ComponentOne

Chart3D for WinForms

ComponentOne, a division of GrapeCity
201 South Highland Avenue, Third Floor
Pittsburgh, PA 15206 USA

Website: http://www.componentone.com
Sales: sales@componentone.com
Telephone: 1.800.858.2739 or 1.412.681.4343 (Pittsburgh, PA USA Office)

Trademarks

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 $2 5 (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

Table of Contents

- 3D Chart for WinForms Overview 6
  - Help with WinForms Edition 6
- Key Features 7-8
- 3D Chart Design-Time Support 9
  - Chart3D Smart Tag 9-10
  - Chart3D Context Menu 10-11
  - Chart3D Collection Editors 11
    - BarColor Collection Editor 11-12
    - Chart3DAxis Collection Editor 12-14
    - Chart3DAxisLabel Collection Editor 14-15
    - Chart3DDataLabel Collection Editor 15-16
    - Chart3DGroup Collection Editor 16-17
    - Chart3DLabel Collection Editor 17-18
    - Chart3DPoint Collection Editor 18-19
    - Chart3DPointSeries Collection Editor 19-20
    - Chart3DStyle Collection Editor 20-21
    - Chart3D Contour Collection Editors 21
      - Chart3DContourLevel Collection Editor 21-22
      - Chart3DContourStyle Collection Editor 22-23
      - Color Collection Editor 23-24
- 3D Chart Fundamentals 25
  - Basic 3D Chart Types 25
    - 3D Surface Charts 25-26
    - 3D Bar Charts 26-27
      - Special 3D Bar Chart Properties 27-28
    - 3D Scatter Plot Charts 28-29
      - Special Scatter Plot Properties 29
        - Drop Lines 29-30
        - Scatter Plot Depth Cue 30
        - Scatter Plot Lines and Symbols 30-32
  - Meshed and Shaded Charts 32-34
    - Adding Colors to the Meshed and Shaded Charts 34
    - Creating Transparency for the Meshed and Shaded Charts 34-35
  - Contour and Zoned Charts 35-37

3D Chart Data Layouts 37
Grid Data Layout 38
Irregular Grid Data Layout 38-39
Point Data Layout 39
3D Axes 40
Axis Appearance 40
Axis Title and Rotation 40-41
Axis Tick Marks 41
Axis Grid Lines 41-42
Axis Bounds 42
Axis Scaling 42
Axes Annotation 42-43
Axes Annotation Position 43
3D Values Annotation 43
3D Value Labels Annotation 43-44
Data Labels Annotation (X and Y-axes only) 44
Distinct Methods to Label Axes 44-45
Design-Time Tools for Creating 3D Charts 46
Working with the Smart Designer 46
Primary Toolbars 46
Chart3D Toolbar 46-47
ChartArea Toolbar 47-49
Secondary Toolbars 49
Header and Footer Toolbars 49-51
Legend Toolbar 52
Working with the Chart3D Wizard 52-53
Step 1. Choose Chart Type 53
Step 2. Setup Chart 53-54
Step 3. Edit the Chart3D View 54-55
Working with the Chart3D Properties Designer 55
Gallery Tab 55-56
Data Tab 56
3D-View Tab 56-57
3D Data 58
Data Organization 58
Handling Missing Data 58

Grid and Irregular Grid Data 58-59
Changing Data Values 59
Adding Rows and Columns 59-60
Working with Point Data 60-62
Data Points 62
Loading Data from a File 62-63
Grid File Format 63
Irregular Grid Format 63
Point File Format 63-64
Charting Data from a Mathematical Calculation 64-65
Displaying 4D Data 65
Creating 4D Charts 65-68
Using Contour Data with 4D Bars Charts 68
3D Labels 69
Attaching and Positioning 3D Chart Labels 69-70
Determining which 3D Plane to Face the Label 70
3D Chart Labels Programming Considerations 70-71
Customizing 3D Chart Labels 71-72
3D Chart Label Connecting Lines and Offset 72
3D Chart Label Text and Position 72
3D Chart Label Border 72
3D Chart Label Colors 72
3D Chart Label Fonts 72
3D Chart Elements 73
3D Chart Titles 73
Title Text and Alignment 73
Title Border 73
Title Positioning 73
Title Colors 73
Title Font 73
3D Chart Legend 74
Legend Types and Orientation 74
Legend Positioning 74
Legend Title 74
Legend Border 74-75
Legend Colors 75

Legend Font 75
3D Chart Borders 75-76
3D Chart Fonts 76-77
3D Chart Colors 77
Choosing Colors Interactively 77-78
Specifying RGB Colors 78
Using Transparent Colors 78
Changing Colors 78
3D Chart Surface Appearance 78
Surface Colors 78
Zoning Method 78-79
3D Mesh Formats 79
Mesh Filtering 79
Mesh Display 79-80
Mesh Colors 80
3D Chart Elements Position and Size 80
Changing Location 80
Changing Width and Height 80
3D Contour Styles 81
Contour Style Appearance Properties 81-82
Contour Styles and Distribution Levels 82
Contour Styles Fill Colors 82-83
Changing the Contour's Line Thickness and Color 83
Displaying Contours and Zones On the Ceiling or Floor 83-84
Customizing the Distribution Table 84
Customizing the Number of Levels 84
Creating a Custom Distribution Table 84
Resetting to Linear Distribution Table 84
Distribution Table Programming Considerations 84-85
3D Chart End-User Interaction 86
Returning Coordinate Values 86
Determining Coordinate Values 86-87
Converting Data Coordinates to Pixel Coordinates 87
Converting Pixel Coordinates to Data Coordinates 87-88
Determining the Closest Data Point 88

Chart 3D for WinForms Samples 89
Chart 3D for WinForms Task-Based Help 90
Changing the Axis Label Color 90
Creating Chart Elements Using the Smart Designer 90
Add a Chart Footer 90-91
Add a Chart Header 91-92
Add a Chart Legend 92-93
Choose a 3D Chart Type 93-94
Choose a Chart sub-type 94-95
Accessing Chart3DAxis Collection Editor 95-96
Accessing ChartGroups 96-97
Modifying Chart Labels 97-98
Modifying Contour Levels 98-99
Modifying Header and Footer Titles 99-100
Modifying the Legend 100-102
Chart 3D for WinForms Frequently Asked Questions 103-104

3D Chart for WinForms Overview

Efficiently create professional looking 2D or 3D charts with **Chart for WinForms**. Using the latest technologies built into Visual Studio, **C1Chart** is fully compatible with the 2.0, 3.5 and 4.0 .NET Frameworks.. ComponentOne's Chart tools completely manages the underlying complexities of a component chart, allowing developers to concentrate on important application-specific tasks.

Chart for WinForms includes two charting controls for creating 2D and 3D charts in Microsoft Visual Studio : C1Chart and C1Chart3D. The **C1Chart** control is a two dimensional charting control that enables you create a variety of dynamic 2D charts for any type of charting application. The **C1Chart3D** control is three-dimensional charting control used to create 3D Surface, 3D Bar, 3D Scatter Plot charts, and 4D Bar and Surface charts.

Chart for WinForms features comprehensive and extensive documentation to help you get the full potential of the C1Chart and C1Chart3D controls.For your convenience, two separate help files are included with **Chart for WinForms**:

- C1Chart2D Help : Includes documentation relating to the C1Chart control.
- C1Chart3D Help : Includes documentation relating to the C1Chart3D control.

Use the 3D Chart to create 3D Surface, Contour, or Bar charts that enables end-users with the ability to easily rotate, scale, or zoom interactively. 3D Chart can display 3D data in fifteen basic surface or bar appearances and automatically generate contours and zones from the data.

