T,IEnumerable<TCollection>>>
collectionSelector,
    System.Linq.Expressions.Expression<Func<T,TCollection,TResult>>
resultSelector
)

```

## Parameters

*collectionSelector*

A transform function to apply to each element of this view.

*resultSelector*

A transform function to apply to each element of the intermediate collection.

## Type Parameters

*TCollection*

The type of the intermediate elements collected by *collectionSelector*.

*TResult*

The type of the elements of the resulting view.

## Return Value

A view whose elements are the result of invoking the one-to-many transform function *collectionSelector* on each element of this view and then mapping each of those collection elements and their corresponding source element to a result element.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

- [View<T> Class](#)
- [View<T> Members](#)
- [Overload List](#)

SelectMany<TResult>(Expression<Func<T,IEnumerableSource<TResult>>>) Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [SelectMany Method](#) :

SelectMany<TResult>(Expression<Func<T,IEnumerableSource<TResult>>>) Method

The type of the elements of the resulting view.

A transform function to apply to each element.

Projects each element of this view to a collection of *TResult* and flattens the resulting collections into one view.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function SelectMany(Of TResult)( _     ByVal selector As System.Linq.Expressions.Expression(Of Func(Of T,IEnumerableSource(Of TResult))) _ ) As View(Of TResult)</pre>	
C#	
<pre>public View&lt;TResult&gt; SelectMany&lt;TResult&gt;(      System.Linq.Expressions.Expression&lt;Func&lt;T,IEnumerableSource&lt;TResult&gt;&gt;&gt; selector )</pre>	

## Parameters

*selector*

A transform function to apply to each element.

## Type Parameters

*TResult*

The type of the elements of the resulting view.

## Return Value



A view whose elements are the result of invoking the one-to-many transform function on each element of this view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)  
[Overload List](#)

SelectMany<TCollection,TResult>(Expression<Func<T,ObservableCollection<TCollection>>>,Expression<Func<T,TCollection,TResult>>) Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [SelectMany Method](#) :

SelectMany<TCollection,TResult>(Expression<Func<T,ObservableCollection<TCollection>>>,Expression<Func<T,TCollection,TResult>>) Method

*collectionSelector* によって収集される中間要素のタイプ。

生成されたビュー内の要素の型。

このビューの各要素に適用する変換関数。

中間コレクションの各要素に適用する変換関数。

このビューの各要素をコレクションのコレクションに投影し、結果のコレクションを1つのコレクションにフラット化し、コレクション内の要素ごとに結果セクタ関数を呼び出します。

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function SelectMany     (Of TCollection,TResult)( _     ByVal collectionSelector As System.Linq.Expressions.Expression(Of Func(Of T,ObservableCollection(Of TCollection))), _     ByVal resultSelector As System.Linq.Expressions.Expression(Of Func(Of T,TCollection,TResult)) _ ) As View(Of TResult)</pre>	
C#	

```
public View<TResult> SelectMany<TCollection,TResult>(
    System.Linq.Expressions.Expression<Func<T,ObservableCollection<TCollection>>>
    collectionSelector,
    System.Linq.Expressions.Expression<Func<T,TCollection,TResult>>
    resultSelector
)
```

## Parameters

*collectionSelector*

このビューの各要素に適用する変換関数。

*resultSelector*

中間コレクションの各要素に適用する変換関数。

## Type Parameters

*TCollection*

*collectionSelector* によって収集される中間要素のタイプ。

*TResult*

生成されたビュー内の要素の型。

## Return Value

このビューの要素ごとに 1 対多の変換関数 *collectionSelector* を呼び出し、これらのコレクション要素ごとに、要素およびそれに対応するソース要素を結果要素に マッピングした結果を要素として含むビュー。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)

[View<T> Members](#)

[Overload List](#)

SelectMany<TResult>(Expression<Func<T,ObservableCollection<TResult>>>) Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [SelectMany Method](#) :

SelectMany<TResult>(Expression<Func<T,ObservableCollection<TResult>>>) Method

生成されたビュー内の要素の型。

各要素に適用する変換関数。

このビューの各要素を *TResult* のコレクションに投影し、結果のコレクションを 1 つのビューにフラット化します。

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function SelectMany(Of TResult)( _  
    ByVal selector As System.Linq.Expressions.Expression(Of Func(Of  
T,ObservableCollection(Of TResult))) _  
) As View(Of TResult)
```

C#

```
public View<TResult> SelectMany<TResult>(  
    System.Linq.Expressions.Expression<Func<T,ObservableCollection<TResult>>>  
selector  
)
```

## Parameters

*selector*

各要素に適用する変換関数。

## Type Parameters

*TResult*

生成されたビュー内の要素の型。

## Return Value

このビューの要素ごとに 1 対多の変換関数を呼び出した結果を要素として含むビュー。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)  
[Overload List](#)

### ToString Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : ToString Method

Returns a string representing this view.

## Syntax

Visual Basic (Declaration)	
<b>Public Overrides Function</b> ToString() <b>As</b> System.String	
C#	
<b>public override</b> System.string ToString()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)

### Union Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Union Method

Set union of two views.

## Overload List

Overload	Description
<a href="#">Union(IObservableSource&lt;T&gt;)</a>	Set union of two views.

<a href="#">Union(ObservableCollection&lt;T&gt;)</a>	このビューとコレクションの和を設定します。
--	-----------------------

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

[Union\(IObservableSource<T>\) Method](#)

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [Union Method](#) : Union(IObservableSource<T>) Method

A collection (usually, a view) whose distinct elements form the second set for the union.

Set union of two views.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Function Union( _     ByVal second As IObservableSource(Of T) _ ) As View(Of T)</pre>	
C#	
<pre>public View&lt;T&gt; Union(     IObservableSource&lt;T&gt; second )</pre>	

### Parameters

*second*

A collection (usually, a view) whose distinct elements form the second set for the union.

### Return Value

The view that contains the elements from both input views, excluding duplicates.

## Remarks

This method excludes duplicates from the result set. This is different behavior to the [Concat](#) method, which returns all the elements in the input sequences including duplicates.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)  
[Overload List](#)

Union(ObservableCollection<T>) Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) > [Union Method](#) : Union(ObservableCollection<T>) Method

設定する和の第 2 の集合となる要素を含むコレクション。

このビューとコレクションの和を設定します。

## Syntax

Visual Basic (Declaration)

```
Public Overloads Function Union( _  
    ByVal second As System.Collections.ObjectModel.ObservableCollection(Of T)  
-  
) As View(Of T)
```

C#

```
public View<T> Union(  
    System.Collections.ObjectModel.ObservableCollection<T> second  
)
```

### Parameters

*second*

設定する和の第 2 の集合となる要素を含むコレクション。

### Return Value

このビューとコレクションの両方から重複を除いた要素を含むビュー。

## Remarks

このメソッドは、結果セットから重複を除きます。これは、重複を含めて入力シーケンス内のすべての要素を返す [Concat\(ObservableCollection<T>\)](#) メソッドとは動作が異なります。

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)  
[View<T> Members](#)  
[Overload List](#)

### Where Method

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Where Method

A function to test each element for a condition.

Filters the source view based on a predicate.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function Where( _     ByVal predicate As System.Linq.Expressions.Expression(Of Func(Of T, Boolean))) _     ) As View(Of T)</pre>	
C#	
<pre>public View&lt;T&gt; Where(     System.Linq.Expressions.Expression&lt;Func&lt;T, bool&gt;&gt; predicate )</pre>	

### Parameters

*predicate*

A function to test each element for a condition.

## Return Value

A view that contains elements of this view that satisfy the condition.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[View<T> Class](#)







[View<T> Members](#)

## Properties




[C1.LiveLinq.LiveViews Namespace](#) : [View<T> Class](#)

For a list of all members of this type, see [View<T> members](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	Gets the total number of elements in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">CurrentItem</a>	Gets the current item in the view. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">DeferredMaintenance</a>	Gets the effective value of MaintenanceMode. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">IsReadOnly</a>	Gets a value indicating whether this view is read-only, not updatable. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Item</a>	Gets the view item (element) at the specified ordinal position.
	<a href="#">MaintenanceMode</a>	Gets or sets a value controlling how the view is synchronized with



		changes in its base data. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">MoveToFirstOnReset</a>	Gets or sets a value indicating that the first item must be made current after initial loading or reset (on any <a href="#">C1.LiveLinq.SourceChangeType.Reset</a> notification) if current item was not set by other means. The default is True. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Order</a>	Gets a value indicating whether and how this view preserves item order if it exists in its base data source. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )
	<a href="#">Transaction</a>	Gets an instance of <a href="#">C1.LiveLinq.ITransaction</a> associated with the view. If a view has a transaction associated with it, that transaction's scope is opened automatically every time the view is updated, so the programmer does not need to do it manually in code. (Inherited from <a href="#">C1.LiveLinq.LiveViews.View</a> )

[Top](#)

## See Also

### Reference

[View<T> Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### Item Property

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Item Property

The zero-based ordinal position of the view item.

Gets the view item (element) at the specified ordinal position.

## Syntax

Visual Basic (Declaration)	
<pre>Public ReadOnly Default Property Item( _     ByVal ordinal As System.Integer _ ) As T</pre>	

C#

public T this[  
 System.int ordinal  
]; {get;}

### Parameters

*ordinal*

The zero-based ordinal position of the view item.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)


[View<T> Members](#)

## Events

[C1.LiveLinq.LiveViews Namespace](#) : [View<T> Class](#)

For a list of all members of this type, see [View<T> members](#).

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after an item of the view or the entire view has changed.

[Top](#)

## See Also

### Reference

[View<T> Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

## Changed Event

[C1.LiveLinq.LiveViews Namespace](#) > [View<T> Class](#) : Changed Event

Occurs after an item of the view or the entire view has changed.

## Syntax

Visual Basic (Declaration)	
<code>Public Event Changed As System.EventHandler(Of SourceChangeEventArgs(Of T))</code>	
C#	
<code>public event System.EventHandler&lt;SourceChangeEventArgs&lt;T&gt;&gt; Changed</code>	

## Event Data

The event handler receives an argument of type [SourceChangeEventArgs<T>](#) containing data related to this event. The following **SourceChangeEventArgs<T>** properties provide information specific to this event.

Property	Description
<a href="#">ChangeType</a>	Gets the type of change.
<a href="#">Item</a>	Gets the object that is being changed.
<a href="#">Ordinal</a>	Gets the ordinal position of the collection item that is being changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[View<T> Class](#)

[View<T> Members](#)

## ViewRow

[C1.LiveLinq.LiveViews Namespace](#) : ViewRow Class

Represents a view element (item) for the purposes of dynamic, programmatic access to its properties and data binding.

## Object Model

ViewRow

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Reflection.DefaultMemberAttribute("Item")&gt; Public MustInherit Class ViewRow</pre>	
C#	
<pre>[System.Reflection.DefaultMember("Item")] public abstract class ViewRow</pre>	

## Inheritance Hierarchy

System.Object  
    **C1.LiveLinq.LiveViews.ViewRow**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)  
[Rows Property](#)

### Overview

[C1.LiveLinq.LiveViews Namespace](#) : ViewRow Class

Represents a view element (item) for the purposes of dynamic, programmatic access to its properties and data binding.

## Object Model

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>  
Public MustInherit Class ViewRow
```

C#

```
[System.Reflection.DefaultMember("Item")]  
public abstract class ViewRow
```

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.ViewRow**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

[Rows Property](#)


## Members





[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.LiveViews Namespace](#) : ViewRow Class

The following tables list the members exposed by [ViewRow](#).






## Public Properties

	Name	Description
	<a href="#">Item</a>	Overloaded. Gets or sets a value of the specified view property.

	<a href="#">RowState</a>	Gets the state of a view row with regard to edit, add and delete operations if they are performed directly on the view.
	<a href="#">Tag</a>	Gets or sets user-supplied data associated with the view item.
	<a href="#">Value</a>	Gets the view element (item) represented by this <a href="#">ViewRow</a> object.
	<a href="#">View</a>	Gets the view to which the <a href="#">ViewRow</a> belongs.


[Top](#)

## Public Methods

	Name	Description
	<a href="#">BeginEdit</a>	Puts the <a href="#">ViewRow</a> into edit mode.
	<a href="#">CancelEdit</a>	Cancels the edit occurring on the row.
	<a href="#">Delete</a>	Deletes a view item.
	<a href="#">EndEdit</a>	Ends the edit occurring on the row.
	<a href="#">ToString</a>	Gets the string representing this view row.


[Top](#)

## Protected Methods

	Name	Description
	<a href="#">OnPropertyChanged</a>	<a href="#">ViewRow.PropertyChanged</a> イベントを発生させます。

[Top](#)

## Public Events

	Name	Description
	<a href="#">PropertyChanged</a>	Occurs when a property value changes, after it has been changed.

[Top](#)

## See Also

### Reference

[ViewRow Class](#)

[C1.LiveLinq.LiveViews Namespace](#)






[Rows Property](#)

### Methods

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRow Class](#)


For a list of all members of this type, see [ViewRow members](#).

## Public Methods

	Name	Description
	<a href="#">BeginEdit</a>	Puts the <a href="#">ViewRow</a> into edit mode.
	<a href="#">CancelEdit</a>	Cancels the edit occurring on the row.
	<a href="#">Delete</a>	Deletes a view item.
	<a href="#">EndEdit</a>	Ends the edit occurring on the row.
	<a href="#">ToString</a>	Gets the string representing this view row.

[Top](#)

## Protected Methods

	Name	Description
	<a href="#">OnPropertyChanged</a>	<a href="#">ViewRow.PropertyChanged</a> イベントを発生させます。

[Top](#)

## See Also

### Reference

[ViewRow Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

[Rows Property](#)

## BeginEdit Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : [BeginEdit\(\)](#) Method

Puts the [ViewRow](#) into edit mode.

## Syntax

Visual Basic (Declaration)	
<pre>Public Sub BeginEdit()</pre>	
C#	
<pre>public void BeginEdit()</pre>	

## Remarks

In edit mode, events and notifications are temporarily suspended, letting the user make changes to more than one property without triggering validation rules.

編集モードは、.NET

データ連結メカニズムの標準機能です。これは、**System.ComponentModel.IEditableObject** インタフェースを

実装するすべてのデータソースによってサポートされます。詳細については、.NET Framework 文書の **System.Data.DataRowView** などを参照してください。

While a view item is in edit mode, changes made to its updatable properties directly in the view are not propagated to the corresponding base data properties until the edit operation is completed by a call to [EndEdit](#) (see [View.IsReadOnly](#) about updatability of view element properties directly in the view).

If you change base data (source) properties, those changes are propagated to this view and other views depending on that base data according to the normal view maintenance process, regardless of whether the view row is in edit mode or not.

Many-to-one relations between view properties are maintained automatically on changes made to updatable properties directly in the view, regardless of whether the view row is in edit mode or not. For example, if you set a CustomerID directly in the view, CustomerName will change accordingly, even if you do it in edit mode.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



## Reference

[ViewRow Class](#)

[ViewRow Members](#)

### CancelEdit Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : CancelEdit Method

Cancels the edit occurring on the row.

## Syntax

Visual Basic (Declaration)	
<code>Public Overridable Sub CancelEdit()</code>	
C#	
<code>public virtual void CancelEdit()</code>	

## Remarks

If the user set some of the updatable properties of the row while it was in edit mode, the changes to those properties are rolled back and not propagated to the corresponding base data properties.

See the [BeginEdit](#) method for more information.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

### Delete Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : Delete Method

Deletes a view item.

## Syntax

Visual Basic (Declaration)	
<b>Public Overridable Sub</b> Delete()	
C#	
<b>public virtual void</b> Delete()	

## Remarks

Deleting a view item is an update operation on the view. As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)), and it results in updating (in this case, deleting an item from) one of the view's base data collections.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

### EndEdit Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : EndEdit Method

Ends the edit occurring on the row.

## Syntax

Visual Basic (Declaration)	
<b>Public Overridable Sub</b> EndEdit()	
C#	
<b>public virtual void</b> EndEdit()	

## Remarks

If the user set some of the updatable properties of the row while it was in edit mode, the changes to those properties are propagated to the corresponding base data properties at this point.

See the [BeginEdit](#) method for more information.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

### OnPropertyChanged Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : OnPropertyChanged Method

The name of the property that is changed.

[ViewRow.PropertyChanged](#) イベントを発生させます。

## Syntax

Visual Basic (Declaration)	
<pre>Protected Overridable Sub OnPropertyChanged( _     ByVal propertyName As System.String _ )</pre>	
C#	
<pre>protected virtual void OnPropertyChanged(     System.string propertyName )</pre>	

### Parameters

*propertyName*

The name of the property that is changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)  
[ViewRow Members](#)

### ToString Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : ToString Method

Gets the string representing this view row.

## Syntax

Visual Basic (Declaration)	
<b>Public Overrides Function</b> ToString() <b>As</b> System.String	
C#	
<b>public override</b> System.string ToString()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference






[ViewRow Class](#)  
[ViewRow Members](#)

### Properties

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRow Class](#)

For a list of all members of this type, see [ViewRow members](#).

## Public Properties

	Name	Description
	<a href="#">Item</a>	Overloaded. Gets or sets a value of the specified view property.
	<a href="#">RowState</a>	Gets the state of a view row with regard to edit, add and delete operations if they are performed directly on the view.
	<a href="#">Tag</a>	Gets or sets user-supplied data associated with the view item.
	<a href="#">Value</a>	Gets the view element (item) represented by this <a href="#">ViewRow</a> object.
	<a href="#">View</a>	Gets the view to which the <a href="#">ViewRow</a> belongs.

[Top](#)

## See Also

### Reference

[ViewRow Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

[Rows Property](#)

### Item Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : Item Property

Gets or sets a value of the specified view property.

## Overload List

Overload	Description
<a href="#">Item(Int32)</a>	Gets or sets a value of the specified view property.
<a href="#">Item(String)</a>	Gets or sets a value of the specified view property.

## Remarks

Setting a view property is an update operation on the view. As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)), and it results in updating one of the view's base data collections.

Only updatable properties can be set. An attempt to set a read-only property results in an exception. See [View.IsReadOnly](#) for more details.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

Item(Int32) Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) > [Item Property](#) : Item(Int32) Property

The ordinal position of the property in the collection of public properties of the type of the view element.

Gets or sets a value of the specified view property.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads Property Item( _     ByVal propertyOrdinal As System.Integer _ ) As System.Object</pre>	
C#	
<pre>public System.object Item(     System.int propertyOrdinal ) {get; set;}</pre>	

### Parameters

*propertyOrdinal*

The ordinal position of the property in the collection of public properties of the type of the view element.

### Property Value

The value of the property.

## Remarks

Setting a view property is an update operation on the view. As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)), and it results in updating one of the view's base data collections.

Only updatable properties can be set. An attempt to set a read-only property results in an exception. See [View.IsReadOnly](#) for more details.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)  
[ViewRow Members](#)  
[Overload List](#)

### Item(String) Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) > [Item Property](#) : Item(String) Property

The name of the property.

Gets or sets a value of the specified view property.

## Syntax

Visual Basic (Declaration)