Getting Started

If you are new to the **C1Chart3D** control, get started with the following topics:

- 3D Chart Fundamentals
- Design Time Tools for Creating 3D Charts
- Chart 3D for WinForms Task-Based Help

Help with WinForms Edition

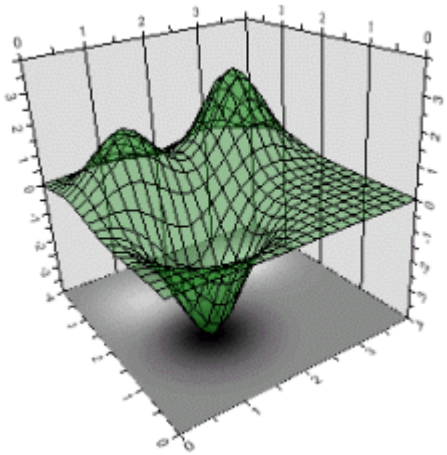
Getting Started

For information on installing **ComponentOne Studio WinForms Edition**, licensing, technical support, namespaces and creating a project with the control, please visit Getting Started with WinForms Edition.

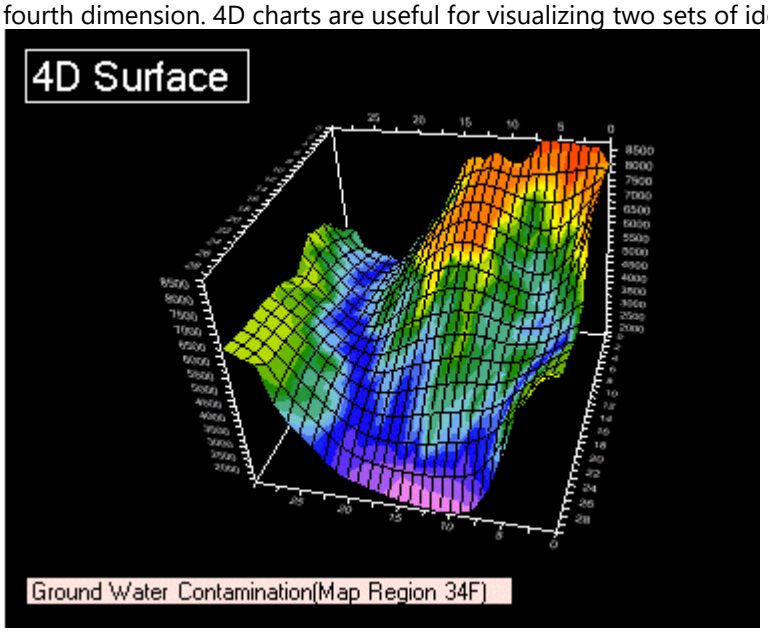
Key Features

Chart for WinForms includes the following features for the **C1Chart3D** component:

- **100% managed code**
- **Surface, Bar, and Scatter charts provide various ways to display data**
- **Automatically generates contours and zones from the data**
  The **C1Chart3D** control automatically generates contours and zones and assigns contour styles to each contour from the 100 built-in styles. For more information on contours and zones, see Contour and Zoned Charts.
- **Interactive viewing allows both the programmer and the end user to utilize chart rotation, shift, and zoom.**
- **Transparent data plotting**
  Create transparency in your data drawing using the **Transparency** property.



- **4D charts capability**
  C1Chart3D enables developers to represent four-dimensional plots and Bar charts using color as the fourth dimension. 4D charts are useful for visualizing two sets of identically sized data in one chart.



  For more information on how to create 4D charts, see Creating 4D Charts.
- **Advanced mouse tracking capabilities to keep track of the region, series, or data point under the mouse pointer**
  Provides a set of conversion methods that when used in conjunction with .NET's **MouseMove** event allow the programmer to keep track of the chart's region, series, or data point under the mouse pointer. This

makes it easier to create interesting application specific features like handling a double-click in the legend, or chart tool tips.

- **Image generation: Charts can be saved to any number of image formats (metafile, BMP, JPG, and more).**
- **SmartDesigner provides highly-interactive chart building capabilities**
  Save substantial time using Chart's SmartDesigner, which handles everyday tasks in chart placement. Accomplish tasks without leaving the design form; each chart element reveals built-in toolbars and editors with the click of your mouse pointer.
  For more information on the **SmartDesigner**, see Working with the Smart Designer.
- **Novice users can create a 3D chart in three simple steps with the Chart3D Wizard**
  The **Chart3D Wizard** walks beginners through the steps of creating a new 3D chart from start to finish: choose the chart type; select the type of data layout: regular grid, irregular grid, or point layout; and edit the chart's 3D view such as rotate the x-axis, change the chart's perspective, or sclae it.
  For more information on the Chart3D Wizard, see Working with the Chart3D Wizard.
- **You no longer have to tirelessly scroll through the Propertes window to create a chart**
  C1Chart3D places the chart elements in an organized Chart Properties designer so you can quickly address chart details. Create or modify existing charts, select the type of data layout: regular grid, irregular grid, or point layout; and edit the chart's 3D view such as rotate the x-axis, change the chart's perspective, or sclae it.
  For more information on the **Chart3D Properties** designer, see Working with the Chart3D Properties designer.
- **Invert axes using one property**
  Enables you to invert the X and Y axis using one simple property.
- **Highly interactive behavior at run time drives up value in chart use**
  C1Chart provides interactive built-in tools for rotation, scaling, and zooming. Using these tools, you can build highly-interactive charts for your users.
- **C1Chart provides flexible image formats for chart rendering**
  Charts can be saved to any number of image formats (metafile, BMP, JPG, and more).

3D Chart Design-Time Support

C1Chart3D provides visual designers that offer rich design-time support and simplify working with the object model. You have complete control to create a powerful and enhancing chart by using one or more of the following visual designers:

Invoking the Smart Tag

You can easily set common properties for the **C1Chart3D** control using its smart tag. For more information about the smart tag in **C1Chart**, see C1Chart3D Smart Tag.

Invoking the Chart3D Wizard

You can easily set up a chart using the built-in chart wizard. For more information on using the Chart Wizard, see Working with the Chart3D Wizard.

Invoking the Chart Properties Editor

The **Chart3D Properties** editor provides an easy and interactive way to create and modify a new or existing chart. The **Chart3D Properties** editor also provides more options to address specific details with the design of the chart you are developing. For more information on the **Chart3D Properties** editor in **C1Chart**, see Working with the Chart3D Properties Designer.

Invoking the Context Menus

You can easily configure the **C1Chart3D** control at design time by using its associated context menu. For more information on **C1Chart3D** context menu, see the C1Chart3D Context Menu.

Invoking the Collection Editors

C1Chart3D provides collection editors which conveniently allows the user to edit Chart Area elements such as Axes, ChartGroups,ChartStyles, ColumnLabels, ContourStyles, data labels, and Contour levels.

For more information on C1Chart3D's design-time editors, see C1Chart3D Collection Editors.

Showing the C1Chart3D Control's Properties

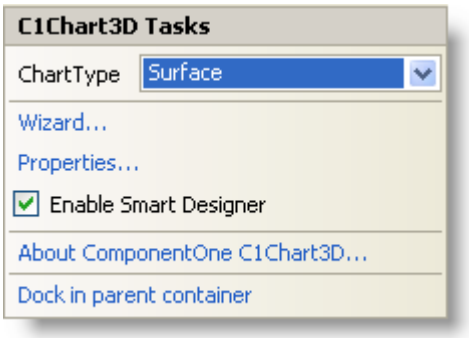
You can access the properties for the **C1Chart3D** control simply by right-clicking on the control and selecting **Properties** or by selecting the class from the drop-down box of the Properties window.