```
Public Overloads Property Item( _  
    ByVal propertyName As System.String _  
) As System.Object
```

C#

```
public System.object Item(  
    System.string propertyName  
) {get; set;}
```

### Parameters

*propertyName*

The name of the property.

## Property Value

The value of the property.

## Remarks

Setting a view property is an update operation on the view. As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)), and it results in updating one of the view's base data collections.

Only updatable properties can be set. An attempt to set a read-only property results in an exception. See [View.IsReadOnly](#) for more details.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

[Overload List](#)

### RowState Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : RowState Property

Gets the state of a view row with regard to edit, add and delete operations if they are performed directly on the view.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> RowState <b>As</b> ViewRowState	
C#	



```
public ViewRowState RowState {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

### Tag Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : Tag Property

Gets or sets user-supplied data associated with the view item.

## Syntax

Visual Basic (Declaration)	
<pre>Public Property Tag As System.Object</pre>	
C#	
<pre>public System.object Tag {get; set;}</pre>	

## Remarks

Use this property to store any object you want to associate in your code with the view item that you need to access quickly.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

## Value Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : Value Property

Gets the view element (item) represented by this [ViewRow](#) object.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Value As System.Object</code>	
C#	
<code>public System.object Value {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)

[ViewRow Members](#)

## View Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : View Property

Gets the view to which the [ViewRow](#) belongs.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property View As View</code>	
C#	
<code>public View View {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference


[ViewRow Class](#)  
[ViewRow Members](#)

## Events

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRow Class](#)

For a list of all members of this type, see [ViewRow members](#).

# Public Events

	Name	Description
	<a href="#">PropertyChanged</a>	Occurs when a property value changes, after it has been changed.

[Top](#)

# See Also

## Reference

[ViewRow Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)  
[Rows Property](#)

## PropertyChanged Event

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRow Class](#) : PropertyChanged Event

Occurs when a property value changes, after it has been changed.

# Syntax

Visual Basic (Declaration)	
<b>Public Event</b> <a href="#">PropertyChanged</a> <b>As</b> System.ComponentModel.PropertyChangedEventHandler	
C#	
<b>public event</b> System.ComponentModel.PropertyChangedEventHandler <a href="#">PropertyChanged</a>	

# Event Data

The event handler receives an argument of type `System.ComponentModel.PropertyChangedEventArgs` containing data related to this event. The following **PropertyChangedEventArgs** properties provide information specific to this event.

Property	Description
<b>PropertyName</b>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRow Class](#)  
[ViewRow Members](#)

## ViewRowAddingEventArgs

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowAddingEventArgs Class

Provides data for the [ViewRowCollection.ViewRowAdding](#) event.

## Object Model

ViewRowAddingEventArgs

## Syntax

Visual Basic (Declaration)	
<pre>Public Class ViewRowAddingEventArgs     Inherits System.EventArgs</pre>	
C#	
<pre>public class ViewRowAddingEventArgs : System.EventArgs</pre>	

## Inheritance Hierarchy

System.Object  
System.EventArgs  
**C1.LiveLinq.LiveViews.ViewRowAddingEventArgs**

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ViewRowAddingEventArgs Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Overview

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRowAddingEventArgs Class](#)

Provides data for the [ViewRowCollection.ViewRowAdding](#) event.

# Object Model

[ViewRowAddingEventArgs](#)

# Syntax

Visual Basic (Declaration)	
<pre>Public Class ViewRowAddingEventArgs     Inherits System.EventArgs</pre>	
C#	
<pre>public class ViewRowAddingEventArgs : System.EventArgs</pre>	

# Inheritance Hierarchy

System.Object  
    System.EventArgs  
        **C1.LiveLinq.LiveViews.ViewRowAddingEventArgs**

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

Reference

[ViewRowAddingEventArgs Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)


Members

[Properties](#)

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRowAddingEventArgs](#) Class

The following tables list the members exposed by [ViewRowAddingEventArgs](#).

Public Properties

	Name	Description
	<a href="#">Row</a>	Gets the new view row that has just been added to <a href="#">ViewRowCollection</a> .

[Top](#)

See Also

Reference


[ViewRowAddingEventArgs Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

Properties

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRowAddingEventArgs](#) Class

For a list of all members of this type, see [ViewRowAddingEventArgs members](#).

Public Properties

	Name	Description
	<a href="#">Row</a>	Gets the new view row that has just been added to <a href="#">ViewRowCollection</a> .

[Top](#)

See Also

Reference

[ViewRowAddingEventArgs Class](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Row Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowAddingEventArgs Class](#) : Row Property

Gets the new view row that has just been added to [ViewRowCollection](#).

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Row As ViewRow</code>	
C#	
<code>public ViewRow Row {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowAddingEventArgs Class](#)

[ViewRowAddingEventArgs Members](#)

## ViewRowCollection

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowCollection Class

Represents a collection of [ViewRow](#) objects used for programmatic access to view elements (items) and for data binding.

## Object Model

**ViewRowCollection**

## Syntax

Visual Basic (Declaration)	
<code>&lt;System.Diagnostics.DebuggerDisplayAttribute(Value="Count = {Count}", Name="", Type="")</code>	

```

    Target=,
    TargetTypeName="")>
<System.Diagnostics.DebuggerTypeProxyAttribute(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.Silverlight.LiveLinq.5, Version=5.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",
    Target=,
    TargetTypeName="")>
<System.Reflection.DefaultMemberAttribute("Item")>
Public MustInherit Class ViewRowCollection
    Implements C1.LiveLinq.IObservableSource(Of ViewRow)

```

C#

```

[System.Diagnostics.DebuggerDisplay(Value="Count = {Count}",
    Name="",
    Type="",
    Target=,
    TargetTypeName="")]
[System.Diagnostics.DebuggerTypeProxy(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.Silverlight.LiveLinq.5, Version=5.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",
    Target=,
    TargetTypeName="")]
[System.Reflection.DefaultMember("Item")]
public abstract class ViewRowCollection :
    C1.LiveLinq.IObservableSource<ViewRow>

```

## Remarks

A **ViewRowCollection** is owned by a view, see [View.Rows](#).

The collection of *view rows* ([ViewRow](#) objects) is always synchronized with the collection of view elements.

[ViewRow](#) objects provide programmatic access to view elements and their properties. Also, the [View.Rows](#) collection serves as the data source for data binding, when you bind a control or another client to a view.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.ViewRowCollection**

## Requirements



**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

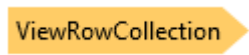
[ViewRowCollection Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Overview

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowCollection Class

Represents a collection of [ViewRow](#) objects used for programmatic access to view elements (items) and for data binding.

## Object Model



## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Diagnostics.DebuggerDisplayAttribute(Value="Count = {Count}",     Name="",     Type="",     Target=,     TargetTypeName="")&gt; &lt;System.Diagnostics.DebuggerTypeProxyAttribute(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.Silverlight.LiveLinq.5, Version=5.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",     Target=,     TargetTypeName="")&gt; &lt;System.Reflection.DefaultMemberAttribute("Item")&gt; Public MustInherit Class ViewRowCollection     Implements C1.LiveLinq.IObservableSource(Of ViewRow)</pre>	
C#	
<pre>[System.Diagnostics.DebuggerDisplay(Value="Count = {Count}",     Name="",     Type="",</pre>	

```

        Target=,
        TargetTypeName="")]
```

[System.Diagnostics.DebuggerTypeProxy(ProxyTypeName="C1.LiveLinq.SequenceDebugView, C1.Silverlight.LiveLinq.5, Version=5.0.20151.455, Culture=neutral, PublicKeyToken=2aa4ec5576d6c3ce",

```

        Target=,
        TargetTypeName="")]
```

[System.Reflection.DefaultMember("Item")]

```

public abstract class ViewRowCollection :
C1.LiveLinq.IObservableSource<ViewRow>
```

## Remarks

A **ViewRowCollection** is owned by a view, see [View.Rows](#).

The collection of *view rows* ([ViewRow](#) objects) is always synchronized with the collection of view elements.

[ViewRow](#) objects provide programmatic access to view elements and their properties. Also, the [View.Rows](#) collection serves as the data source for data binding, when you bind a control or another client to a view.

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.ViewRowCollection**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Members](#)

[C1.LiveLinq.LiveViews Namespace](#)











## Members

[Properties](#) [Methods](#) [Events](#)

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRowCollection Class](#)


The following tables list the members exposed by [ViewRowCollection](#).








## Public Properties

	Name	Description
	<a href="#">AllowClear</a>	Gets a value indicating whether the <a href="#">Clear</a> operation is allowed on the view directly.
	<a href="#">AllowEdit</a>	Gets a value indicating whether modifying property values is allowed on the view directly.
	<a href="#">AllowNew</a>	Gets a value indicating whether the <a href="#">CreateRow</a> operation is allowed on the view directly.
	<a href="#">AllowRemove</a>	Gets a value indicating whether deleting rows is allowed on the view directly.
	<a href="#">Count</a>	Gets the number of elements in the view.
	<a href="#">GroupDescriptions</a>	Gets a collection of <b>System.ComponentModel.GroupDescription</b> objects that describe how the items in the collection are grouped.
	<a href="#">Groups</a>	Gets the top-level groups.
	<a href="#">Item</a>	Gets the view row at the specified ordinal position.
	<a href="#">Properties</a>	Returns the collection of properties available in the view element type to programmatic access through <a href="#">ViewRow</a> and to data binding.
	<a href="#">SortDescriptions</a>	Gets a collection of <b>System.ComponentModel.SortDescription</b> objects that describe how the items in the collection are sorted.

[Top](#)



## Public Methods

	Name	Description
	<a href="#">Clear</a>	Deletes all view elements.

	<a href="#">Contains</a>	Determines whether the <a href="#">ViewRowCollection</a> contains a specific view row.
	<a href="#">CreateRow</a>	Creates a new item directly in the view, and a view row associated with it, and adds it to the <a href="#">ViewRowCollection</a> .
	<a href="#">DeferRefresh</a>	Enters a defer cycle that you can use to merge changes to the view and delay automatic refresh.
	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the collection.
	<a href="#">IndexOf</a>	Determines the ordinal position of a specific view row in the <a href="#">ViewRowCollection</a> .
	<a href="#">Remove</a>	Deletes the specified view item.
	<a href="#">RemoveAt</a>	Deletes the view row at a specified ordinal position in <a href="#">ViewRowCollection</a> .

[Top](#)

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after a view row has changed.
	<a href="#">ViewRowAdding</a>	Occurs after a new view row is created so it can be populated with default values.

[Top](#)

## See Also

### Reference

[ViewRowCollection Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

## Methods

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRowCollection Class](#)

For a list of all members of this type, see [ViewRowCollection members](#).

## Public Methods

	Name	Description
≡	<a href="#">Clear</a>	Deletes all view elements.
≡	<a href="#">Contains</a>	Determines whether the <a href="#">ViewRowCollection</a> contains a specific view row.
≡	<a href="#">CreateRow</a>	Creates a new item directly in the view, and a view row associated with it, and adds it to the <a href="#">ViewRowCollection</a> .
≡	<a href="#">DeferRefresh</a>	Enters a defer cycle that you can use to merge changes to the view and delay automatic refresh.
≡	<a href="#">GetEnumerator</a>	Returns an enumerator that iterates through the collection.
≡	<a href="#">IndexOf</a>	Determines the ordinal position of a specific view row in the <a href="#">ViewRowCollection</a> .
≡	<a href="#">Remove</a>	Deletes the specified view item.
≡	<a href="#">RemoveAt</a>	Deletes the view row at a specified ordinal position in <a href="#">ViewRowCollection</a> .

[Top](#)

## See Also

### Reference

[ViewRowCollection Class](#)

[C1.LiveInq.LiveViews Namespace](#)

### Clear Method

[C1.LiveInq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Clear Method

Deletes all view elements.

## Syntax

Visual Basic (Declaration)

```
Public Sub Clear()
```

```
C#
```

```
public void Clear()
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### Contains Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Contains Method

The view row to locate in the collection.

Determines whether the [ViewRowCollection](#) contains a specific view row.

## Syntax

Visual Basic (Declaration)

```
Public Function Contains( _  
    ByVal row As ViewRow _  
) As System.Boolean
```

```
C#
```

```
public System.bool Contains(  
    ViewRow row  
)
```

### Parameters

*row*

The view row to locate in the collection.

### Return Value

**true** if the view row is found in the collection; otherwise, **false**.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### CreateRow Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : CreateRow Method

Creates a new item directly in the view, and a view row associated with it, and adds it to the [ViewRowCollection](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Function CreateRow() As ViewRow</pre>	
C#	
<pre>public ViewRow CreateRow()</pre>	

### Return Value

The newly created view row.

## Remarks

Creating a new row is an update operation on the view. As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)), and it results in updating (in this case, adding a new item to) one of the view's base data collections.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ViewRowCollection Class](#)  
[ViewRowCollection Members](#)

## DeferRefresh Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : DeferRefresh Method

Enters a defer cycle that you can use to merge changes to the view and delay automatic refresh.

# Syntax

Visual Basic (Declaration)	
<code>Public Function DeferRefresh() As System.IDisposable</code>	
C#	
<code>public System.IDisposable DeferRefresh()</code>	

## Return Value

An **System.IDisposable** object that you can use to dispose of the calling object.

# Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

# See Also

## Reference

[ViewRowCollection Class](#)  
[ViewRowCollection Members](#)

## GetEnumerator Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : GetEnumerator Method

Returns an enumerator that iterates through the collection.

# Syntax

Visual Basic (Declaration)	
----------------------------	--



```
Public Function GetEnumerator() As System.Collections.Generic.IEnumerator(Of
ViewRow)
```

C#

```
public System.Collections.Generic.IEnumerator<ViewRow> GetEnumerator()
```

## Return Value

A **System.Collections.Generic.IEnumerator`1** that can be used to iterate through the collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### IndexOf Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : IndexOf(ViewRow) Method

The view row to locate in the collection.

Determines the ordinal position of a specific view row in the [ViewRowCollection](#).

## Syntax

Visual Basic (Declaration)

```
Public Function IndexOf( _
    ByVal row As ViewRow _
) As System.Integer
```

C#

```
public System.int IndexOf(
    ViewRow row
)
```

### Parameters

*row*

The view row to locate in the collection.

## Return Value

The ordinal position of the view row in the collection if it is found; otherwise, -1.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

## Remove Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Remove Method

The view row representing the item to delete.

Deletes the specified view item.

## Syntax

Visual Basic (Declaration)	
<pre>Public Function Remove( _     ByVal row As ViewRow _ ) As System.Boolean</pre>	
C#	
<pre>public System.bool Remove(     ViewRow row )</pre>	

## Parameters

*row*

The view row representing the item to delete.

## Return Value

**true**, if the item was deleted as a result of this operation; otherwise, **false**.

## Remarks

This is an update operation on the view equivalent to calling [ViewRow.Delete](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### RemoveAt Method

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : RemoveAt Method

The zero-based ordinal position of the item to remove.

Deletes the view row at a specified ordinal position in [ViewRowCollection](#).

## Syntax

Visual Basic (Declaration)

```
Public Sub RemoveAt( _  
    ByVal ordinal As System.Integer _  
)
```

C#

```
public void RemoveAt(  
    System.int ordinal  
)
```

### Parameters

*ordinal*

The zero-based ordinal position of the item to remove.

## Remarks

This is an update operation on the view equivalent to calling [ViewRow.Delete](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)








[ViewRowCollection Members](#)




## Properties

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowCollection Class

For a list of all members of this type, see [ViewRowCollection members](#).

## Public Properties

	Name	Description
	<a href="#">AllowClear</a>	Gets a value indicating whether the <a href="#">Clear</a> operation is allowed on the view directly.
	<a href="#">AllowEdit</a>	Gets a value indicating whether modifying property values is allowed on the view directly.
	<a href="#">AllowNew</a>	Gets a value indicating whether the <a href="#">CreateRow</a> operation is allowed on the view directly.
	<a href="#">AllowRemove</a>	Gets a value indicating whether deleting rows is allowed on the view directly.
	<a href="#">Count</a>	Gets the number of elements in the view.
	<a href="#">GroupDescriptions</a>	Gets a collection of <b>System.ComponentModel.GroupDescription</b> objects that describe how the items in the collection are grouped.
	<a href="#">Groups</a>	Gets the top-level groups.

	<a href="#">Item</a>	Gets the view row at the specified ordinal position.
	<a href="#">Properties</a>	Returns the collection of properties available in the view element type to programmatic access through <a href="#">ViewRow</a> and to data binding.
	<a href="#">SortDescriptions</a>	Gets a collection of <b>System.ComponentModel.SortDescription</b> objects that describe how the items in the collection are sorted.

[Top](#)

## See Also

### Reference

[ViewRowCollection Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### AllowClear Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : AllowClear Property

Gets a value indicating whether the [Clear](#) operation is allowed on the view directly.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> AllowClear <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool AllowClear {get;}	

## Remarks

As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)). Note that the same operation is often allowed on the base data collection, and it will normally result in the corresponding change of the view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### AllowEdit Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : AllowEdit Property

Gets a value indicating whether modifying property values is allowed on the view directly.

## Syntax

Visual Basic (Declaration)	
<b>Public ReadOnly Property</b> AllowEdit <b>As</b> System.Boolean	
C#	
<b>public</b> System.bool AllowEdit { <b>get</b> ;}	

## Remarks

As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)).

Although read-only properties of a view cannot be modified directly in the view, they still reflect up-to-date values of the source, so the difference is often not critical, you can always modify corresponding property in the source, that will automatically change the property in the view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### AllowNew Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : AllowNew Property

Gets a value indicating whether the [CreateRow](#) operation is allowed on the view directly.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property AllowNew As System.Boolean</code>	
C#	
<code>public System.bool AllowNew {get;}</code>	

## Remarks

As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)). Note that the same operation is usually allowed on the base data collection, and it will normally result in the corresponding change of the view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### AllowRemove Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : AllowRemove Property

Gets a value indicating whether deleting rows is allowed on the view directly.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property AllowRemove As System.Boolean</code>	
C#	
<code>public System.bool AllowRemove {get;}</code>	

## Remarks

As any other update operation performed directly on the view (as opposed to on the base data collection on which that view depends, see [C1.LiveLinq.LiveViewExtensions.AsUpdatable<T>](#)), it is allowed only if the view is not read-only (see [View.IsReadOnly](#)). Note that the same operation is often allowed on the base data collection, and it will normally result in the corresponding change of the view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### Count Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Count Property

Gets the number of elements in the view.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Count As System.Integer</code>	
C#	
<code>public System.int Count {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference



[ViewRowCollection Class](#)  
[ViewRowCollection Members](#)

## GroupDescriptions Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : GroupDescriptions Property

Gets a collection of **System.ComponentModel.GroupDescription** objects that describe how the items in the collection are grouped.

## Syntax

Visual Basic (Declaration)	
<pre>Public ReadOnly Property GroupDescriptions As System.Collections.ObjectModel.ObservableCollection(Of GroupDescription)</pre>	
C#	
<pre>public System.Collections.ObjectModel.ObservableCollection&lt;GroupDescription&gt; GroupDescriptions {get;}</pre>	

## Property Value

A collection of **System.ComponentModel.GroupDescription** objects that describe how the items in the collection are grouped.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)  
[ViewRowCollection Members](#)

## Groups Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Groups Property

Gets the top-level groups.

## Syntax

Visual Basic (Declaration)	
----------------------------	--

Public ReadOnly Property Groups As System.Collections.ObjectModel.ReadOnlyObservableCollection(Of Object)	
C#	
public System.Collections.ObjectModel.ReadOnlyObservableCollection<object> Groups {get;}	

### Property Value

A read-only collection of the top-level groups.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)  
[ViewRowCollection Members](#)

### Item Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Item Property

The zero-based ordinal position of the view row.

Gets the view row at the specified ordinal position.

## Syntax

Visual Basic (Declaration)	
Public ReadOnly Default Property Item( _ ByVal ordinal As System.Integer _ ) As ViewRow	
C#	
public ViewRow this[ System.int ordinal ]; {get;}	

### Parameters

*ordinal*

The zero-based ordinal position of the view row.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### Properties Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Properties Property

Returns the collection of properties available in the view element type to programmatic access through [ViewRow](#) and to data binding.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property Properties As ViewRowPropertyInfo()</code>	
C#	
<code>public ViewRowPropertyInfo[] Properties {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

## SortDescriptions Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : SortDescriptions Property

Gets a collection of **System.ComponentModel.SortDescription** objects that describe how the items in the collection are sorted.

## Syntax

Visual Basic (Declaration)	
<pre>Public ReadOnly Property SortDescriptions As System.ComponentModel.SortDescriptionCollection</pre>	
C#	
<pre>public System.ComponentModel.SortDescriptionCollection SortDescriptions {get;}</pre>	

### Property Value

A collection of **System.ComponentModel.SortDescription** objects that describe how the items in the collection are sorted.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)


[ViewRowCollection Members](#)


## Events

[C1.LiveLinq.LiveViews Namespace](#) : [ViewRowCollection Class](#)

For a list of all members of this type, see [ViewRowCollection members](#).

## Public Events

	Name	Description
	<a href="#">Changed</a>	Occurs after a view row has changed.

	<b>ViewRowAdding</b>	Occurs after a new view row is created so it can be populated with default values.
---	----------------------	--

[Top](#)

## See Also

### Reference

[ViewRowCollection Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

### Changed Event

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : Changed Event

Occurs after a view row has changed.

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> Changed <b>As</b> System.EventHandler(Of SourceChangeEventArgs(Of ViewRow))	
C#	
<b>public event</b> System.EventHandler<SourceChangeEventArgs<ViewRow>> Changed	

## Event Data

The event handler receives an argument of type [SourceChangeEventArgs<T>](#) containing data related to this event. The following **SourceChangeEventArgs<T>** properties provide information specific to this event.

Property	Description
<a href="#">ChangeType</a>	Gets the type of change.
<a href="#">Item</a>	Gets the object that is being changed.
<a href="#">Ordinal</a>	Gets the ordinal position of the collection item that is being changed.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)

[ViewRowCollection Members](#)

### ViewRowAdding Event

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowCollection Class](#) : ViewRowAdding Event

Occurs after a new view row is created so it can be populated with default values.

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> ViewRowAdding <b>As</b> System.EventHandler(Of ViewRowAddingEventArgs)	
C#	
<b>public event</b> System.EventHandler<ViewRowAddingEventArgs> ViewRowAdding	

## Event Data

The event handler receives an argument of type [ViewRowAddingEventArgs](#) containing data related to this event. The following **ViewRowAddingEventArgs** properties provide information specific to this event.

Property	Description
<a href="#">Row</a>	Gets the new view row that has just been added to <a href="#">ViewRowCollection</a> .

## Remarks

This event occurs only if the new row is created directly in the view, as a result of a view update operation, that is, with [CreateRow](#) or via data binding. It does not occur when new rows appear in the view as a result of view maintenance on changes made to the view's source data collections.

This event occurs immediately on creating the new row, before the method creating it returns, before the row enters edit mode (see [ViewRowState](#) about edit mode).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowCollection Class](#)  
[ViewRowCollection Members](#)

## ViewRowPropertyInfo

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowPropertyInfo Class

Allows to control certain behavior of a property of the element type of a [View](#).

## Object Model

ViewRowPropertyInfo

## Syntax

Visual Basic (Declaration)	
<b>Public Class</b> ViewRowPropertyInfo	
C#	
<b>public class</b> ViewRowPropertyInfo	

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.ViewRowPropertyInfo**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowPropertyInfo Members](#)  
[C1.LiveLinq.LiveViews Namespace](#)

## Overview

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowPropertyInfo Class

Allows to control certain behavior of a property of the element type of a [View](#).

## Object Model

ViewRowPropertyInfo

## Syntax

Visual Basic (Declaration)

```
Public Class ViewRowPropertyInfo
```

C#

```
public class ViewRowPropertyInfo
```

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.ViewRowPropertyInfo**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowPropertyInfo Members](#)

[C1.LiveLinq.LiveViews Namespace](#)

## Members



[Properties](#)

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowPropertyInfo Class

The following tables list the members exposed by [ViewRowPropertyInfo](#).

## Public Properties



	Name	Description
	<a href="#">ImmediateUpdate</a>	Gets or sets a boolean value indicating whether changes made to this property through data binding must be immediately sent to the corresponding view item even if the view is in editing mode.
	<a href="#">PropertyName</a>	Gets the name of the property.

[Top](#)

## See Also



### Reference

[ViewRowPropertyInfo Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

## Properties

>

Name	Description
 <a href="#">ImmediateUpdate</a>	Gets or sets a boolean value indicating whether changes made to this property through data binding must be immediately sent to the corresponding view item even if the view is in editing mode.
 <a href="#">PropertyName</a>	Gets the name of the property.

[Top](#)

## See Also

### Reference

[ViewRowPropertyInfo Class](#)

[C1.LiveLinq.LiveViews Namespace](#)

## ImmediateUpdate Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowPropertyInfo Class](#) : ImmediateUpdate Property

Gets or sets a boolean value indicating whether changes made to this property through data binding must be immediately sent to the corresponding view item even if the view is in editing mode.

## Syntax

Visual Basic (Declaration)	
<code>Public Property ImmediateUpdate As System.Boolean</code>	
C#	
<code>public System.bool ImmediateUpdate {get; set;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ViewRowPropertyInfo Class](#)

[ViewRowPropertyInfo Members](#)

[BeginEdit Method](#)

### PropertyName Property

[C1.LiveLinq.LiveViews Namespace](#) > [ViewRowPropertyInfo Class](#) : PropertyName Property

Gets the name of the property.

## Syntax

Visual Basic (Declaration)	
<code>Public ReadOnly Property PropertyName As System.String</code>	
C#	
<code>public System.string PropertyName {get;}</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

## Enumerations

### ViewMaintenanceMode

[C1.LiveInq.LiveViews Namespace](#) : ViewMaintenanceMode Enumeration

Specifies how a view is synchronized with changes in its base data.

### Syntax

Visual Basic (Declaration)	
<pre>Public Enum ViewMaintenanceMode     Inherits System.Enum</pre>	
C#	
<pre>public enum ViewMaintenanceMode : System.Enum</pre>	

### Members

Member	Description
<b>Default</b>	<p>By default, the view is in a "smart mode": It is in <b>Deferred</b> mode if nobody is interested in its data, and it is synchronized and goes to <b>Immediate</b> mode if there is a client interested in receiving notifications of changes in that view's data.</p> <p>In other words, a view in <b>Default</b> mode is effectively in <b>Deferred</b> mode if no listeners registered to be notified of its changes, and it is effectively in <b>Immediate</b> mode if there are such listeners (for example, if there is GUI control bound to it).</p>
<b>Deferred</b>	<p>When a change in base data occur, the view is not synchronized with it immediately. It is allowed to go stale, out of sync with its base data as long as there are no requests for this view's data. The view is synchronized on demand, that is, it is synchronized when any request for its data arrives.</p>
<b>Immediate</b>	<p>The view is synchronized with its base data automatically and immediately after any change in its base data occurs. The view is kept synchronized with</p>

	its base data at all times.
--	-----------------------------

## Remarks

**See Also:** View Maintenance Mode.

## Inheritance Hierarchy

System.Object

System.ValueType

System.Enum

**C1.LiveLinq.LiveViews.ViewMaintenanceMode**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.LiveLinq.LiveViews Namespace](#)

[MaintenanceMode Property](#)

[DeferredMaintenance Property](#)

## ViewOrder

[C1.LiveLinq.LiveViews Namespace](#) : ViewOrder Enumeration

Specifies whether and how a view must preserve item order if it exists in the source.

## Syntax

Visual Basic (Declaration)	
<pre><b>Public Enum</b> ViewOrder     <b>Inherits</b> System.Enum</pre>	
C#	
<pre><b>public enum</b> ViewOrder : System.Enum</pre>	

## Members

Member	Description
<b>NotPreserved</b>	Source order is not preserved. Preserving source order is not guaranteed even at view creation.
<b>PartiallyPreserved</b>	Source order is partially preserved. When the view is created, it preserves source order, but later, when changes occur in the source, view items added or modified to reflect those changes aren't guaranteed to appear at the same order position in the view as in the source.
<b>Preserved</b>	Source order is preserved completely. Order of items in the view is always the same as in the source (if source has a particular order), even after the view is maintained to reflect changes that occurred in the source.

## Inheritance Hierarchy

System.Object

System.ValueType

System.Enum

**C1.LiveLinq.LiveViews.ViewOrder**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1.LiveLinq.LiveViews Namespace](#)

## ViewRowState

[C1.LiveLinq.LiveViews Namespace](#) : ViewRowState Enumeration

The state of a view row with regard to edit, add and delete operations if they are performed directly on the view.

## Syntax

Visual Basic (Declaration)

<b>Public Enum</b> ViewState <b>Inherits</b> System.Enum
C#
<b>public enum</b> ViewState : System.Enum

## Members

Member	Description
<b>Detached</b>	The row was deleted, or it is a new row after exiting edit mode.
<b>Modified</b>	The row is in edit mode ( <a href="#">BeginEdit</a> was called, neither <a href="#">EndEdit</a> nor <a href="#">ViewRow.CancelEdit</a> was called yet), and the row is not new (was not created by <a href="#">CreateRow</a> ).
<b>New</b>	The row was added to the view directly (not by adding to a basic data collection) by calling <a href="#">CreateRow</a> or through data binding (such new row enters edit mode once it is created) and still in edit mode (neither <a href="#">EndEdit</a> nor <a href="#">ViewRow.CancelEdit</a> was called yet).
<b>Unmodified</b>	The row is a regular view row, not in edit mode and not deleted.

## Remarks

This state concerns edit, add and delete operations performed directly on the view (programmatically via [ViewRow](#) objects or through data binding). It does not concern modifications made to the view's base (source) data collections. Modifications made to source data can also change view items, as a result of the normal view maintenance process, see [MaintenanceMode](#), but in that case the state of thus added or modified rows remains **Unmodified**.

### Note on adding rows directly to the view:

If a row is added directly to the view (as opposed to adding it to one of its base data collections), the following happens:

When a new row is created with [CreateRow](#) or through data binding, it enters edit mode.

When it is committed with [EndEdit](#), a new row is added to the view's base data collection, and, usually, a corresponding row appears in the view, that has **Unmodified** state, although in some cases it may be more than one row or none, depending on the view query. The original view row no longer corresponds to a view item after the [EndEdit](#) or [ViewRow.CancelEdit](#) call, its state becomes **Detached**. Accessing it after that would throw an exception.

## Inheritance Hierarchy

System.Object  
  System.ValueType  
    System.Enum  
      **C1.LiveLinq.LiveViews.ViewRowState**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also


### Reference

[C1.LiveLinq.LiveViews Namespace](#)

# C1.LiveLinq.LiveViews.Xml Namespace

## Overview

## Classes

	Class	Description
	<a href="#">XmlExtensions</a>	Provides a set of static (extension) methods for LiveLinq to XML.

## See Also

### Reference

[C1.Silverlight.LiveLinq.5 Assembly](#)

## Classes

## XmlExtensions

[C1.LiveLinq.LiveViews.Xml Namespace](#) : XmlExtensions Class

Provides a set of static (extension) methods for LiveLinq to XML.

## Object Model

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public MustInherit NotInheritable Class XmlExtensions</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static class XmlExtensions</pre>	

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.Xml.XmlExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Members](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#)

## Overview

[C1.LiveLinq.LiveViews.Xml Namespace](#) : XmlExtensions Class

Provides a set of static (extension) methods for LiveLinq to XML.

## Object Model

## Syntax

Visual Basic (Declaration)	
----------------------------	--