Chart3D Smart Tag

In Visual Studio 2005, the **C1Chart3D** control includes a smart tag. A smart tag represents a short-cut tasks menu that provides the most commonly used properties in each component/command.

The C1Chart3D control provides quick and easy access to the designers and common properties through its smart tag.

To access the **C1Chart3D Tasks** menu, click on the smart tag (▶) in the upper-right corner of the **C1Chart3D** control. This will open the **C1Chart3D Tasks** menu.



The **C1Chart3D Tasks** menu operates as follows:

ChartType

Clicking on the **ChartType**'s drop-down list box displays a list of available Chart3D chart types to choose from.

Wizard

Clicking on the **Wizard** item opens the **Chart3D Wizard** designer. For more information about the elements in the **Chart3D Wizard** dialog box and how to use them, see Working with the Chart3D Wizard.

Properties

Clicking on the **Properties** item opens the **Chart3D Properties** designer. For more information about the elements in the **Chart3D Properties** editor and how to use them, see Working with the Chart3D Properties Designer.

Enable Smart Designer

Selecting the **Enable Smart Designer** check box sets the **Enabled** property to **True**, and enables the Smart Designer of the **C1Chart3D** control. The default value is **True** (checked). For more information about the Smart Designer's elements see, Working with the Smart Designer.

About C1Chart3D

Clicking on the **About** item displays the **About C1Chart3D** dialog box, which is helpful in finding the version number of **C1Chart3D** and online resources.

Dock in parent container

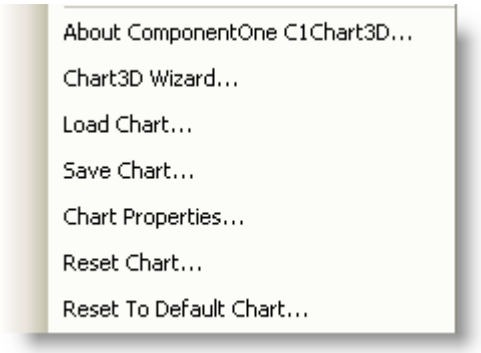
Clicking on the **Dock in parent container** link docks the **C1Chart3D** control inside its parent container.

Chart3D Context Menu

C1Chart3D provides a context menu for additional functionality to use at design time.

To access C1Chart3D's context menu:

Right-click on the **C1Chart3D** control to open its context menu.



The **C1Chart3D** context menu operates as follows:

About C1Chart3D

Displays the **About C1Chart3D** dialog box, which is helpful in finding the version number of **C1Chart3D** and online resources.

Chart3D Wizard

Opens the Chart3D Wizard.

Load Chart

Loads the saved layout of the **C1Chart3D** control.

Save Chart

Saves the layout of the **C1Chart3D** control as an XML file.

Chart Properties

Opens the **Chart3D Properties** designer.

Reset Chart

Resets the Chart

Reset To Default Chart

Resets the Chart back to its default settings.

Chart3D Collection Editors

C1Chart3D includes the following editors:

BarColor Collection Editor

The **BarColor Collection Editor** allows the user to add colors to the rows and columns for the bars.

To access the BarColor Collection Editor:

1. Right-click on the **C1Chart3D** control and select **Properties** from its context menu.
2. In the Properties window, expand the **ChartGroups** node, then expand **Group0**, and expand the **Bar** node and click on the **ellipsis** button next to the **Colors** property. The **BarColor Collection Editor** opens.

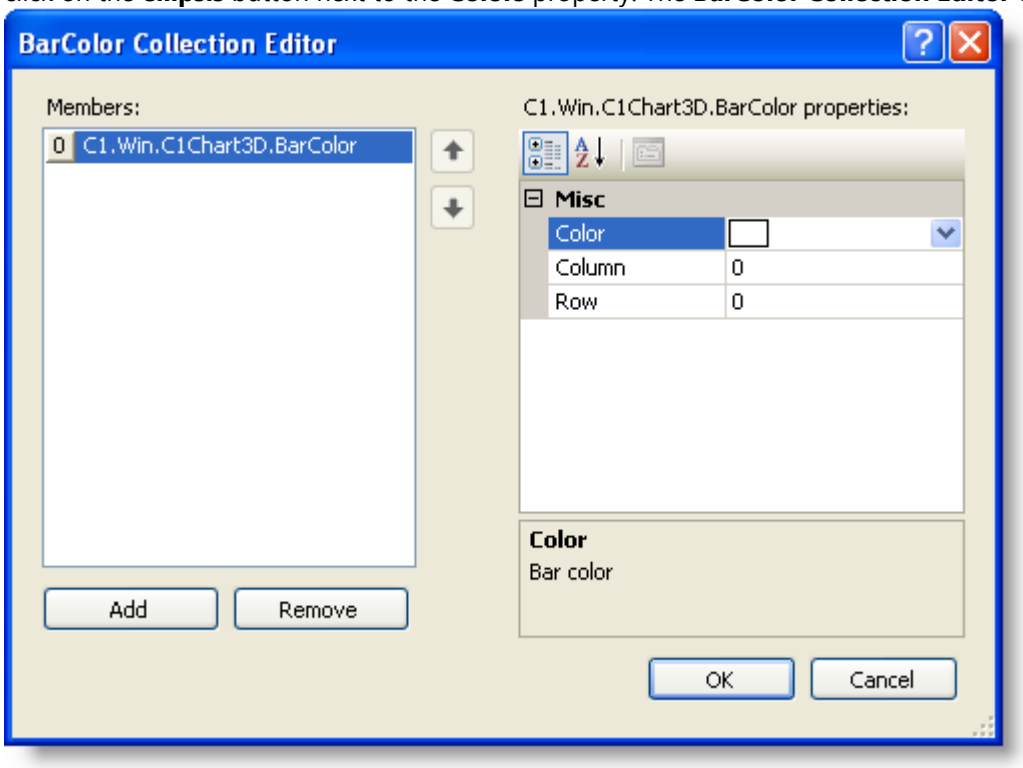
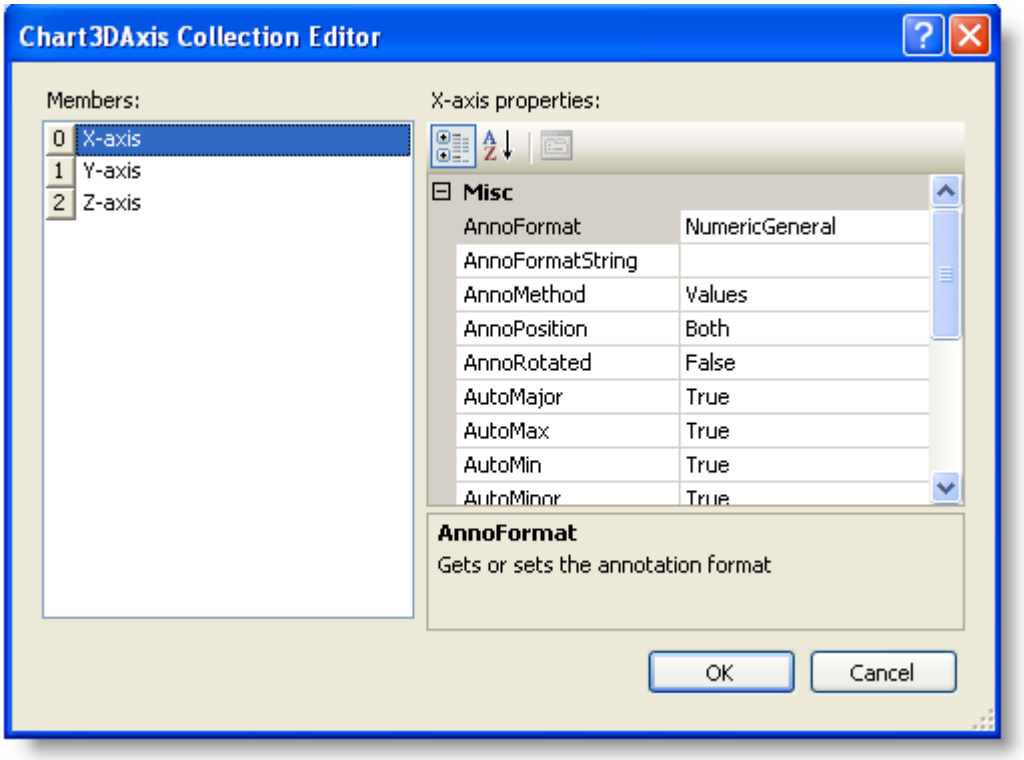