```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public MustInherit NotInheritable Class XmlExtensions
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static class XmlExtensions
```

## Inheritance Hierarchy

System.Object

**C1.LiveLinq.LiveViews.Xml.XmlExtensions**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Members](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#)







## Members









[Methods](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#) : XmlExtensions Class

The following tables list the members exposed by [XmlExtensions](#).

## Public Methods

	Name	Description
 	<a href="#">AsLive</a>	Overloaded. Creates a view based on the specified XML document.
 	<a href="#">Attributes</a>	Overloaded. Returns a view representing the collection of the attributes of every element in the source view.
 	<a href="#">BeginUpdate</a>	Suspends notifications while massive changes are being

		made to an XML node and its descendants.
≡  <a href="#">DescendantNodes&lt;T&gt;</a>		Returns a view representing the collection of the descendent nodes of every element and document in the source view.
≡  <a href="#">DescendantNodesAndSelf</a>		Returns a view representing the collection of nodes that contains every element in the source view, and the descendent nodes of every element in the source view.
≡  <a href="#">Descendants</a>		Overloaded. Returns a view representing the collection of elements that contains the descendent elements of every element and document in the source view.
≡  <a href="#">DescendantsAndSelf</a>		Overloaded. Returns a view representing a collection of elements that contains every element in the source view, and the descendent elements of every element in the source view.
≡  <a href="#">Elements</a>		Overloaded. Returns a view representing the collection of child elements of every element and document in the source view..
≡  <a href="#">EndUpdate</a>		Ends notification suspension started with <a href="#">BeginUpdate</a> .
≡  <a href="#">Nodes&lt;T&gt;</a>		Returns a view representing the collection of child nodes of every document and element in the source view.
≡  <a href="#">Root</a>		Gets the view for the root element of the XML tree for this document.

[Top](#)

## See Also












### Reference

[XmlExtensions Class](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#)

## Methods

>

Name	Description
 <a href="#">AsLive</a>	Overloaded. Creates a view based on the specified XML document.
 <a href="#">Attributes</a>	Overloaded. Returns a view representing the collection of the attributes of every element in the source view.
 <a href="#">BeginUpdate</a>	Suspends notifications while massive changes are being made to an XML node and its descendants.
 <a href="#">DescendantNodes&lt;T&gt;</a>	Returns a view representing the collection of the descendent nodes of every element and document in the source view.
 <a href="#">DescendantNodesAndSelf</a>	Returns a view representing the collection of nodes that contains every element in the source view, and the descendent nodes of every element in the source view.
 <a href="#">Descendants</a>	Overloaded. Returns a view representing the collection of elements that contains the descendent elements of every element and document in the source view.
 <a href="#">DescendantsAndSelf</a>	Overloaded. Returns a view representing a collection of elements that contains every element in the source view, and the descendent elements of every element in the source view.
 <a href="#">Elements</a>	Overloaded. Returns a view representing the collection of child elements of every element and document in the source view..
 <a href="#">EndUpdate</a>	Ends notification suspension started with <a href="#">BeginUpdate</a> .
 <a href="#">Nodes&lt;T&gt;</a>	Returns a view representing the collection of child nodes of every document and element in the source view.
 <a href="#">Root</a>	Gets the view for the root element of the XML tree for this document.

[Top](#)

## See Also

### Reference

[XmlExtensions Class](#)

[C1.LiveLinq.LiveViews.Xml Namespace](#)

## AsLive Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : AsLive Method

Creates a view based on the specified XML document.

## Overload List

Overload	Description
<a href="#">AsLive(XDocument)</a>	Creates a view based on the specified XML document.
<a href="#">AsLive(XElement)</a>	Creates a view based on the specified XML element.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

### AsLive(XDocument) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [AsLive Method](#) : AsLive(XDocument) Method

The XML document to expose as a view.

Creates a view based on the specified XML document.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt;&gt; Public Overloads Shared Function AsLive( _     ByVal document As System.Xml.Linq.XDocument _ ) As View(Of XDocument)</pre>	

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static View<XDocument> AsLive(  
    System.Xml.Linq.XDocument document  
)
```

## Parameters

*document*

The XML document to expose as a view.

## Return Value

A view representing the specified XML document.

## Remarks

Since there can be only one root element in a document, the view based on a document (`document.AsLive()`) is similar to the view based on its root element (`document.Root.AsLive()`). The difference is that the document view contains the document as its only item, and the root view contains the root as its only item. This difference is essential only when the root of the document is replaced with a different element (and so the whole XML tree changes), then the document view remains valid and shows the changed document contents, whereas the root view becomes disconnected from the document.

**Note:** This view is owned by the **System.Xml.Linq.XDocument** object (it is stored as one of its annotations), so, if you create a view for the same document several times, it will be the same object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

[Overload List](#)

## AsLive(XElement) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [AsLive Method](#) : AsLive(XElement) Method

The XML element to expose as a view.

Creates a view based on the specified XML element.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function AsLive( _     ByVal element As System.Xml.Linq.XElement _ ) As View(Of XElement)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XElement&gt; AsLive(     System.Xml.Linq.XElement element )</pre>	

## Parameters

*element*

The XML element to expose as a view.

## Return Value

A view representing the specified XML element.

## Remarks

This view represents a single XML node. Therefore, as a collection of items, this view's **Count** is always 1. This view is usually used as a starting point to construct a view containing elements or attributes from this node's descendants by using a query with operators from [XmlExtensions](#) such as **Elements**, **Descendants** and others, in combination with standard LINQ query operators **where**, **join** and others.

**Note:** This view is owned by the **System.Xml.Linq.XElement** object (it is stored as one of its annotations), so, if you create a view for the same element several times, it will be the same object.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

### Attributes Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : Attributes Method

Returns a view representing the collection of the attributes of every element in the source view.

## Overload List

Overload	Description
<a href="#">Attributes(View&lt;XElement&gt;)</a>	Returns a view representing the collection of the attributes of every element in the source view.
<a href="#">Attributes(View&lt;XElement&gt;,XName)</a>	Returns a view representing a filtered collection of the attributes of every element in the source view. Only attributes that have a matching <b>System.Xml.Linq.XName</b> are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)

### Attributes(View<XElement>) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [Attributes Method](#) :

Attributes(View<XElement>) Method

The source view.

Returns a view representing the collection of the attributes of every element in the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function Attributes( _     ByVal view As View(Of XElement) _ ) As View(Of XAttribute)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XAttribute&gt; Attributes(     View&lt;XElement&gt; view )</pre>	

### Parameters

*view*

The source view.

### Return Value

A view containing the attributes of every element in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

Attributes(View<XElement>,XName) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [Attributes Method](#) :



Attributes(View<XElement>,XName) Method

The source view.

The **System.Xml.Linq.XName** to match.

Returns a view representing a filtered collection of the attributes of every element in the source view. Only attributes that have a matching **System.Xml.Linq.XName** are included.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function Attributes( _     ByVal view As View(Of XElement), _     ByVal name As System.Xml.Linq.XName _ ) As View(Of XAttribute)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XAttribute&gt; Attributes(     View&lt;XElement&gt; view,     System.Xml.Linq.XName name )</pre>	

## Parameters

*view*

The source view.

*name*

The **System.Xml.Linq.XName** to match.

## Return Value

A view that contains a filtered collection of the attributes of every element in the source view. Only attributes that have a matching **System.Xml.Linq.XName** are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

### BeginUpdate Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : BeginUpdate Method

The node that is the root of a tree where massive changes are made in code.

Suspends notifications while massive changes are being made to an XML node and its descendants.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Sub BeginUpdate( _     ByVal node As System.Xml.Linq.XContainer _ )</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static void BeginUpdate(     System.Xml.Linq.XContainer node )</pre>	

### Parameters

*node*

The node that is the root of a tree where massive changes are made in code.

## Remarks

This method must be followed by [EndUpdate](#).

Use this method when you already have indexes over this XML or live views based on it, and you need to perform massive changes on the contents of this node and its descendants. Without this method, every single change you make causes LiveLinq to perform necessary operations for maintaining your indexes and live views dependent on this node and its descendants. In case of massive changes, this can be slower than to wait until the massive changes are done and rebuild the indexes and live views.

Between **BeginUpdate** and [EndUpdate](#) calls, indexes, live views, bound controls and other change notification listeners are not updated, they don't receive change notifications. When [EndUpdate](#) is called, a [SourceChangeType.Reset](#) notification is sent, meaning all indexes, live views and other collections dependent on this node and its descendants must be rebuilt from scratch.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)

### DescendantNodes<T> Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : DescendantNodes<T> Method

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

The source view.

Returns a view representing the collection of the descendent nodes of every element and document in the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Shared Function DescendantNodes(Of T As System.Xml.Linq.XContainer)( _  
    ByVal view As View(Of T) _  
) As View(Of XElement)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static View<XNode> DescendantNodes<T>(  
    View<T> view  
)  
where T: System.Xml.Linq.XContainer
```

### Parameters

*view*

The source view.

## Type Parameters

*T*

The type of the objects in the source view, constrained to **System.Xml.Linq.XContainer**.

## Return Value

A view containing every descendent node of every document and element in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

## DescendantNodesAndSelf Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : DescendantNodesAndSelf Method

The source view.

Returns a view representing the collection of nodes that contains every element in the source view, and the descendent nodes of every element in the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Shared Function DescendantNodesAndSelf( _  
    ByVal view As View(Of XElement) _  
) As View(Of XNode)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<XNode> DescendantNodesAndSelf(
    View<XElement> view
)
```

## Parameters

*view*

The source view.

## Return Value

A view containing every element in the source view, and the descendent nodes of every element in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

## Descendants Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : Descendants Method

Returns a view representing the collection of elements that contains the descendent elements of every element and document in the source view.

## Overload List

Overload	Description
<a href="#">Descendants&lt;T&gt;(View&lt;T&gt;)</a>	Returns a view representing the collection of elements that contains the descendent elements of every element and document in the source view.
<a href="#">Descendants&lt;T&gt;(View&lt;T&gt;,XName)</a>	Returns a view representing a filtered collection of elements that contains the descendent elements of every element and document in the source view. Only elements

	that have a matching <b>System.Xml.Linq.XName</b> are included.
--	---

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)

Descendants<T>(View<T>) Method  
[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [Descendants Method](#) :  
 Descendants<T>(View<T>) Method

The type of the objects in the source view, constrained to **System.Xml.Linq.XContainer**.

The source view.

Returns a view representing the collection of elements that contains the descendent elements of every element and document in the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function Descendants(Of T As System.Xml.Linq.XContainer)( _     ByVal view As View(Of T) _ ) As View(Of XElement)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XElement&gt; Descendants&lt;T&gt;(     View&lt;T&gt; view ) where T: System.Xml.Linq.XContainer</pre>	

### Parameters

*view*

The source view.

## Type Parameters

*T*

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

## Return Value

A view containing every descendent element of every element and document in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

[Overload List](#)

Descendants<T>(View<T>,XName) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [Descendants Method](#) :

Descendants<T>(View<T>,XName) Method

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

The source view.

The **System.Xml.Linq.XName** to match.

Returns a view representing a filtered collection of elements that contains the descendent elements of every element and document in the source view. Only elements that have a matching **System.Xml.Linq.XName** are included.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
```

```
Public Overloads Shared Function Descendants(Of T As
System.Xml.Linq.XContainer)( _
    ByVal view As View(Of T), _
    ByVal name As System.Xml.Linq.XName _
) As View(Of XElement)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<XElement> Descendants<T>(
    View<T> view,
    System.Xml.Linq.XName name
)
where T: System.Xml.Linq.XContainer
```

## Parameters

*view*

The source view.

*name*

The **System.Xml.Linq.XName** to match.

## Type Parameters

*T*

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

## Return Value

A view containing descendants of elements and documents in the source view. Only elements that have a matching **System.Xml.Linq.XName** are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference



[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

## DescendantsAndSelf Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : DescendantsAndSelf Method

Returns a view representing a collection of elements that contains every element in the source view, and the descendent elements of every element in the source view.

## Overload List

Overload	Description
<a href="#">DescendantsAndSelf(View&lt;XElement&gt;)</a>	Returns a view representing a collection of elements that contains every element in the source view, and the descendent elements of every element in the source view.
<a href="#">DescendantsAndSelf(View&lt;XElement&gt;,XName)</a>	Returns a view representing a filtered collection of elements that contains every element in the source view, and the descendants of every element in the source view. Only elements that have a matching <b>System.Xml.Linq.XName</b> are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)

DescendantsAndSelf(View<XElement>) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [DescendantsAndSelf Method](#) :  
DescendantsAndSelf(View<XElement>) Method

The source view.

Returns a view representing a collection of elements that contains every element in the source view, and the descendent elements of every element in the source view.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function DescendantsAndSelf( _     ByVal view As View(Of XElement) _ ) As View(Of XElement)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XElement&gt; DescendantsAndSelf(     View&lt;XElement&gt; view )</pre>	

### Parameters

*view*

The source view.

### Return Value

A view containing every element in the source view, and the descendent elements of every element in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

[Overload List](#)

DescendantsAndSelf(View<XElement>,XName) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [DescendantsAndSelf Method](#) :

DescendantsAndSelf(View<XElement>,XName) Method

The source view.

The **System.Xml.Linq.XName** to match.

Returns a view representing a filtered collection of elements that contains every element in the source view, and the descendants of every element in the source view. Only elements that have a matching **System.Xml.Linq.XName** are included.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function DescendantsAndSelf( _     ByVal view As View(Of XElement), _     ByVal name As System.Xml.Linq.XName _ ) As View(Of XElement)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XElement&gt; DescendantsAndSelf(     View&lt;XElement&gt; view,     System.Xml.Linq.XName name )</pre>	

## Parameters

*view*

The source view.

*name*

The **System.Xml.Linq.XName** to match.

## Return Value

A view containing elements in the source view, and the descendants of elements in the source view. Only elements that have a matching **System.Xml.Linq.XName** are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

### Elements Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : Elements Method

Returns a view representing the collection of child elements of every element and document in the source view..

## Overload List

Overload	Description
<a href="#">Elements&lt;T&gt;(View&lt;T&gt;)</a>	Returns a view representing the collection of child elements of every element and document in the source view..
<a href="#">Elements&lt;T&gt;(View&lt;T&gt;,XName)</a>	Returns a view representing filtered collection of child elements of every element and document in the source view. Only elements that have a matching <b>System.Xml.Linq.XName</b> are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)

### Elements<T>(View<T>) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [Elements Method](#) :  
Elements<T>(View<T>) Method

The type of the objects in the source view, constrained to **System.Xml.Linq.XContainer**.

The source view.

Returns a view representing the collection of child elements of every element and document in the source view..

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Overloads Shared Function Elements(Of T As System.Xml.Linq.XContainer)( _     ByVal view As View(Of T) _ ) As View(Of XElement)</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static View&lt;XElement&gt; Elements&lt;T&gt;(     View&lt;T&gt; view ) where T: System.Xml.Linq.XContainer</pre>	

### Parameters

*view*

The source view.

### Type Parameters

*T*

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

### Return Value

A view containing the child elements of every element and document in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

Elements<T>(View<T>,XName) Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) > [Elements Method](#) :

Elements<T>(View<T>,XName) Method

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

The source view.

The **System.Xml.Linq.XName** to match.

Returns a view representing filtered collection of child elements of every element and document in the source view. Only elements that have a matching **System.Xml.Linq.XName** are included.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Overloads Shared Function Elements(Of T As  
System.Xml.Linq.XContainer)( _  
    ByVal view As View(Of T), _  
    ByVal name As System.Xml.Linq.XName _  
) As View(Of XElement)
```

C#

```
[System.Runtime.CompilerServices.Extension()]  
public static View<XElement> Elements<T>(  
    View<T> view,  
    System.Xml.Linq.XName name  
)  
where T: System.Xml.Linq.XContainer
```

### Parameters

*view*

The source view.

*name*

The **System.Xml.Linq.XName** to match.

### Type Parameters

*T*

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

## Return Value

A view that contains a filtered collection of child elements of every element and document in the source view. Only elements that have a matching **System.Xml.Linq.XName** are included.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)  
[XmlExtensions Members](#)  
[Overload List](#)

## EndUpdate Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : EndUpdate Method

The node that is the root of a tree where massive changes have been made since the [BeginUpdate](#) call.

Ends notification suspension started with [BeginUpdate](#).

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.Runtime.CompilerServices.ExtensionAttribute(&gt; Public Shared Sub EndUpdate( _     ByVal node As System.Xml.Linq.XContainer _ )</pre>	
C#	
<pre>[System.Runtime.CompilerServices.Extension()] public static void EndUpdate(     System.Xml.Linq.XContainer node</pre>	

```
)
```

## Parameters

*node*

The node that is the root of a tree where massive changes have been made since the [BeginUpdate](#) call.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

### Nodes<T> Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : Nodes<T> Method

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

The source view.

Returns a view representing the collection of child nodes of every document and element in the source view.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>
Public Shared Function Nodes(Of T As System.Xml.Linq.XContainer)( _
    ByVal view As View(Of T) _
) As View(Of XElement)
```

C#

```
[System.Runtime.CompilerServices.Extension()]
public static View<XNode> Nodes<T>(
    View<T> view
)
```



**where** T: System.Xml.Linq.XContainer

## Parameters

*view*

The source view.

## Type Parameters

*T*

The type of the objects in the source *view*, constrained to **System.Xml.Linq.XContainer**.

## Return Value

A view containing every child node of every document and element in the source view.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

## Root Method

[C1.LiveLinq.LiveViews.Xml Namespace](#) > [XmlExtensions Class](#) : Root Method

The view representing this XML document.

Gets the view for the root element of the XML tree for this document.

## Syntax

Visual Basic (Declaration)

```
<System.Runtime.CompilerServices.ExtensionAttribute(>  
Public Shared Function Root( _  
    ByVal documentView As View(Of XElement) _  
) As View(Of XElement)
```

C#

```
[System.Runtime.CompilerServices.Extension()]\npublic static View<XElement> Root(\n    View<XDocument> documentView\n)
```

### Parameters

*documentView*

The view representing this XML document.

### Return Value

A view containing a single element, the root of the XML tree.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[XmlExtensions Class](#)

[XmlExtensions Members](#)

## C1.LiveLinq.Metadata Namespace

## Overview

[Inheritance Hierarchy](#)

## See Also

### Reference

[C1.Silverlight.LiveLinq.5 Assembly](#)


# C1.WPF.Data.Entity.4 Assembly

## Namespaces

### C1.WPF.Data Namespace

#### Overview

#### Classes

	Class	Description
	<a href="#">ControlHandler</a>	Represents a control handler that connect GUI controls of supported types to a C1DataSource so that those controls can be given additional functionality such as <a href="#">lookup columns</a> and <a href="#">virtual mode</a> .

#### See Also

#### Reference

[C1.WPF.Data.Entity.4 Assembly](#)

## Classes

### ControlHandler

[C1.WPF.Data Namespace](#) : ControlHandler Class

Represents a control handler that connect GUI controls of supported types to a C1DataSource so that those controls can be given additional functionality such as [lookup columns](#) and [virtual mode](#).

#### Object Model



#### Syntax

Visual Basic (Declaration)	
<pre>Public Class ControlHandler     Inherits C1.Data.DataSource.BaseControlHandler</pre>	

C#

```
public class ControlHandler : C1.Data.DataSource.BaseControlHandler
```

## Remarks

To connect a [control handler](#) to a GUI control, assign an instance of this class to the [C1.WPF.Data.Entities.C1DataSource.ControlHandlerProperty](#) attached property of the GUI control.

The supported GUI controls are: **System.Windows.Controls.DataGrid**, [C1.WPF.FlexGrid.C1FlexGrid](#), [C1.WPF.DataGrid.C1DataGrid](#).

## Inheritance Hierarchy

System.Object  
  System.Windows.Threading.DispatcherObject  
    System.Windows.DependencyObject  
      [C1.Data.DataSource.BaseControlHandler](#)  
        **C1.WPF.Data.ControlHandler**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ControlHandler Members](#)  
[C1.WPF.Data Namespace](#)

### Overview

[C1.WPF.Data Namespace](#) : ControlHandler Class

Represents a control handler that connect GUI controls of supported types to a [C1DataSource](#) so that those controls can be given additional functionality such as [lookup columns](#) and [virtual mode](#).

## Object Model

ControlHandler

## Syntax

Visual Basic (Declaration)	
<pre>Public Class ControlHandler     Inherits C1.Data.DataSource.BaseControlHandler</pre>	
C#	
<pre>public class ControlHandler : C1.Data.DataSource.BaseControlHandler</pre>	

## Remarks

To connect a [control handler](#) to a GUI control, assign an instance of this class to the [C1.WPF.Data.Entities.C1DataSource.ControlHandlerProperty](#) attached property of the GUI control.

The supported GUI controls are: **System.Windows.Controls.DataGrid**, C1.WPF.FlexGrid.C1FlexGrid, C1.WPF.DataGrid.C1DataGrid.

## Inheritance Hierarchy

```
System.Object
  System.Windows.Threading.DispatcherObject
    System.Windows.DependencyObject
      C1.Data.DataSource.BaseControlHandler
        C1.WPF.Data.ControlHandler
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ControlHandler Members](#)  
[C1.WPF.Data Namespace](#)


### Members

[Fields](#) [Properties](#) [Methods](#)

[C1.WPF.Data Namespace](#) : ControlHandler Class


The following tables list the members exposed by [ControlHandler](#).

## Public Constructors

	Name	Description
	<a href="#">ControlHandler Constructor</a>	








[Top](#)

## Public Fields

	Name	Description
	<a href="#">DataSourceProperty</a>	The DependencyProperty for the <a href="#">DataSource</a> property.

[Top](#)

## Public Properties

	Name	Description
	<a href="#">AutoLookup</a>	Gets or sets a value indicating whether data grid columns bound to navigation (foreign key, lookup) properties must be converted to combo box columns, so the user can see the right value and edit it by choosing a value from a drop-down list. The default value is False. (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )
	<a href="#">DataSource</a>	Gets or sets a <a href="#">data source</a> of the control handler.
	<a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)
	<a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SupportsVirtualMode</a>	Gets a value indicating whether this <a href="#">control handler</a> supports Virtual Mode. (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )
	<a href="#">VirtualMode</a>	Gets or sets a value indicating whether virtual mode specified in a <a href="#">C1.Data.DataSource.ClientViewSource</a> is managed by this control

		handler. (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )
--	--	--

[Top](#)


## Public Methods

	Name	Description
≡	<a href="#">Apply</a>	Forces this <a href="#">control handler</a> to apply its settings to the current control. (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )
≡	<a href="#">ClearValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)
≡	<a href="#">CoerceValue</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">Equals</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">GetHashCode</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">GetLocalValueEnumerator</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">InvalidateProperty</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">SetCurrentValue</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">SetValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)

[Top](#)

## Protected Methods

	Name	Description
--	------	-------------

	<a href="#">OnPropertyChanged</a>	(Inherited from <a href="#">System.Windows.DependencyObject</a> )
---	-----------------------------------	---

[Top](#)

## See Also

### Reference

[ControlHandler Class](#)  
[C1.WPF.Data Namespace](#)

## ControlHandler Constructor

[C1.WPF.Data Namespace](#) > [ControlHandler Class](#) : ControlHandler Constructor

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public ControlHandler()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[ControlHandler Class](#)  
[ControlHandler Members](#)

## Properties








[C1.WPF.Data Namespace](#) : ControlHandler Class

For a list of all members of this type, see [ControlHandler members](#).

## Public Properties

Name	Description



	<a href="#">AutoLookup</a>	Gets or sets a value indicating whether data grid columns bound to navigation (foreign key, lookup) properties must be converted to combo box columns, so the user can see the right value and edit it by choosing a value from a drop-down list. The default value is False. (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )
	<a href="#">DataSource</a>	Gets or sets a <a href="#">data source</a> of the control handler.
	<a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)
	<a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">SupportsVirtualMode</a>	Gets a value indicating whether this <a href="#">control handler</a> supports Virtual Mode. (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )
	<a href="#">VirtualMode</a>	Gets or sets a value indicating whether virtual mode specified in a <a href="#">C1.Data.DataSource.ClientViewSource</a> is managed by this control handler. (Inherited from <a href="#">C1.Data.DataSource.BaseControlHandler</a> )

[Top](#)

## See Also

### Reference

[ControlHandler Class](#)

[C1.WPF.Data Namespace](#)

### DataSource Property

[C1.WPF.Data Namespace](#) > [ControlHandler Class](#) : DataSource Property

Gets or sets a [data source](#) of the control handler.

## Syntax

Visual Basic (Declaration)

Public Property DataSource As C1DataSource	
C#	
public C1DataSource DataSource {get; set;}	

## Remarks

Setting this property is required if the GUI control is not bound directly to a C1DataSource. For example, it must be set if the GUI control is bound to a [live view](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference


[ControlHandler Class](#)  
[ControlHandler Members](#)

### Fields

[C1.WPF.Data Namespace](#) : ControlHandler Class

For a list of all members of this type, see [ControlHandler members](#).

## Public Fields

	Name	Description
 S	<a href="#">DataSourceProperty</a>	The DependencyProperty for the <a href="#">DataSource</a> property.

[Top](#)

## See Also

### Reference

[ControlHandler Class](#)  
[C1.WPF.Data Namespace](#)

### DataSourceProperty Field

[C1.WPF.Data Namespace](#) > [ControlHandler Class](#) : DataSourceProperty Field

The `DependencyProperty` for the [DataSource](#) property.

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared ReadOnly DataSourceProperty As System.Windows.DependencyProperty</pre>	
C#	
<pre>public static readonly System.Windows.DependencyProperty DataSourceProperty</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also



### Reference

[ControlHandler Class](#)  
[ControlHandler Members](#)

# C1.WPF.Data.Entities Namespace

## Overview

## Classes

	Class	Description
	<a href="#">C1DataSource</a>	A data source control that simplifies data binding of GUI controls to data in a <a href="#">C1.Data.Entities.EntityClientCache</a> . Can be used to bind multiple controls to different queries.
	<a href="#">EntityViewSourceCollection</a>	An observable collection of <a href="#">C1.Data.Entities.EntityViewSource</a> objects.

## See Also

### Reference

## Classes

### C1DataSource

[C1.WPF.Data.Entities Namespace](#) : C1DataSource Class

A data source control that simplifies data binding of GUI controls to data in a [C1.Data.Entities.EntityClientCache](#). Can be used to bind multiple controls to different queries.

## Object Model

C1DataSource

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>
<System.Windows.Markup.ContentPropertyAttribute("ViewSources")>
<System.ComponentModel.LicenseProviderAttribute()>
Public Class C1DataSource
    Inherits System.Windows.Controls.Control
```

C#

```
[System.Reflection.DefaultMember("Item")]
[System.Windows.Markup.ContentProperty("ViewSources")]
[System.ComponentModel.LicenseProvider()]
public class C1DataSource : System.Windows.Controls.Control
```

## Remarks

To bind a control to data in a [C1.Data.Entities.EntityClientCache](#), add a [C1DataSource](#) to a XAML file, specify the [context type](#), populate the [ViewSources](#) collection with [C1.Data.Entities.EntityViewSource](#) objects to define views (based on queries), and bind GUI controls like this: `<DataGrid ItemsSource="{Binding Customers, ElementName=c1DataSource}" />` where Customers is the name of an [C1.Data.Entities.EntityViewSource](#) in the [ViewSources](#) collection and c1DataSource is the name of the [C1DataSource](#).

## Inheritance Hierarchy

System.Object

System.Windows.Threading.DispatcherObject

System.Windows.DependencyObject  
System.Windows.Media.Visual  
System.Windows.UIElement  
System.Windows.FrameworkElement  
System.Windows.Controls.Control  
**C1.WPF.Data.Entities.C1DataSource**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Members](#)

[C1.WPF.Data.Entities Namespace](#)

### Overview

[C1.WPF.Data.Entities Namespace](#) : C1DataSource Class

A data source control that simplifies data binding of GUI controls to data in a [C1.Data.Entities.EntityClientCache](#). Can be used to bind multiple controls to different queries.

## Object Model

C1DataSource

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>  
<System.Windows.Markup.ContentPropertyAttribute("ViewSources")>  
<System.ComponentModel.LicenseProviderAttribute()>  
Public Class C1DataSource  
    Inherits System.Windows.Controls.Control
```

C#

```
[System.Reflection.DefaultMember("Item")]  
[System.Windows.Markup.ContentProperty("ViewSources")]  
[System.ComponentModel.LicenseProvider()]
```

```
public class C1DataSource : System.Windows.Controls.Control
```

## Remarks

To bind a control to data in a [C1.Data.Entities.EntityClientCache](#), add a [C1DataSource](#) to a XAML file, specify the [context type](#), populate the [ViewSources](#) collection with [C1.Data.Entities.EntityViewSource](#) objects to define views (based on queries), and bind GUI controls like this: `<DataGrid ItemsSource="{Binding Customers, ElementName=c1DataSource}" />` where Customers is the name of an [C1.Data.Entities.EntityViewSource](#) in the [ViewSources](#) collection and c1DataSource is the name of the [C1DataSource](#).