Chart3DAxis Collection Editor

The **Chart3DAxis Collection Editor** is used for modifying the properties for the X, Y, and Z axes in **C1Chart3D**.

To access the Chart3DAxis Collection Editor:

1. Right-click on the **C1Chart3D** control and select **Properties** from its context menu.
2. In the Properties window, expand the **ChartArea** node, then click on the **ellipsis** button next to the **AxesCollection** object. The **Chart3DAxis Collection Editor** opens.



Within the **Chart3DAxis Collection Editor** a **Chart3DAxis Label Collection Editor** exists. For more information on the **Chart3DAxis Label Collection Editor** see, Chart3DAxisLabel Collection Editor.

Properties available in the Chart3DAxis Collection Editor

The following properties are available for the user in the **Chart3DAxis Collection Editor** at design time or they can be used in the Chart3DAxis class at run-time:

Members	Description
AnnoFormat	Gets or sets the annotation format.
AnnoFormatString	Gets or set the annotation format string used with manual formats.
AnnoMethod	Gets or sets annotation method of the axis.
AnnoPosition	Gets or sets the position of the axis annotation and text.
AnnoRotated	Gets or sets whether annotation rotated.
AutoMajor	Gets or sets whether major tick mark values are calculated automatically.
AutoMax	Gets or sets whether the axis minimum value is calculated automatically.
AutoMin	Gets or sets whether the axis minimum value is calculated automatically.
AutoMinor	Gets or sets whether minor tick mark values are calculated automatically.

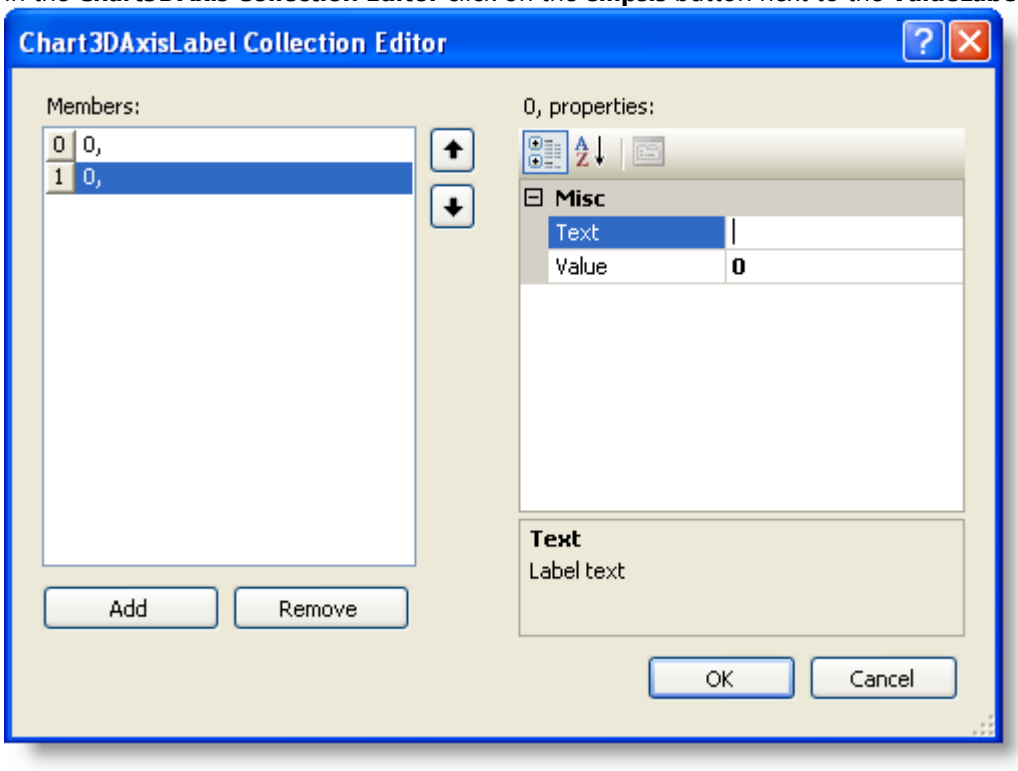
MajorGrid	Gets the major grid lines object.
Chart3DLineStyle	Grid line style.
Max	Gets or sets the maximum value of the axis.
Min	Gets or sets the minimum value of the axis.
Title	Gets or sets the title of the axis.
UnitMajor	Gets or sets the units between major tickmarks.
UnitMinor	Gets or sets the units between minor tickmarks.
ValueLabels	Gets the axis value labels.
Visible	Gets or sets the axis visibility.

Chart3DAxisLabel Collection Editor

The **Chart3DAxisLabel Collection Editor** is used for adding or removing the label's text for specific axes labels in **C1Chart3D**.

To access the Chart3DAxis Collection Editor:

1. Right-click on the **C1Chart3D** control and select **Properties** from its context menu.
2. In the Properties window, expand the **ChartArea** node, then click on the **ellipsis** button next to the **AxesCollection** property. The **Chart3DAxis Collection Editor** opens.
3. In the **Chart3DAxis Collection Editor** click on the **ellipsis** button next to the **ValueLabels** object.



The **Chart3DaxisLabel Collection Editor** appears.