## Inheritance Hierarchy

```
System.Object
  System.Windows.Threading.DispatcherObject
    System.Windows.DependencyObject
      System.Windows.Media.Visual
        System.Windows.UIElement
          System.Windows.FrameworkElement
            System.Windows.Controls.Control
              C1.WPF.Data.Entities.C1DataSource
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Members](#)

[C1.WPF.Data.Entities Namespace](#)

## Members


[Fields](#) [Properties](#) [Methods](#) [Events](#)

[C1.WPF.Data.Entities Namespace](#) : [C1DataSource Class](#)

The following tables list the members exposed by [C1DataSource](#).


## Public Constructors

Name	Description
------	-------------

	<a href="#">C1DataSource Constructor</a>	Initializes a new instance of the <a href="#">C1DataSource</a> class.
---	--	---












[Top](#)

















## Public Fields

	Name	Description
 <b>S</b>	<a href="#">ControlHandlerProperty</a>	Identifies the C1.WPF.Data.Entities.C1DataSource.ControlHandler attached property.




















[Top](#)




















## Public Properties


















	Name	Description
	<a href="#">ActualHeight</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ActualWidth</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">AllowDrop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">AreAnyTouchesCaptured</a>	(Inherited from System.Windows.UIElement)
	<a href="#">AreAnyTouchesCapturedWithin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">AreAnyTouchesDirectlyOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">AreAnyTouchesOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Background</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">BindingGroup</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">BitmapEffect</a>	(Inherited from System.Windows.UIElement)
	<a href="#">BitmapEffectInput</a>	(Inherited from System.Windows.UIElement)



















	<a href="#">BorderBrush</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">BorderThickness</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">CacheMode</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ClientCache</a>	Gets or sets the <a href="#">C1.Data.Entities.EntityClientCache</a> used by this <a href="#">C1DataSource</a> to access the data.
	<a href="#">ClientScope</a>	Gets the <a href="#">client scope</a> to which this <a href="#">C1DataSource</a> belongs.
	<a href="#">Clip</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ClipToBounds</a>	(Inherited from System.Windows.UIElement)
	<a href="#">CommandBindings</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ContextMenu</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ContextType</a>	Gets or sets the type of a <b>System.Data.Entity.DbContext</b> or an <b>System.Data.Objects.ObjectContext</b> used to obtain the <a href="#">default client cache</a> .
	<a href="#">Cursor</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DataContext</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DbContext</a>	Gets the <b>System.Data.Entity.DbContext</b> the <a href="#">ClientCache</a> is connected to.
	<a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">DesiredSize</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)









	Effect	(Inherited from System.Windows.UIElement)
	FlowDirection	(Inherited from System.Windows.FrameworkElement)
	Focusable	(Inherited from System.Windows.UIElement)
	FocusVisualStyle	(Inherited from System.Windows.FrameworkElement)
	FontFamily	(Inherited from System.Windows.Controls.Control)
	FontSize	(Inherited from System.Windows.Controls.Control)
	FontStretch	(Inherited from System.Windows.Controls.Control)
	FontStyle	(Inherited from System.Windows.Controls.Control)
	FontWeight	(Inherited from System.Windows.Controls.Control)
	ForceCursor	(Inherited from System.Windows.FrameworkElement)
	Foreground	(Inherited from System.Windows.Controls.Control)
	HasAnimatedProperties	(Inherited from System.Windows.UIElement)
	Height	(Inherited from System.Windows.FrameworkElement)
	HorizontalAlignment	(Inherited from System.Windows.FrameworkElement)
	HorizontalContentAlignment	(Inherited from System.Windows.Controls.Control)
	InputBindings	(Inherited from System.Windows.UIElement)
	InputScope	(Inherited from System.Windows.FrameworkElement)
	IsArrangeValid	(Inherited from System.Windows.UIElement)
	IsEnabled	(Inherited from System.Windows.UIElement)

	<a href="#">IsFocused</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsHitTestVisible</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsInitialized</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">IsInputMethodEnabled</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsKeyboardFocused</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsKeyboardFocusWithin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsLoaded</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">IsManipulationEnabled</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsMeasureValid</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsMouseCaptured</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsMouseCaptureWithin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsMouseDirectlyOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsMouseOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">IsStylusCaptured</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsStylusCaptureWithin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsStylusDirectlyOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsStylusOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsTabStop</a>	(Inherited from System.Windows.Controls.Control)











 <a href="#">IsVisible</a>	(Inherited from System.Windows.UIElement)
 <a href="#">Item</a>	Overloaded. Gets the <a href="#">C1.Data.DataSource.ClientCollectionView</a> of the <a href="#">C1.Data.Entities.EntityViewSource</a> with the specified <a href="#">name</a> in the <a href="#">ViewSources</a> collection.
 <a href="#">Language</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">LayoutTransform</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">Margin</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">MaxHeight</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">MaxWidth</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">MinHeight</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">MinWidth</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">Name</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">ObjectContext</a>	Gets the <b>System.Data.Objects.ObjectContext</b> the <a href="#">ClientCache</a> is connected to.
 <a href="#">Opacity</a>	(Inherited from System.Windows.UIElement)
 <a href="#">OpacityMask</a>	(Inherited from System.Windows.UIElement)
 <a href="#">OverridesDefaultStyle</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">Padding</a>	(Inherited from System.Windows.Controls.Control)
 <a href="#">Parent</a>	(Inherited from System.Windows.FrameworkElement)
 <a href="#">PersistId</a>	(Inherited from System.Windows.UIElement)









	<a href="#">RefreshInterval</a>	Gets or sets the interval between automatic <a href="#">Refresh</a> operations to refresh the data with any changes that may have occurred on the server.
	<a href="#">RenderSize</a>	(Inherited from System.Windows.UIElement)
	<a href="#">RenderTransform</a>	(Inherited from System.Windows.UIElement)
	<a href="#">RenderTransformOrigin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Resources</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">SnapsToDevicePixels</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Style</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">TabIndex</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">Tag</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Template</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">TemplatedParent</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ToolTip</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">TouchesCaptured</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TouchesCapturedWithin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TouchesDirectlyOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TouchesOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Triggers</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Uid</a>	(Inherited from System.Windows.UIElement)

	<a href="#">UseLayoutRounding</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">VerticalAlignment</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">VerticalContentAlignment</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">ViewSources</a>	Gets a collection of <a href="#">C1.Data.Entities.EntityViewSource</a> objects that define views (based on queries) in this <a href="#">C1DataSource</a> .
	<a href="#">Visibility</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Width</a>	(Inherited from System.Windows.FrameworkElement)

[Top](#)









## Protected Properties


















	Name	Description
	<a href="#">IsEnabledCore</a>	(Inherited from System.Windows.UIElement)
	<a href="#">StylusPlugIns</a>	(Inherited from System.Windows.UIElement)
	<a href="#">VisualBitmapEffect</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualBitmapEffectInput</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualBitmapScalingMode</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualCacheMode</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualChildrenCount</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">VisualClip</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualEdgeMode</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualEffect</a>	(Inherited from System.Windows.Media.Visual)

	<a href="#">VisualOffset</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualOpacity</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualOpacityMask</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualParent</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualScrollableAreaClip</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualTransform</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualXSnappingGuidelines</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualYSnappingGuidelines</a>	(Inherited from System.Windows.Media.Visual)

[Top](#)

## Public Methods




	Name	Description
	<a href="#">AddHandler</a>	Overloaded. (Inherited from System.Windows.UIElement)
	<a href="#">AddToEventRoute</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ApplyAnimationClock</a>	Overloaded. (Inherited from System.Windows.UIElement)
	<a href="#">ApplyTemplate</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Arrange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">BeginAnimation</a>	Overloaded. (Inherited from System.Windows.UIElement)
	<a href="#">BeginInit</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">BeginStoryboard</a>	Overloaded. (Inherited from System.Windows.FrameworkElement)

≡ 	<a href="#">BringIntoView</a>	Overloaded. (Inherited from System.Windows.FrameworkElement)
≡ 	<a href="#">CaptureMouse</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">CaptureStylus</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">CaptureTouch</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">ClearValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">CoerceValue</a>	(Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">EndInit</a>	(Inherited from System.Windows.FrameworkElement)
≡ 	<a href="#">Equals</a>	(Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">FindCommonVisualAncestor</a>	(Inherited from System.Windows.Media.Visual)
≡ 	<a href="#">FindName</a>	(Inherited from System.Windows.FrameworkElement)
≡ 	<a href="#">FindResource</a>	(Inherited from System.Windows.FrameworkElement)
≡ 	<a href="#">Focus</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">GetAnimationBaseValue</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">GetBindingExpression</a>	(Inherited from System.Windows.FrameworkElement)
≡  <b>S</b>	<a href="#">GetControlHandler</a>	Gets the value of the C1.WPF.Data.Entities.C1DataSource.ControlHandler attached property from a given <a href="#">control</a> .
≡ 	<a href="#">GetHashCode</a>	(Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">GetLocalValueEnumerator</a>	(Inherited from System.Windows.DependencyObject)

≡	<a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">InputHitTest</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">InvalidateArrange</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">InvalidateMeasure</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">InvalidateProperty</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">InvalidateVisual</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">IsAncestorOf</a>	(Inherited from System.Windows.Media.Visual)
≡	<a href="#">IsDescendantOf</a>	(Inherited from System.Windows.Media.Visual)
≡	<a href="#">Load</a>	Loads all <a href="#">C1.Data.Entities.EntityViewSource</a> objects in the <a href="#">ViewSources</a> collection.
≡	<a href="#">Measure</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">MoveFocus</a>	(Inherited from System.Windows.FrameworkElement)
≡	<a href="#">OnApplyTemplate</a>	(Inherited from System.Windows.FrameworkElement)
≡	<a href="#">PointFromScreen</a>	(Inherited from System.Windows.Media.Visual)
≡	<a href="#">PointToScreen</a>	(Inherited from System.Windows.Media.Visual)
≡	<a href="#">PredictFocus</a>	(Inherited from System.Windows.FrameworkElement)
≡	<a href="#">RaiseEvent</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">Refresh</a>	Refreshes all <a href="#">C1.Data.Entities.EntityViewSource</a> objects in the <a href="#">ViewSources</a> collection.

































≡	<a href="#">RegisterName</a>	(Inherited from System.Windows.FrameworkElement)
≡	<a href="#">RejectChanges</a>	Rejects the changes for every entity in the context.
≡	<a href="#">ReleaseAllTouchCaptures</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">ReleaseMouseCapture</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">ReleaseStylusCapture</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">ReleaseTouchCapture</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">RemoveHandler</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">SaveChanges</a>	Persists all changes to the server.
≡	<a href="#">SetBinding</a>	Overloaded. (Inherited from System.Windows.FrameworkElement)
≡ S	<a href="#">SetControlHandler</a>	Sets the value of the C1.WPF.Data.Entities.C1DataSource.ControlHandler attached property to a given <a href="#">control</a> .
≡	<a href="#">SetCurrentValue</a>	(Inherited from System.Windows.DependencyObject)
≡	<a href="#">SetResourceReference</a>	(Inherited from System.Windows.FrameworkElement)
≡	<a href="#">SetValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)
≡	<a href="#">ToString</a>	(Inherited from System.Windows.Controls.Control)
≡	<a href="#">TransformToAncestor</a>	Overloaded. (Inherited from System.Windows.Media.Visual)
≡	<a href="#">TransformToDescendant</a>	(Inherited from System.Windows.Media.Visual)
≡	<a href="#">TransformToVisual</a>	(Inherited from System.Windows.Media.Visual)




















	<a href="#">TranslatePoint</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TryFindResource</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">UnregisterName</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">UpdateDefaultStyle</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">UpdateLayout</a>	(Inherited from System.Windows.UIElement)




















[Top](#)




















## Protected Methods




















	Name	Description
	<a href="#">AddVisualChild</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">ArrangeCore</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ArrangeOverride</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">GetLayoutClip</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">GetVisualChild</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">HitTestCore</a>	Overloaded. (Inherited from System.Windows.UIElement)
	<a href="#">MeasureCore</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">MeasureOverride</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnAccessKey</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnChildDesiredSizeChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnContextMenuClosing</a>	(Inherited from System.Windows.FrameworkElement)








	<a href="#">OnContextMenuOpening</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnCreateAutomationPeer</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnDragEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnDragLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnDragOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnDrop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnGiveFeedback</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnGotFocus</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnGotKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnGotMouseCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnGotStylusCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnGotTouchCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnInitialized</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnIsKeyboardFocusedChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsKeyboardFocusWithinChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsMouseCapturedChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsMouseCaptureWithinChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsMouseDirectlyOverChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsStylusCapturedChanged</a>	(Inherited from System.Windows.UIElement)

	<a href="#">OnIsStylusCaptureWithinChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsStylusDirectlyOverChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnKeyDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnKeyUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnLostFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnLostKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnLostMouseCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnLostStylusCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnLostTouchCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnManipulationBoundaryFeedback</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnManipulationCompleted</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnManipulationDelta</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnManipulationInertiaStarting</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnManipulationStarted</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnManipulationStarting</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseDoubleClick</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseLeave</a>	(Inherited from System.Windows.UIElement)

	<a href="#">OnMouseLeftButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseLeftButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseRightButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseRightButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseWheel</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewDragEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewDragLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewDragOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewDrop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewGiveFeedback</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewGotKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewKeyDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewKeyUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewLostKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseDoubleClick</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnPreviewMouseDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseLeftButtonDown</a>	(Inherited from System.Windows.UIElement)











	<a href="#">OnPreviewMouseLeftButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseRightButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseRightButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseWheel</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewQueryContinueDrag</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusInAirMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusInRange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusOutOfRange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusSystemGesture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewTextInput</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewTouchDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewTouchMove</a>	(Inherited from System.Windows.UIElement)

	<a href="#">OnPreviewTouchUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnQueryContinueDrag</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnQueryCursor</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnRender</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnStylusButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnStylusButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnStylusDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnStylusEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnStylusInAirMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnStylusInRange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnStylusLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnStylusMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnStylusOutOfRange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnStylusSystemGesture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnStylusUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnTemplateChanged</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnTextInput</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnToolTipClosing</a>	(Inherited from System.Windows.FrameworkElement)




















	<a href="#">OnToolTipOpening</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnTouchDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnTouchEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnTouchLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnTouchMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnTouchUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">RemoveVisualChild</a>	(Inherited from System.Windows.Media.Visual)




















[Top](#)




















## Public Events




















	Name	Description
	<a href="#">ContextMenuClosing</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ContextMenuOpening</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DataContextChanged</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DragEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">DragLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">DragOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Drop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">FocusableChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">GiveFeedback</a>	(Inherited from System.Windows.UIElement)
	<a href="#">GotFocus</a>	(Inherited from System.Windows.UIElement)





























	GotKeyboardFocus	(Inherited from System.Windows.UIElement)
	GotMouseCapture	(Inherited from System.Windows.UIElement)
	GotStylusCapture	(Inherited from System.Windows.UIElement)
	GotTouchCapture	(Inherited from System.Windows.UIElement)
	Initialized	(Inherited from System.Windows.FrameworkElement)
	IsEnabledChanged	(Inherited from System.Windows.UIElement)
	IsHitTestVisibleChanged	(Inherited from System.Windows.UIElement)
	IsKeyboardFocusedChanged	(Inherited from System.Windows.UIElement)
	IsKeyboardFocusWithinChanged	(Inherited from System.Windows.UIElement)
	IsMouseCapturedChanged	(Inherited from System.Windows.UIElement)
	IsMouseCaptureWithinChanged	(Inherited from System.Windows.UIElement)
	IsMouseDirectlyOverChanged	(Inherited from System.Windows.UIElement)
	IsStylusCapturedChanged	(Inherited from System.Windows.UIElement)
	IsStylusCaptureWithinChanged	(Inherited from System.Windows.UIElement)
	IsStylusDirectlyOverChanged	(Inherited from System.Windows.UIElement)
	IsVisibleChanged	(Inherited from System.Windows.UIElement)
	KeyDown	(Inherited from System.Windows.UIElement)
	KeyUp	(Inherited from System.Windows.UIElement)
	LayoutUpdated	(Inherited from System.Windows.UIElement)

	<a href="#">Loaded</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">LostFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">LostKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">LostMouseCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">LostStylusCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">LostTouchCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ManipulationBoundaryFeedback</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ManipulationCompleted</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ManipulationDelta</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ManipulationInertiaStarting</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ManipulationStarted</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ManipulationStarting</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseDoubleClick</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">MouseDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseLeftButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseLeftButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseMove</a>	(Inherited from System.Windows.UIElement)

	<a href="#">MouseRightButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseRightButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseWheel</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewDragEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewDragLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewDragOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewDrop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewGiveFeedback</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewGotKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewKeyDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewKeyUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewLostKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseDoubleClick</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">PreviewMouseDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseLeftButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseLeftButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseRightButtonDown</a>	(Inherited from System.Windows.UIElement)

	<a href="#">PreviewMouseRightButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseWheel</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewQueryContinueDrag</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusInAirMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusInRange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusOutOfRange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusSystemGesture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewTextInput</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewTouchDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewTouchMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewTouchUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">QueryContinueDrag</a>	(Inherited from System.Windows.UIElement)
	<a href="#">QueryCursor</a>	(Inherited from System.Windows.UIElement)

	<a href="#">RequestBringIntoView</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">SavedChanges</a>	Occurs after a save operation is completed.
	<a href="#">SavingChanges</a>	Occurs before changes are saved.
	<a href="#">SizeChanged</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">SourceUpdated</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">StylusButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">StylusButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">StylusDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">StylusEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">StylusInAirMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">StylusInRange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">StylusLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">StylusMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">StylusOutOfRange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">StylusSystemGesture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">StylusUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TargetUpdated</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">TextInput</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ToolTipClosing</a>	(Inherited from System.Windows.FrameworkElement)

	<a href="#">ToolTipOpening</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">TouchDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TouchEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TouchLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TouchMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TouchUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Unloaded</a>	(Inherited from System.Windows.FrameworkElement)

[Top](#)

## See Also

### Reference

[C1DataSource Class](#)

[C1.WPF.Data.Entities Namespace](#)

## C1DataSource Constructor

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : C1DataSource Constructor

Initializes a new instance of the [C1DataSource](#) class.

## Syntax

Visual Basic (Declaration)	
<code>Public Function New()</code>	
C#	
<code>public C1DataSource()</code>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)



















### Methods

[C1.WPF.Data.Entities Namespace](#) : C1DataSource Class



















For a list of all members of this type, see [C1DataSource members](#).



















### Public Methods



	Name	Description
≡	<a href="#">AddHandler</a>	Overloaded. (Inherited from System.Windows.UIElement)
≡	<a href="#">AddToEventRoute</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">ApplyAnimationClock</a>	Overloaded. (Inherited from System.Windows.UIElement)
≡	<a href="#">ApplyTemplate</a>	(Inherited from System.Windows.FrameworkElement)
≡	<a href="#">Arrange</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">BeginAnimation</a>	Overloaded. (Inherited from System.Windows.UIElement)
≡	<a href="#">BeginInit</a>	(Inherited from System.Windows.FrameworkElement)
≡	<a href="#">BeginStoryboard</a>	Overloaded. (Inherited from System.Windows.FrameworkElement)
≡	<a href="#">BringIntoView</a>	Overloaded. (Inherited from System.Windows.FrameworkElement)
≡	<a href="#">CaptureMouse</a>	(Inherited from System.Windows.UIElement)
≡	<a href="#">CaptureStylus</a>	(Inherited from System.Windows.UIElement)

≡ 	<a href="#">CaptureTouch</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">ClearValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">CoerceValue</a>	(Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">EndInit</a>	(Inherited from System.Windows.FrameworkElement)
≡ 	<a href="#">Equals</a>	(Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">FindCommonVisualAncestor</a>	(Inherited from System.Windows.Media.Visual)
≡ 	<a href="#">FindName</a>	(Inherited from System.Windows.FrameworkElement)
≡ 	<a href="#">FindResource</a>	(Inherited from System.Windows.FrameworkElement)
≡ 	<a href="#">Focus</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">GetAnimationBaseValue</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">GetBindingExpression</a>	(Inherited from System.Windows.FrameworkElement)
≡  	<a href="#">GetControlHandler</a>	Gets the value of the C1.WPF.Data.Entities.C1DataSource.ControlHandler attached property from a given <a href="#">control</a> .
≡ 	<a href="#">GetHashCode</a>	(Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">GetLocalValueEnumerator</a>	(Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">GetValue</a>	(Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">InputHitTest</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">InvalidateArrange</a>	(Inherited from System.Windows.UIElement)

















	<a href="#">InvalidateMeasure</a>	(Inherited from System.Windows.UIElement)
	<a href="#">InvalidateProperty</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">InvalidateVisual</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsAncestorOf</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">IsDescendantOf</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">Load</a>	Loads all <a href="#">C1.Data.Entities.EntityViewSource</a> objects in the <a href="#">ViewSources</a> collection.
	<a href="#">Measure</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MoveFocus</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnApplyTemplate</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">PointFromScreen</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">PointToScreen</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">PredictFocus</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">RaiseEvent</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ReadLocalValue</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">Refresh</a>	Refreshes all <a href="#">C1.Data.Entities.EntityViewSource</a> objects in the <a href="#">ViewSources</a> collection.
	<a href="#">RegisterName</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">RejectChanges</a>	Rejects the changes for every entity in the context.
	<a href="#">ReleaseAllTouchCaptures</a>	(Inherited from System.Windows.UIElement)




















≡ 	<a href="#">ReleaseMouseCapture</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">ReleaseStylusCapture</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">ReleaseTouchCapture</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">RemoveHandler</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">SaveChanges</a>	Persists all changes to the server.
≡ 	<a href="#">SetBinding</a>	Overloaded. (Inherited from System.Windows.FrameworkElement)
≡  	<a href="#">SetControlHandler</a>	Sets the value of the C1.WPF.Data.Entities.C1DataSource.ControlHandler attached property to a given <a href="#">control</a> .
≡ 	<a href="#">SetCurrentValue</a>	(Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">SetResourceReference</a>	(Inherited from System.Windows.FrameworkElement)
≡ 	<a href="#">SetValue</a>	Overloaded. (Inherited from System.Windows.DependencyObject)
≡ 	<a href="#">ToString</a>	(Inherited from System.Windows.Controls.Control)
≡ 	<a href="#">TransformToAncestor</a>	Overloaded. (Inherited from System.Windows.Media.Visual)
≡ 	<a href="#">TransformToDescendant</a>	(Inherited from System.Windows.Media.Visual)
≡ 	<a href="#">TransformToVisual</a>	(Inherited from System.Windows.Media.Visual)
≡ 	<a href="#">TranslatePoint</a>	(Inherited from System.Windows.UIElement)
≡ 	<a href="#">TryFindResource</a>	(Inherited from System.Windows.FrameworkElement)
≡ 	<a href="#">UnregisterName</a>	(Inherited from System.Windows.FrameworkElement)




















	<a href="#">UpdateDefaultStyle</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">UpdateLayout</a>	(Inherited from System.Windows.UIElement)




















[Top](#)




















## Protected Methods







































	Name	Description
	<a href="#">AddVisualChild</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">ArrangeCore</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ArrangeOverride</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">GetLayoutClip</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">GetVisualChild</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">HitTestCore</a>	Overloaded. (Inherited from System.Windows.UIElement)
	<a href="#">MeasureCore</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">MeasureOverride</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnAccessKey</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnChildDesiredSizeChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnContextMenuClosing</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnContextMenuOpening</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnCreateAutomationPeer</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnDragEnter</a>	(Inherited from System.Windows.UIElement)

	<a href="#">OnDragLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnDragOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnDrop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnGiveFeedback</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnGotFocus</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnGotKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnGotMouseCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnGotStylusCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnGotTouchCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnInitialized</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnIsKeyboardFocusedChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsKeyboardFocusWithinChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsMouseCapturedChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsMouseCaptureWithinChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsMouseDirectlyOverChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsStylusCapturedChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsStylusCaptureWithinChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnIsStylusDirectlyOverChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnKeyDown</a>	(Inherited from System.Windows.UIElement)





	<a href="#">OnKeyUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnLostFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnLostKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnLostMouseCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnLostStylusCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnLostTouchCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnManipulationBoundaryFeedback</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnManipulationCompleted</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnManipulationDelta</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnManipulationInertiaStarting</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnManipulationStarted</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnManipulationStarting</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseDoubleClick</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnMouseDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseLeftButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseLeftButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseMove</a>	(Inherited from System.Windows.UIElement)

	<a href="#">OnMouseRightButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseRightButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnMouseWheel</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewDragEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewDragLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewDragOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewDrop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewGiveFeedback</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewGotKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewKeyDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewKeyUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewLostKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseDoubleClick</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">OnPreviewMouseDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseLeftButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseLeftButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseRightButtonDown</a>	(Inherited from System.Windows.UIElement)

	<a href="#">OnPreviewMouseRightButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewMouseWheel</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewQueryContinueDrag</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusInAirMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusInRange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusOutOfRange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusSystemGesture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewStylusUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewTextInput</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewTouchDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewTouchMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPreviewTouchUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">OnQueryContinueDrag</a>	(Inherited from System.Windows.UIElement)

 	<a href="#">OnQueryCursor</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnRender</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnStylusButtonDown</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnStylusButtonUp</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnStylusDown</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnStylusEnter</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnStylusInAirMove</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnStylusInRange</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnStylusLeave</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnStylusMove</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnStylusOutOfRange</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnStylusSystemGesture</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnStylusUp</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnTemplateChanged</a>	(Inherited from System.Windows.Controls.Control)
 	<a href="#">OnTextInput</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnToolTipClosing</a>	(Inherited from System.Windows.FrameworkElement)
 	<a href="#">OnToolTipOpening</a>	(Inherited from System.Windows.FrameworkElement)
 	<a href="#">OnTouchDown</a>	(Inherited from System.Windows.UIElement)
 	<a href="#">OnTouchEnter</a>	(Inherited from System.Windows.UIElement)



	<a href="#">OnTouchLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnTouchMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OnTouchUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">RemoveVisualChild</a>	(Inherited from System.Windows.Media.Visual)

[Top](#)

## See Also

### Reference

[C1DataSource Class](#)

[C1.WPF.Data.Entities Namespace](#)

### GetControlHandler Method

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : GetControlHandler Method

The control from which to read the property value.

Gets the value of the C1.WPF.Data.Entities.C1DataSource.ControlHandler attached property from a given [control](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared Function GetControlHandler( _     ByVal control As System.Windows.DependencyObject _ ) As BaseControlHandler</pre>	
C#	
<pre>public static BaseControlHandler GetControlHandler(     System.Windows.DependencyObject control )</pre>	

### Parameters

*control*

The control from which to read the property value.

### Return Value

The value of the `C1.WPF.Data.Entities.C1DataSource.ControlHandler` attached property.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

[ControlHandlerProperty Field](#)

### Load Method

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : Load Method

Loads all [C1.Data.Entities.EntityViewSource](#) objects in the [ViewSources](#) collection.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Load()	
C#	
<b>public void</b> Load()	

## Remarks

This method calls [Load](#) for all elements of the [ViewSources](#) collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

## Refresh Method

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : Refresh Method

Refreshes all [C1.Data.Entities.EntityViewSource](#) objects in the [ViewSources](#) collection.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> Refresh()	
C#	
<b>public void</b> Refresh()	

## Remarks

This method calls [C1.Data.DataSource.ClientViewSource.Refresh](#) for all elements of the [ViewSources](#) collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

## RejectChanges Method

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : RejectChanges Method

Rejects the changes for every entity in the context.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> RejectChanges()	
C#	
<b>public void</b> RejectChanges()	

## Remarks

Changes will be rejected for all entities in the context, including those that were not loaded through this [C1DataSource](#). This will also cancel a pending Add or Edit in the [collection views](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### SaveChanges Method

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : SaveChanges Method

Persists all changes to the server.

## Syntax

Visual Basic (Declaration)	
<b>Public Sub</b> SaveChanges()	
C#	
<b>public void</b> SaveChanges()	

## Remarks

Changes will be saved for all entities in the context, including those that were not loaded through this [C1DataSource](#). This will also commit a pending Add or Edit in the [collection views](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

## SetControlHandler Method

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : SetControlHandler Method

The object on which to set the C1.WPF.Data.Entities.C1DataSource.ControlHandler attached property.

The property value to set.

Sets the value of the C1.WPF.Data.Entities.C1DataSource.ControlHandler attached property to a given [control](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared Sub SetControlHandler( _     ByVal control As System.Windows.DependencyObject, _     ByVal handler As BaseControlHandler _ )</pre>	
C#	
<pre>public static void SetControlHandler(     System.Windows.DependencyObject control,     BaseControlHandler handler )</pre>	

## Parameters

*control*

The object on which to set the C1.WPF.Data.Entities.C1DataSource.ControlHandler attached property.

*handler*

The property value to set.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[C1DataSource Class](#)

[C1DataSource Members](#)














[ControlHandlerProperty Field](#)

















## Properties




















[C1.WPF.Data.Entities Namespace](#) : [C1DataSource Class](#)

For a list of all members of this type, see [C1DataSource members](#).




















## Public Properties

















	Name	Description
	<a href="#">ActualHeight</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ActualWidth</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">AllowDrop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">AreAnyTouchesCaptured</a>	(Inherited from System.Windows.UIElement)
	<a href="#">AreAnyTouchesCapturedWithin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">AreAnyTouchesDirectlyOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">AreAnyTouchesOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Background</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">BindingGroup</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">BitmapEffect</a>	(Inherited from System.Windows.UIElement)
	<a href="#">BitmapEffectInput</a>	(Inherited from System.Windows.UIElement)
	<a href="#">BorderBrush</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">BorderThickness</a>	(Inherited from System.Windows.Controls.Control)




















	<a href="#">CacheMode</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ClientCache</a>	Gets or sets the <a href="#">C1.Data.Entities.EntityClientCache</a> used by this <a href="#">C1DataSource</a> to access the data.
	<a href="#">ClientScope</a>	Gets the <a href="#">client scope</a> to which this <a href="#">C1DataSource</a> belongs.
	<a href="#">Clip</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ClipToBounds</a>	(Inherited from System.Windows.UIElement)
	<a href="#">CommandBindings</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ContextMenu</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ContextType</a>	Gets or sets the type of a <b>System.Data.Entity.DbContext</b> or an <b>System.Data.Objects.ObjectContext</b> used to obtain the <a href="#">default client cache</a> .
	<a href="#">Cursor</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DataContext</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DbContext</a>	Gets the <b>System.Data.Entity.DbContext</b> the <a href="#">ClientCache</a> is connected to.
	<a href="#">DependencyObjectType</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">DesiredSize</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Dispatcher</a>	(Inherited from System.Windows.Threading.DispatcherObject)
	<a href="#">Effect</a>	(Inherited from System.Windows.UIElement)
	<a href="#">FlowDirection</a>	(Inherited from System.Windows.FrameworkElement)





	<a href="#">Focusable</a>	(Inherited from System.Windows.UIElement)
	<a href="#">FocusVisualStyle</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">FontFamily</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">FontSize</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">FontStretch</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">FontStyle</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">FontWeight</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">ForceCursor</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Foreground</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">HasAnimatedProperties</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Height</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">HorizontalAlignment</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">HorizontalContentAlignment</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">InputBindings</a>	(Inherited from System.Windows.UIElement)
	<a href="#">InputScope</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">IsArrangeValid</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsEnabled</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsFocused</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsHitTestVisible</a>	(Inherited from System.Windows.UIElement)



	<a href="#">IsInitialized</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">IsInputMethodEnabled</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsKeyboardFocused</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsKeyboardFocusWithin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsLoaded</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">IsManipulationEnabled</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsMeasureValid</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsMouseCaptured</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsMouseCaptureWithin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsMouseDirectlyOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsMouseOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsSealed</a>	(Inherited from System.Windows.DependencyObject)
	<a href="#">IsStylusCaptured</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsStylusCaptureWithin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsStylusDirectlyOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsStylusOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">IsTabStop</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">IsVisible</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Item</a>	Overloaded. Gets the <a href="#">C1.Data.DataSource.ClientCollectionView</a> of the













		<a href="#">C1.Data.Entities.EntityViewSource</a> with the specified <a href="#">name</a> in the <a href="#">ViewSources</a> collection.
	<a href="#">Language</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">LayoutTransform</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Margin</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">MaxHeight</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">MaxWidth</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">MinHeight</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">MinWidth</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Name</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ObjectContext</a>	Gets the <b>System.Data.Objects.ObjectContext</b> the <a href="#">ClientCache</a> is connected to.
	<a href="#">Opacity</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OpacityMask</a>	(Inherited from System.Windows.UIElement)
	<a href="#">OverridesDefaultStyle</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Padding</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">Parent</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">PersistId</a>	(Inherited from System.Windows.UIElement)
	<a href="#">RefreshInterval</a>	Gets or sets the interval between automatic <a href="#">Refresh</a> operations to refresh the data with any changes that may have occurred on the server.







	<a href="#">RenderSize</a>	(Inherited from System.Windows.UIElement)
	<a href="#">RenderTransform</a>	(Inherited from System.Windows.UIElement)
	<a href="#">RenderTransformOrigin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Resources</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">SnapsToDevicePixels</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Style</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">TabIndex</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">Tag</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Template</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">TemplatedParent</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ToolTip</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">TouchesCaptured</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TouchesCapturedWithin</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TouchesDirectlyOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TouchesOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Triggers</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">Uid</a>	(Inherited from System.Windows.UIElement)
	<a href="#">UseLayoutRounding</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">VerticalAlignment</a>	(Inherited from System.Windows.FrameworkElement)

	<a href="#">VerticalContentAlignment</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">ViewSources</a>	Gets a collection of <a href="#">C1.Data.Entities.EntityViewSource</a> objects that define views (based on queries) in this <a href="#">C1DataSource</a> .
	<a href="#">Visibility</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Width</a>	(Inherited from System.Windows.FrameworkElement)

[Top](#)

## Protected Properties

	Name	Description
	<a href="#">IsEnabledCore</a>	(Inherited from System.Windows.UIElement)
	<a href="#">StylusPlugIns</a>	(Inherited from System.Windows.UIElement)
	<a href="#">VisualBitmapEffect</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualBitmapEffectInput</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualBitmapScalingMode</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualCacheMode</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualChildrenCount</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">VisualClip</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualEdgeMode</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualEffect</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualOffset</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualOpacity</a>	(Inherited from System.Windows.Media.Visual)

	<a href="#">VisualOpacityMask</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualParent</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualScrollableAreaClip</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualTransform</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualXSnappingGuidelines</a>	(Inherited from System.Windows.Media.Visual)
	<a href="#">VisualYSnappingGuidelines</a>	(Inherited from System.Windows.Media.Visual)

[Top](#)

## See Also

### Reference

[C1DataSource Class](#)

[C1.WPF.Data.Entities Namespace](#)

### ClientCache Property

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : ClientCache Property

Gets or sets the [C1.Data.Entities.EntityClientCache](#) used by this [C1DataSource](#) to access the data.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute()&gt; Public Property ClientCache As EntityClientCache</pre>	
C#	
<pre>[System.ComponentModel.DefaultValue()] public EntityClientCache ClientCache {get; set;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### ClientScope Property

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : ClientScope Property

Gets the [client scope](#) to which this [C1DataSource](#) belongs.

## Syntax

Visual Basic (Declaration)