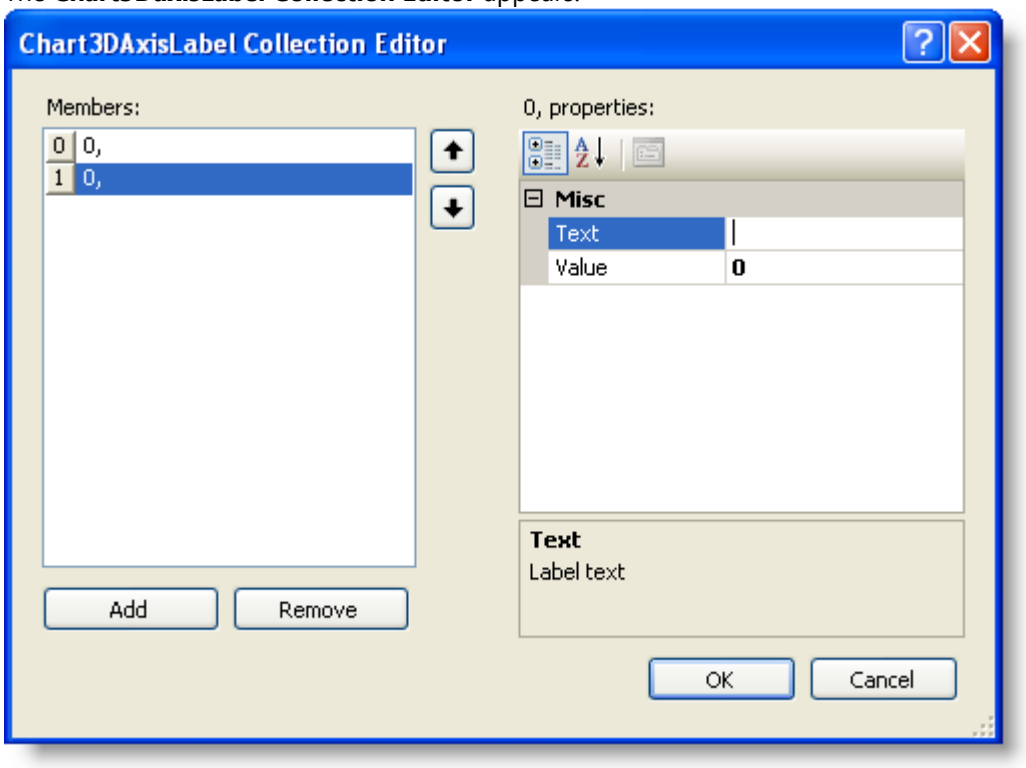


Chart3DDataLabel Collection Editor

The **Chart3DDataLabel Collection Editor** allows the user to add or remove data labels and also apply text for each label.

To access the Chart3DDataLabel Collection Editor:

1. Right-click on the **C1Chart3D** control and select **Properties** from its context menu.
2. In the Properties window, expand the **ChartGroups** node, then click on the **ellipsis** button next to the **ChartLabels** property. The **Chart3DDataLabel Collection Editor** opens.

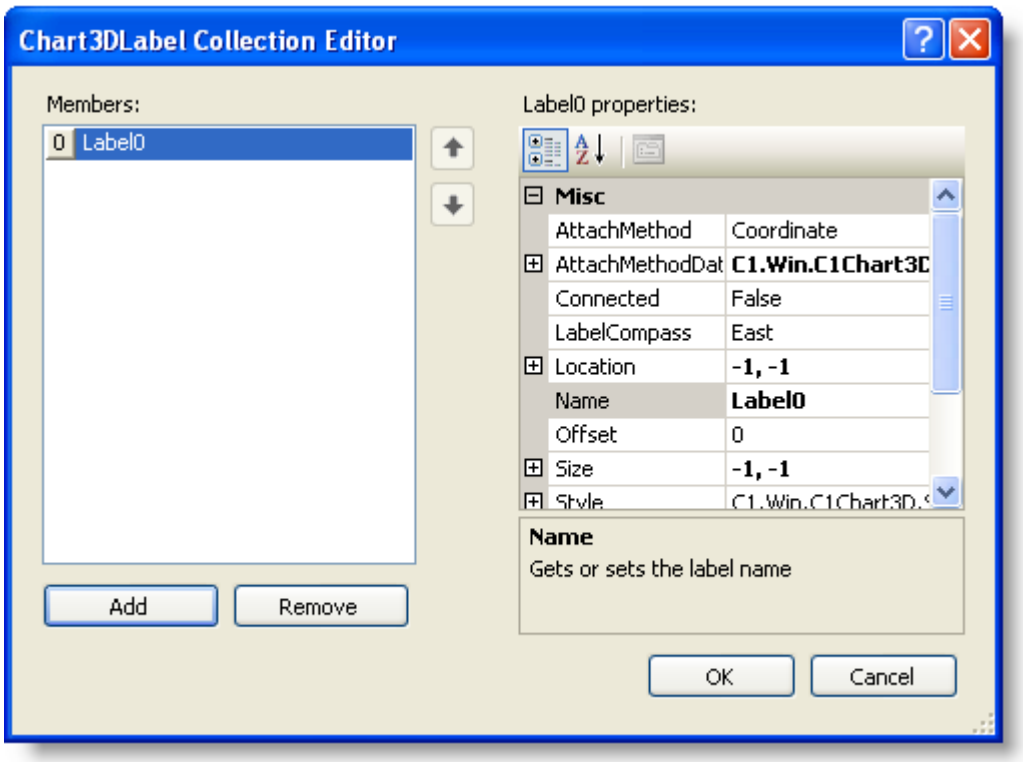
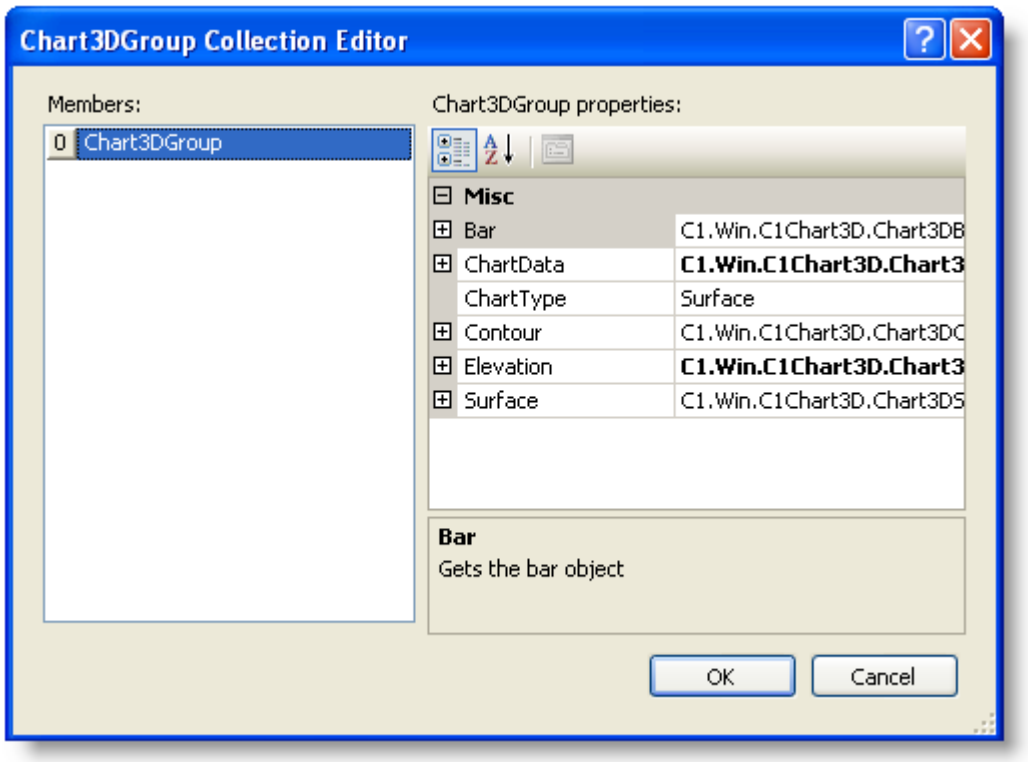


Chart3DGroup Collection Editor

The **Chart3DGroup Collection Editor** allows the user to set the Bar, ChartData, ChartType, Contour, Elevation, and Surface for Group0's properties.

To access the Chart3DGroup Collection Editor:

1. Right-click on the **C1Chart3D** control and select **Properties** from its context menu.
2. In the Properties window, expand the **ChartGroups** node, then click on the **ellipsis** button next to the **ChartGroupsCollection** property. The **Chart3DGroup Collection Editor** opens.



Within the **Chart3DGroup Collection Editor** is a **Bar Collection Editor** and **Chart3D Contour Level Collection Editor**.

Chart3DLabel Collection Editor

The **Chart3DLabel Collection Editor** allows the user to add labels by clicking on the **Add** button and attach them to the chart by coordinate, data coordinate, or data index or to remove the labels by clicking on the **Remove** button. The user can also modify the label's properties.

To access the Chart3DLabel Collection Editor:

1. Right-click on the **C1Chart3D** control and select **Properties** from its context menu.
2. In the Properties window, expand the **ChartLabels** node, then click on the **ellipsis** button next to the **LabelsCollection** property. The **Chart3DLabel Collection Editor** opens.

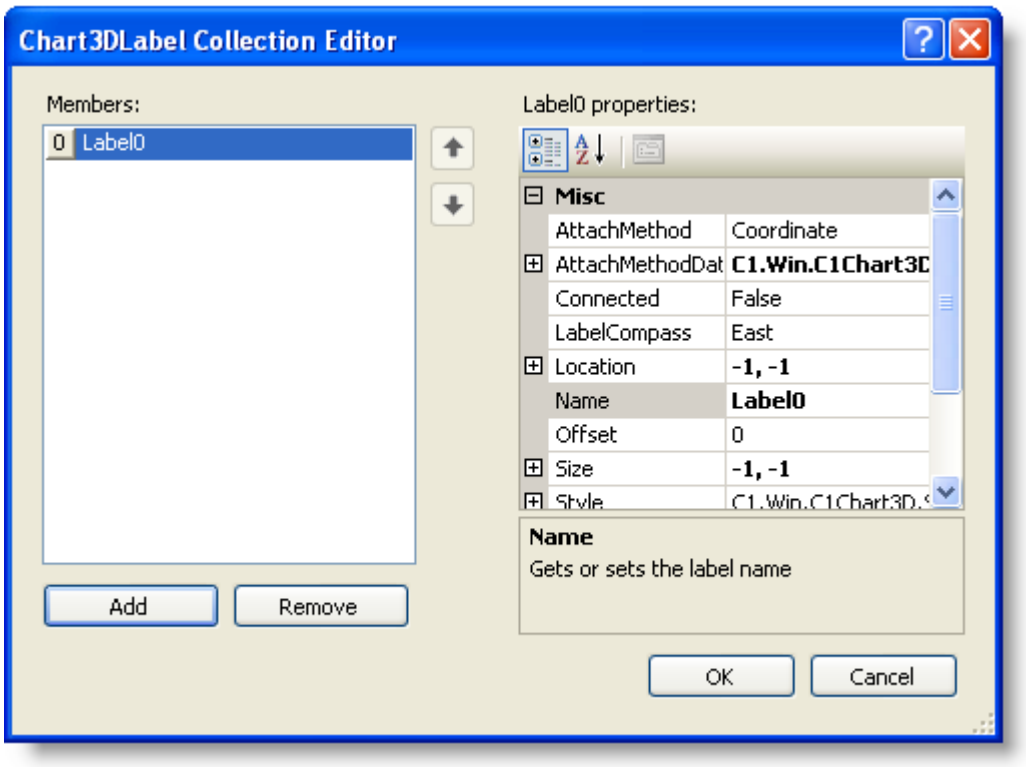
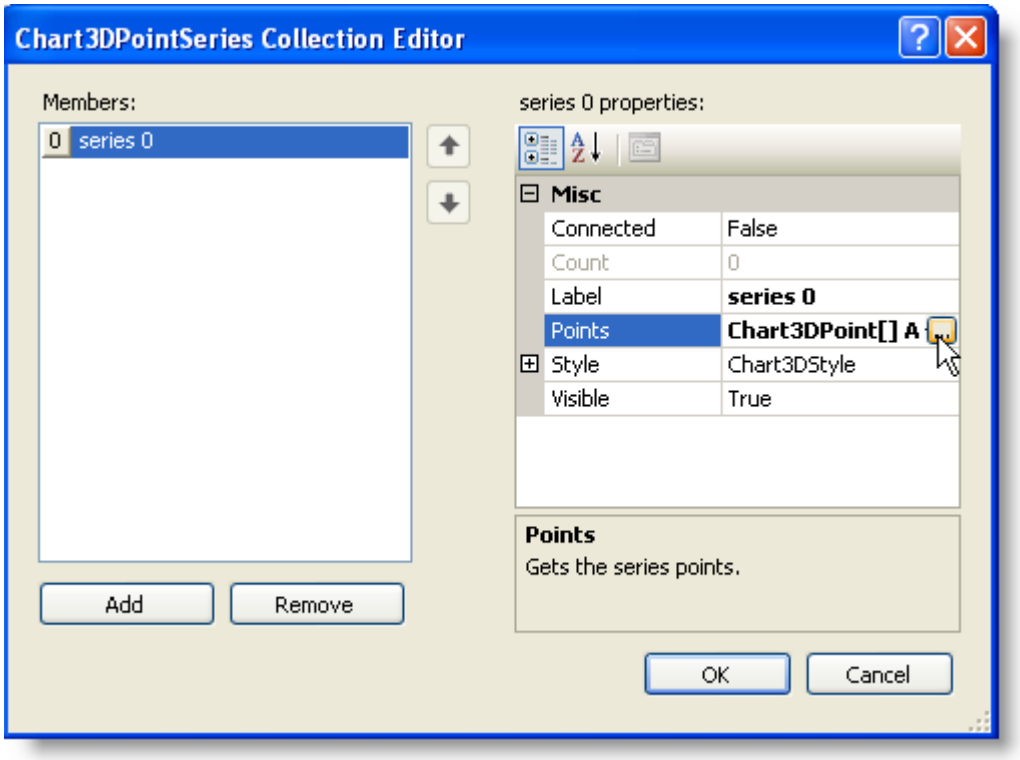


Chart3DPoint Collection Editor

The **Chart3DPoint Collection Editor** allows the user to add or remove points for points in 3D space.

To access the Chart3DPoint Collection Editor:

1. Right-click on the **C1Chart3D** control and select **Properties** from its context menu.
2. In the Properties window, expand the **ChartGroups** node, then click on the **ellipsis** button next to the **ChartGroupsCollection** property. The **Chart3DGroup Collection Editor** opens.
3. In the **Chart3DGroup Collection Editor**, expand the **ChartData** node, then set the Layout property to PointData.
4. Expand the **Set** node, then click on the **ellipsis** button next to the **SeriesCollection** property. The **Chart3DPointSeries Collection Editor** opens.
5. Click **Add**, to add a series to the point data layout. The properties appear for the Chart3DPointSeries Collection.
6. Click on the **ellipsis** button next to the **Points** property.



The **Chart3DPoint Collection Editor** appears.