```
Public ReadOnly Property ClientScope As EntityClientScope
```

C#

```
public EntityClientScope ClientScope {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### ContextType Property

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : ContextType Property

Gets or sets the type of a **System.Data.Entity.DbContext** or an **System.Data.Objects.ObjectContext** used to obtain the [default client cache](#).

## Syntax

Visual Basic (Declaration)

```
<System.ComponentModel.CategoryAttribute("Common")>  
<System.ComponentModel.DefaultValueAttribute()>
```

```
Public Property ContextType As System.Type
```

```
C#
```

```
[System.ComponentModel.Category("Common")]  
[System.ComponentModel.DefaultValue()]  
public System.Type ContextType {get; set;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### DbContext Property

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : DbContext Property

Gets the **System.Data.Entity.DbContext** the [ClientCache](#) is connected to.

## Syntax

```
Visual Basic (Declaration)
```

```
<System.ComponentModel.BrowsableAttribute(False)>  
Public ReadOnly Property DbContext As System.Data.Entity.DbContext
```

```
C#
```

```
[System.ComponentModel.Browsable(false)]  
public System.Data.Entity.DbContext DbContext {get;}
```

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

## Item Property

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : Item Property

Gets the [C1.Data.DataSource.ClientCollectionView](#) of the [C1.Data.Entities.EntityViewSource](#) with the specified [name](#) in the [ViewSources](#) collection.

## Overload List

Overload	Description
<a href="#">Item(String)</a>	Gets the <a href="#">C1.Data.DataSource.ClientCollectionView</a> of the <a href="#">C1.Data.Entities.EntityViewSource</a> with the specified <a href="#">name</a> in the <a href="#">ViewSources</a> collection.
<a href="#">Item(Int32)</a>	Gets the <a href="#">C1.Data.DataSource.ClientCollectionView</a> of the <a href="#">C1.Data.Entities.EntityViewSource</a> at the specified <a href="#">index</a> in the <a href="#">ViewSources</a> collection.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### Item(String) Property

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) > [Item Property](#) : Item(String) Property

The name of the [C1.Data.Entities.EntityViewSource](#) to take the [C1.Data.DataSource.ClientCollectionView](#) from.

Gets the [C1.Data.DataSource.ClientCollectionView](#) of the [C1.Data.Entities.EntityViewSource](#) with the specified [name](#) in the [ViewSources](#) collection.

## Syntax



Visual Basic (Declaration)	
<pre>Public Overloads ReadOnly Property Item( _     ByVal name As System.String _ ) As ClientCollectionView</pre>	
C#	
<pre>public ClientCollectionView Item(     System.string name ) {get;}</pre>	

## Parameters

*name*

The name of the [C1.Data.Entities.EntityViewSource](#) to take the [C1.Data.DataSource.ClientCollectionView](#) from.

## Property Value

The [C1.Data.DataSource.ClientCollectionView](#) of the [C1.Data.Entities.EntityViewSource](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[C1DataSource Class](#)  
[C1DataSource Members](#)  
[Overload List](#)

Item(Int32) Property

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) > [Item Property](#) : Item(Int32) Property

The index of the [C1.Data.Entities.EntityViewSource](#) to take the [C1.Data.DataSource.ClientCollectionView](#) from.

Gets the [C1.Data.DataSource.ClientCollectionView](#) of the [C1.Data.Entities.EntityViewSource](#) at the specified [index](#) in the [ViewSources](#) collection.

## Syntax

Visual Basic (Declaration)	
<pre>Public Overloads ReadOnly Property Item( _     ByVal index As System.Integer _ ) As ClientCollectionView</pre>	
C#	
<pre>public ClientCollectionView Item(     System.int index ) {get;}</pre>	

## Parameters

*index*

The index of the [C1.Data.Entities.EntityViewSource](#) to take the [C1.Data.DataSource.ClientCollectionView](#) from.

## Property Value

The [C1.Data.DataSource.ClientCollectionView](#) of the [C1.Data.Entities.EntityViewSource](#).

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[C1DataSource Class](#)  
[C1DataSource Members](#)  
[Overload List](#)

## ObjectContext Property

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : ObjectContext Property

Gets the **System.Data.Objects.ObjectContext** the [ClientCache](#) is connected to.

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.DefaultValueAttribute(&gt;&gt;</pre>	

<b>Public ReadOnly Property</b> ObjectContext <b>As</b> System.Data.Entity.Core.Objects.ObjectContext
--

C#
----

[System.ComponentModel.DefaultValue()] <b>public</b> System.Data.Entity.Core.Objects.ObjectContext ObjectContext { <b>get</b> ; }
--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### RefreshInterval Property

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : RefreshInterval Property

Gets or sets the interval between automatic [Refresh](#) operations to refresh the data with any changes that may have occurred on the server.

## Syntax

Visual Basic (Declaration)
----------------------------

<System.ComponentModel.DescriptionAttribute("Gets or sets the interval between automatic Refresh operations.")> <System.ComponentModel.CategoryAttribute("Common")> <System.ComponentModel.DefaultValueAttribute()> <b>Public Property</b> RefreshInterval <b>As</b> System.TimeSpan
--

C#
----

[System.ComponentModel.Description("Gets or sets the interval between automatic Refresh operations.")] [System.ComponentModel.Category("Common")] [System.ComponentModel.DefaultValue()] <b>public</b> System.TimeSpan RefreshInterval { <b>get</b> ; <b>set</b> ; }
--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

### ViewSources Property

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : ViewSources Property

Gets a collection of [C1.Data.Entities.EntityViewSource](#) objects that define views (based on queries) in this [C1DataSource](#).

## Syntax

Visual Basic (Declaration)	
<pre>&lt;System.ComponentModel.CategoryAttribute("Common")&gt; Public ReadOnly Property ViewSources As EntityViewSourceCollection</pre>	
C#	
<pre>[System.ComponentModel.Category("Common")] public EntityViewSourceCollection ViewSources {get;}</pre>	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)


[C1DataSource Members](#)

### Fields

[C1.WPF.Data.Entities Namespace](#) : C1DataSource Class

For a list of all members of this type, see [C1DataSource members](#).

# Public Fields

	Name	Description
 <b>S</b>	<a href="#">ControlHandlerProperty</a>	Identifies the C1.WPF.Data.Entities.C1DataSource.ControlHandler attached property.

[Top](#)

## See Also

### Reference

[C1DataSource Class](#)  
[C1.WPF.Data.Entities Namespace](#)

### ControlHandlerProperty Field

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : ControlHandlerProperty Field

Identifies the C1.WPF.Data.Entities.C1DataSource.ControlHandler attached property.

## Syntax

Visual Basic (Declaration)	
<pre>Public Shared ReadOnly ControlHandlerProperty As System.Windows.DependencyProperty</pre>	
C#	
<pre>public static readonly System.Windows.DependencyProperty ControlHandlerProperty</pre>	

## Remarks

Use this attached property to connect a [control handler](#) to a control.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

## Reference

[C1DataSource Class](#)














[C1DataSource Members](#)




















## Events




















[C1.WPF.Data.Entities Namespace](#) : C1DataSource Class

For a list of all members of this type, see [C1DataSource members](#).




















## Public Events




















	Name	Description
	<a href="#">ContextMenuClosing</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">ContextMenuOpening</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DataContextChanged</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">DragEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">DragLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">DragOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Drop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">FocusableChanged</a>	(Inherited from System.Windows.UIElement)
	<a href="#">GiveFeedback</a>	(Inherited from System.Windows.UIElement)
	<a href="#">GotFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">GotKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">GotMouseCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">GotStylusCapture</a>	(Inherited from System.Windows.UIElement)




















	GotTouchCapture	(Inherited from System.Windows.UIElement)
	Initialized	(Inherited from System.Windows.FrameworkElement)
	IsEnabledChanged	(Inherited from System.Windows.UIElement)
	IsHitTestVisibleChanged	(Inherited from System.Windows.UIElement)
	IsKeyboardFocusedChanged	(Inherited from System.Windows.UIElement)
	IsKeyboardFocusWithinChanged	(Inherited from System.Windows.UIElement)
	IsMouseCapturedChanged	(Inherited from System.Windows.UIElement)
	IsMouseCaptureWithinChanged	(Inherited from System.Windows.UIElement)
	IsMouseDirectlyOverChanged	(Inherited from System.Windows.UIElement)
	IsStylusCapturedChanged	(Inherited from System.Windows.UIElement)
	IsStylusCaptureWithinChanged	(Inherited from System.Windows.UIElement)
	IsStylusDirectlyOverChanged	(Inherited from System.Windows.UIElement)
	IsVisibleChanged	(Inherited from System.Windows.UIElement)
	KeyDown	(Inherited from System.Windows.UIElement)
	KeyUp	(Inherited from System.Windows.UIElement)
	LayoutUpdated	(Inherited from System.Windows.UIElement)
	Loaded	(Inherited from System.Windows.FrameworkElement)
	LostFocus	(Inherited from System.Windows.UIElement)
	LostKeyboardFocus	(Inherited from System.Windows.UIElement)





	<a href="#">LostMouseCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">LostStylusCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">LostTouchCapture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ManipulationBoundaryFeedback</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ManipulationCompleted</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ManipulationDelta</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ManipulationInertiaStarting</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ManipulationStarted</a>	(Inherited from System.Windows.UIElement)
	<a href="#">ManipulationStarting</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseDoubleClick</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">MouseDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseLeftButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseLeftButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseRightButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseRightButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">MouseUp</a>	(Inherited from System.Windows.UIElement)



	<a href="#">MouseWheel</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewDragEnter</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewDragLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewDragOver</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewDrop</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewGiveFeedback</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewGotKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewKeyDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewKeyUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewLostKeyboardFocus</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseDoubleClick</a>	(Inherited from System.Windows.Controls.Control)
	<a href="#">PreviewMouseDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseLeftButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseLeftButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseRightButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseRightButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewMouseWheel</a>	(Inherited from System.Windows.UIElement)

	<a href="#">PreviewQueryContinueDrag</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusButtonDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusButtonUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusInAirMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusInRange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusOutOfRange</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusSystemGesture</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewStylusUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewTextInput</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewTouchDown</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewTouchMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">PreviewTouchUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">QueryContinueDrag</a>	(Inherited from System.Windows.UIElement)
	<a href="#">QueryCursor</a>	(Inherited from System.Windows.UIElement)
	<a href="#">RequestBringIntoView</a>	(Inherited from System.Windows.FrameworkElement)
	<a href="#">SavedChanges</a>	Occurs after a save operation is completed.
	<a href="#">SavingChanges</a>	Occurs before changes are saved.

	SizeChanged	(Inherited from System.Windows.FrameworkElement)
	SourceUpdated	(Inherited from System.Windows.FrameworkElement)
	StylusButtonDown	(Inherited from System.Windows.UIElement)
	StylusButtonUp	(Inherited from System.Windows.UIElement)
	StylusDown	(Inherited from System.Windows.UIElement)
	StylusEnter	(Inherited from System.Windows.UIElement)
	StylusInAirMove	(Inherited from System.Windows.UIElement)
	StylusInRange	(Inherited from System.Windows.UIElement)
	StylusLeave	(Inherited from System.Windows.UIElement)
	StylusMove	(Inherited from System.Windows.UIElement)
	StylusOutOfRange	(Inherited from System.Windows.UIElement)
	StylusSystemGesture	(Inherited from System.Windows.UIElement)
	StylusUp	(Inherited from System.Windows.UIElement)
	TargetUpdated	(Inherited from System.Windows.FrameworkElement)
	TextInput	(Inherited from System.Windows.UIElement)
	ToolTipClosing	(Inherited from System.Windows.FrameworkElement)
	ToolTipOpening	(Inherited from System.Windows.FrameworkElement)
	TouchDown	(Inherited from System.Windows.UIElement)
	TouchEnter	(Inherited from System.Windows.UIElement)

	<a href="#">TouchLeave</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TouchMove</a>	(Inherited from System.Windows.UIElement)
	<a href="#">TouchUp</a>	(Inherited from System.Windows.UIElement)
	<a href="#">Unloaded</a>	(Inherited from System.Windows.FrameworkElement)

[Top](#)

## See Also

### Reference

[C1DataSource Class](#)

[C1.WPF.Data.Entities Namespace](#)

### SavedChanges Event

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : SavedChanges Event

Occurs after a save operation is completed.

## Syntax

Visual Basic (Declaration)	
<code>Public Event SavedChanges As System.EventHandler(Of SavedChangesEventArgs)</code>	
C#	
<code>public event System.EventHandler&lt;SavedChangesEventArgs&gt; SavedChanges</code>	

## Event Data

The event handler receives an argument of type [SavedChangesEventArgs](#) containing data related to this event. The following **SavedChangesEventArgs** properties provide information specific to this event.

Property	Description
<a href="#">Error</a>	Gets a value showing the error that occurred during a save operation.
<a href="#">HasError</a>	Gets a value indicating whether the save operation has failed. If true, inspect the <a href="#">Error</a> property for details.

<a href="#">IsErrorHandled</a>	Gets a value indicating whether the error has been marked as handled by calling <a href="#">MarkErrorAsHandled</a> .
--------------------------------	--

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

[SaveChanges Method](#)

### SavingChanges Event

[C1.WPF.Data.Entities Namespace](#) > [C1DataSource Class](#) : SavingChanges Event

Occurs before changes are saved.

## Syntax

Visual Basic (Declaration)	
<b>Public Event</b> SavingChanges <b>As</b> System.EventHandler(Of CancelEventArgs)	
C#	
<b>public event</b> System.EventHandler<CancelEventArgs> SavingChanges	

## Event Data

The event handler receives an argument of type `System.ComponentModel.CancelEventArgs` containing data related to this event. The following **CancelEventArgs** properties provide information specific to this event.

Property	Description
<b>Cancel</b>	

## Remarks

This event is raised from the [SaveChanges](#) method and allows a handler to cancel the operation before it begins. When a handler sets **System.ComponentModel.CancelEventArgs.Cancel** to True, the operation will be aborted and a subsequent SavedChanges event will not be raised.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[C1DataSource Class](#)

[C1DataSource Members](#)

[SaveChanges Method](#)

## EntityViewSourceCollection

[C1.WPF.Data.Entities Namespace](#) : EntityViewSourceCollection Class

An observable collection of [C1.Data.Entities.EntityViewSource](#) objects.

## Object Model

EntityViewSourceCollection

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>
Public Class EntityViewSourceCollection
    Inherits System.Collections.ObjectModel.ObservableCollection(Of
EntityViewSource)
```

C#

```
[System.Reflection.DefaultMember("Item")]
public class EntityViewSourceCollection :
System.Collections.ObjectModel.ObservableCollection<EntityViewSource>
```

## Inheritance Hierarchy

System.Object

System.Collections.ObjectModel.Collection<T>

System.Collections.ObjectModel.ObservableCollection<T>

**C1.WPF.Data.Entities.EntityViewSourceCollection**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityViewSourceCollection Members](#)

[C1.WPF.Data.Entities Namespace](#)

## Overview

[C1.WPF.Data.Entities Namespace](#) : EntityViewSourceCollection Class

An observable collection of [C1.Data.Entities.EntityViewSource](#) objects.

## Object Model

EntityViewSourceCollection

## Syntax

Visual Basic (Declaration)

```
<System.Reflection.DefaultMemberAttribute("Item")>
Public Class EntityViewSourceCollection
    Inherits System.Collections.ObjectModel.ObservableCollection(Of
EntityViewSource)
```

C#

```
[System.Reflection.DefaultMember("Item")]
public class EntityViewSourceCollection :
System.Collections.ObjectModel.ObservableCollection<EntityViewSource>
```

## Inheritance Hierarchy

System.Object

System.Collections.ObjectModel.Collection<T>

System.Collections.ObjectModel.ObservableCollection<T>

**C1.WPF.Data.Entities.EntityViewSourceCollection**

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityViewSourceCollection Members](#)  
[C1.WPF.Data.Entities Namespace](#)

## Members

[Properties](#) [Methods](#) [Events](#)

[C1.WPF.Data.Entities Namespace](#) : [EntityViewSourceCollection Class](#)



The following tables list the members exposed by [EntityViewSourceCollection](#).

## Public Constructors

	Name	Description
	<a href="#">EntityViewSourceCollection Constructor</a>	

[Top](#)

## Public Properties

	Name	Description
	<a href="#">Count</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;EntityViewSource&gt;</a> )
	<a href="#">Item</a>	Gets the <a href="#">C1.Data.Entities.EntityViewSource</a> with the specified <a href="#">name</a> .

[Top](#)

## Protected Properties

	Name	Description
	<a href="#">Items</a>	(Inherited from



		System.Collections.ObjectModel.Collection<EntityViewSource>)
--	--	--










[Top](#)

## Public Methods

	Name	Description
⇒	<a href="#">Add</a>	(Inherited from System.Collections.ObjectModel.Collection<EntityViewSource>)
⇒	<a href="#">Clear</a>	(Inherited from System.Collections.ObjectModel.Collection<EntityViewSource>)
⇒	<a href="#">Contains</a>	(Inherited from System.Collections.ObjectModel.Collection<EntityViewSource>)
⇒	<a href="#">CopyTo</a>	(Inherited from System.Collections.ObjectModel.Collection<EntityViewSource>)
⇒	<a href="#">GetEnumerator</a>	(Inherited from System.Collections.ObjectModel.Collection<EntityViewSource>)
⇒	<a href="#">IndexOf</a>	(Inherited from System.Collections.ObjectModel.Collection<EntityViewSource>)
⇒	<a href="#">Insert</a>	(Inherited from System.Collections.ObjectModel.Collection<EntityViewSource>)
⇒	<a href="#">Move</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<EntityViewSource>)
⇒	<a href="#">Remove</a>	(Inherited from System.Collections.ObjectModel.Collection<EntityViewSource>)
⇒	<a href="#">RemoveAt</a>	(Inherited from System.Collections.ObjectModel.Collection<EntityViewSource>)

[Top](#)


## Protected Methods

	Name	Description
	<a href="#">BlockReentrancy</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<EntityViewSource>)
	<a href="#">CheckReentrancy</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<EntityViewSource>)
	<a href="#">ClearItems</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<EntityViewSource>)
	<a href="#">InsertItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<EntityViewSource>)
	<a href="#">MoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<EntityViewSource>)
	<a href="#">OnCollectionChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<EntityViewSource>)
	<a href="#">OnPropertyChanged</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<EntityViewSource>)
	<a href="#">RemoveItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<EntityViewSource>)
	<a href="#">SetItem</a>	(Inherited from System.Collections.ObjectModel.ObservableCollection<EntityViewSource>)

		ce>)
--	--	------


[Top](#)

## Public Events

	Name	Description
	<a href="#">CollectionChange</a> d	(Inherited from System.Collections.ObjectModel.ObservableCollection<EntityViewSource e>)

[Top](#)

## Protected Events

	Name	Description
	<a href="#">PropertyChange</a> d	(Inherited from System.Collections.ObjectModel.ObservableCollection<EntityViewSource >)

[Top](#)

## See Also

### Reference

[EntityViewSourceCollection Class](#)  
[C1.WPF.Data.Entities Namespace](#)

## EntityViewSourceCollection Constructor

[C1.WPF.Data.Entities Namespace](#) > [EntityViewSourceCollection Class](#) : EntityViewSourceCollection  
 Constructor

## Syntax

Visual Basic (Declaration)	
<b>Public Function</b> <a href="#">New()</a>	
C#	
<b>public</b> EntityViewSourceCollection()	

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityViewSourceCollection Class](#)



[EntityViewSourceCollection Members](#)

## Properties

[C1.WPF.Data.Entities Namespace](#) : [EntityViewSourceCollection Class](#)


For a list of all members of this type, see [EntityViewSourceCollection members](#).

## Public Properties

	Name	Description
	<a href="#">Count</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;EntityViewSource&gt;</a> )
	<a href="#">Item</a>	Gets the <a href="#">C1.Data.Entities.EntityViewSource</a> with the specified <a href="#">name</a> .

[Top](#)

## Protected Properties

	Name	Description
	<a href="#">Items</a>	(Inherited from <a href="#">System.Collections.ObjectModel.Collection&lt;EntityViewSource&gt;</a> )

[Top](#)

## See Also

### Reference

[EntityViewSourceCollection Class](#)

[C1.WPF.Data.Entities Namespace](#)

## Item Property

[C1.WPF.Data.Entities Namespace](#) > [EntityViewSourceCollection Class](#) : Item Property

!The name of the [C1.Data.Entities.EntityViewSource](#) to get from the collection.

Gets the [C1.Data.Entities.EntityViewSource](#) with the specified [name](#).

## Syntax

Visual Basic (Declaration)	
<pre>Public Shadows ReadOnly Default Property Item( _     ByVal name As System.String _ ) As EntityViewSource</pre>	
C#	
<pre>public new EntityViewSource this[     System.string name ]; {get;}</pre>	

### Parameters

*name*

!The name of the [C1.Data.Entities.EntityViewSource](#) to get from the collection.

### Property Value

The [C1.Data.Entities.EntityViewSource](#) with the specified [name](#), or null if it does not exist.

## Requirements

**Target Platforms:** Windows 7, Windows Vista SP1 or later, Windows XP SP3, Windows Server 2008 (Server Core not supported), Windows Server 2008 R2 (Server Core supported with SP1 or later), Windows Server 2003 SP2

## See Also

### Reference

[EntityViewSourceCollection Class](#)  
[EntityViewSourceCollection Members](#)