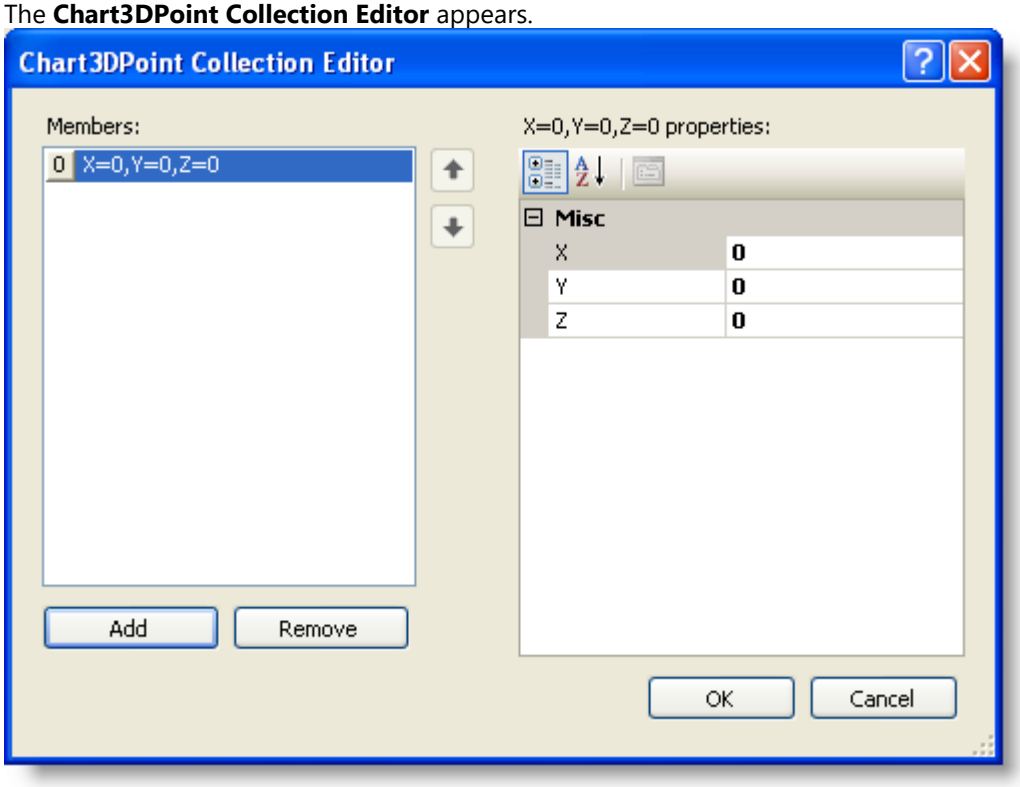


Chart3DPointSeries Collection Editor

The **Chart3DPointSeries Collection Editor** allows the user to add or remove series for point data layout (Chart3DdataSetPoint class).

To access the Chart3DPointSeries Collection Editor:

1. Right-click on the **C1Chart3D** control and select **Properties** from its context menu.
2. In the Properties window, expand the **ChartGroups** node, then click on the **ellipsis** button next to the **ChartGroupsCollection** property. The **Chart3DGroup Collection Editor** opens.
3. In the **Chart3DGroup Collection Editor**, expand the **ChartData** node, then set the Layout property to PointData.
4. Expand the **Set** node, then click on the **ellipsis** button next to the **SeriesCollection** property. The **Chart3DPointSeries Collection Editor** opens.

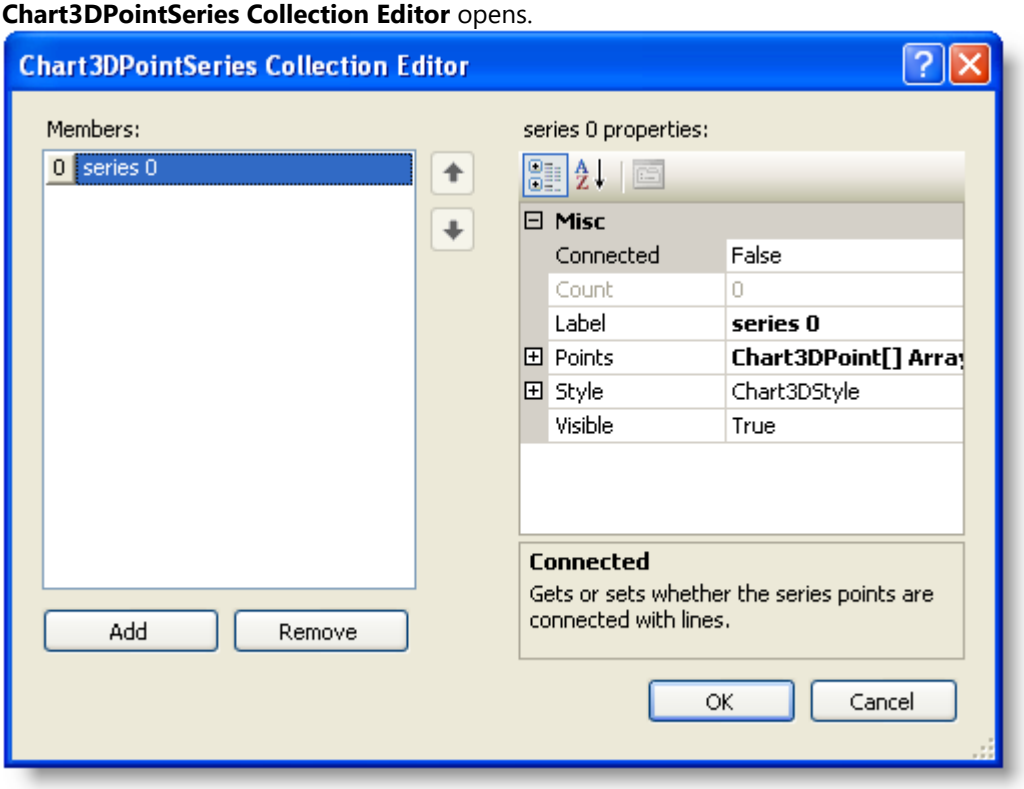


Chart3DStyle Collection Editor

The **Chart3DStyle Collection Editor** allows the user to apply line styles and symbol styles to the 3D Chart.

To access the Chart3DGroup Collection Editor:

1. Right-click on the **C1Chart3D** control and select **Properties** from its context menu.
2. In the Properties window, expand the **ChartGroups** node, then click on the **ellipsis** button next to the **ChartStyles** property.
   The **Chart3DStyle Collection Editor** opens.

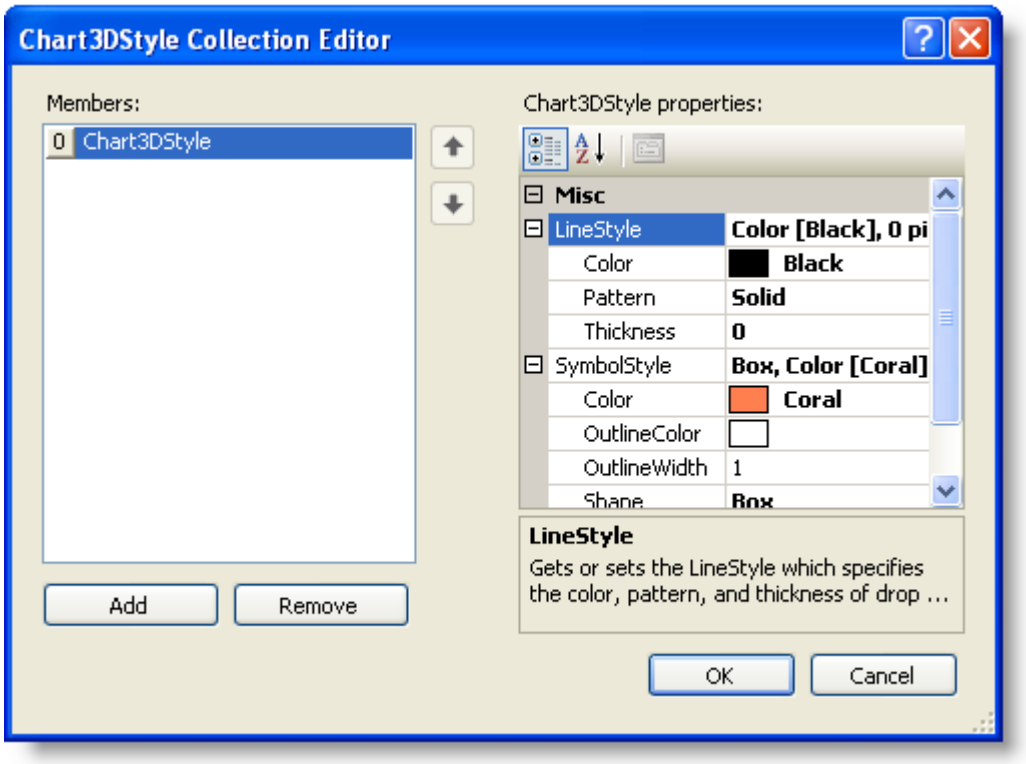


Chart3D Contour Collection Editors

C1Chart3D has several editors for adding and modifying the contour's style and level. **C1Chart3D** provides the following contour collection editors:

- **Chart3DContourLevel Collection Editor** : This editor is used to modify each level's style such as their fill color and line style.
- **Chart3DContourStyle Collection Editor** : This editor is used for adding or removing line styles for drawing contours.
- **Color Collection Editor** : This editor is used to add or remove colors in the collection.

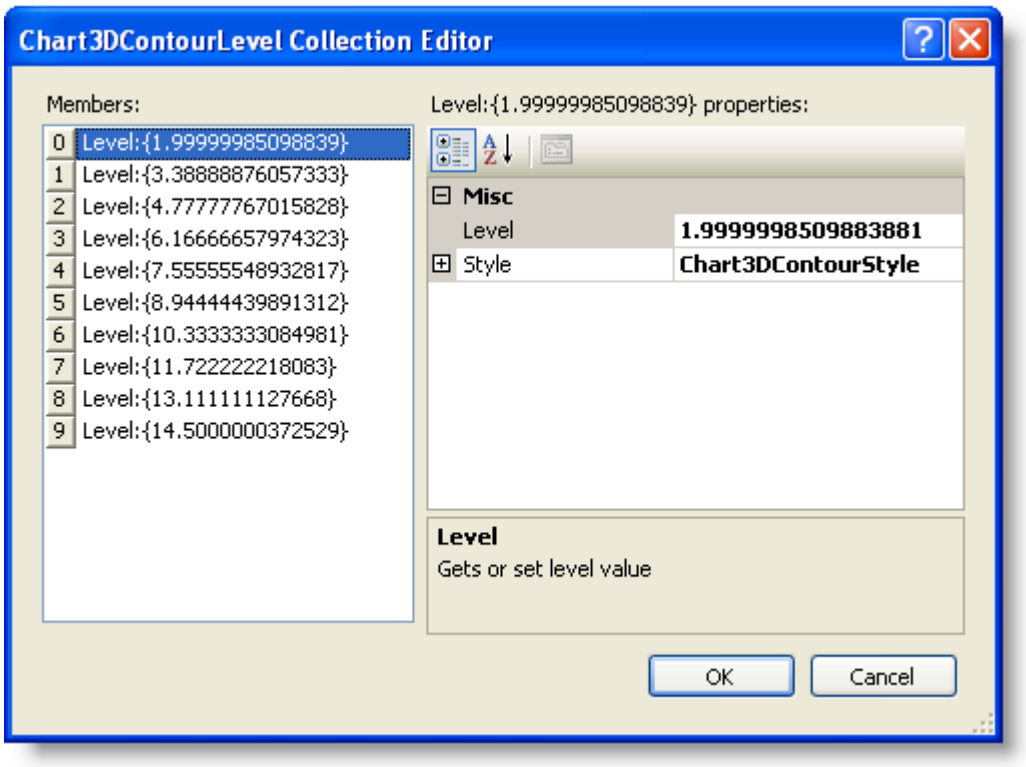
The following section briefly introduces each **Contour Collection Editor** and explains how to access them.

Chart3DContourLevel Collection Editor

The **Chart3DContourLevel Collection Editor** is used for editing each contour level's style such as their fill color and line style.

To access the Chart3DContourLevel Collection Editor:

1. Right-click on the **C1Chart3D** control and select **Properties** from its context menu.
2. In the Properties window, expand the **ChartGroups** node, expand **Group0**, and then expand the **Contours** node and click on the **ellipsis** button next to the Levels property. The **Chart3DContourLevel Collection Editor** opens.



Properties available in the Chart3DContourLevel Collection Editor

The following properties are available for the user in the **Chart3D ContourLevel Collection Editor** at design time or they can be used in the Chart3DContourLevel class at run-time:

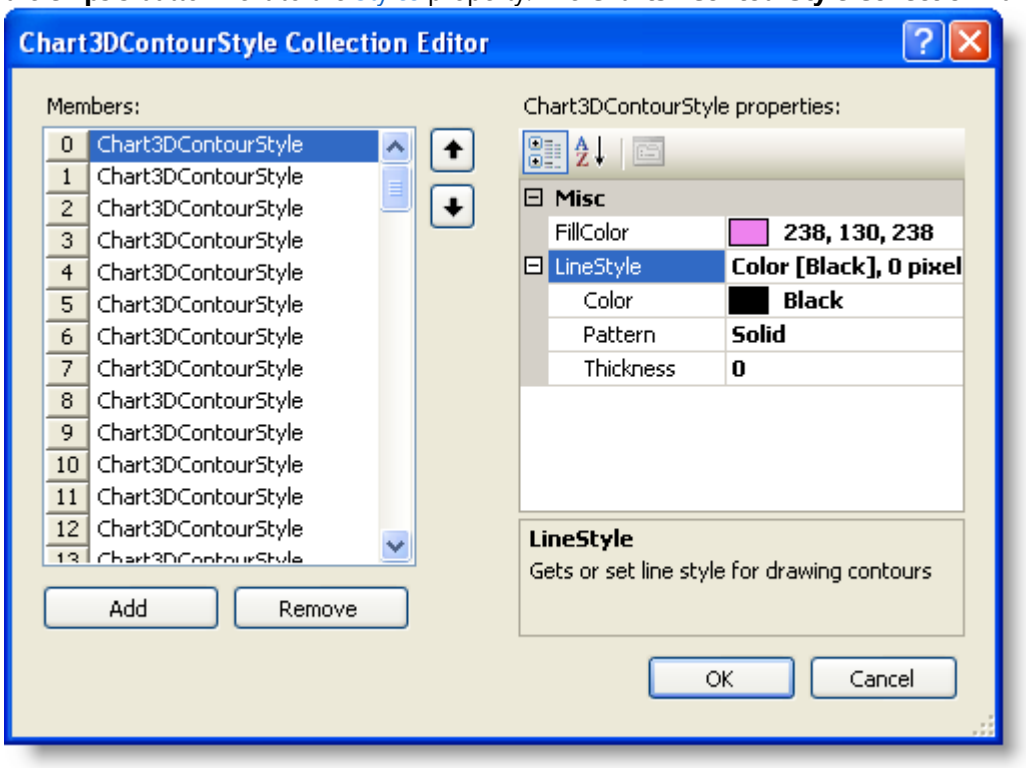
Members	Description
Level	Gets or sets the level value.
FillColor	Gets or sets zone fill color.
LineStyle	Gets or sets the line style for drawing contours.
Color	Gets or sets the color of a plotted line.
Pattern	Gets or sets the pattern of a plotted line.
Thickness	Gets or sets the thickness of a plotted line.

Chart3DContourStyle Collection Editor

The **Chart3DContourStyle Collection Editor** is used for editing each contour level's style such as their fill color and line style.

To access the Chart3DContourStyle Collection Editor:

1. Right-click on the **C1Chart3D** control and select **Properties** from its context menu.
2. In the Properties window, expand the **ChartGroups** node, then expand the **ContourStyles** node and click on the **ellipsis** button next to the Styles property. The **Chart3DContourStyle Collection Editor** opens.



Properties available in the Chart3DContourStyle Collection Editor

The following properties are available for the user in the **Chart3DContourStyle Collection Editor** at design time or they can be used in the Chart3DContourStyles class at run time:

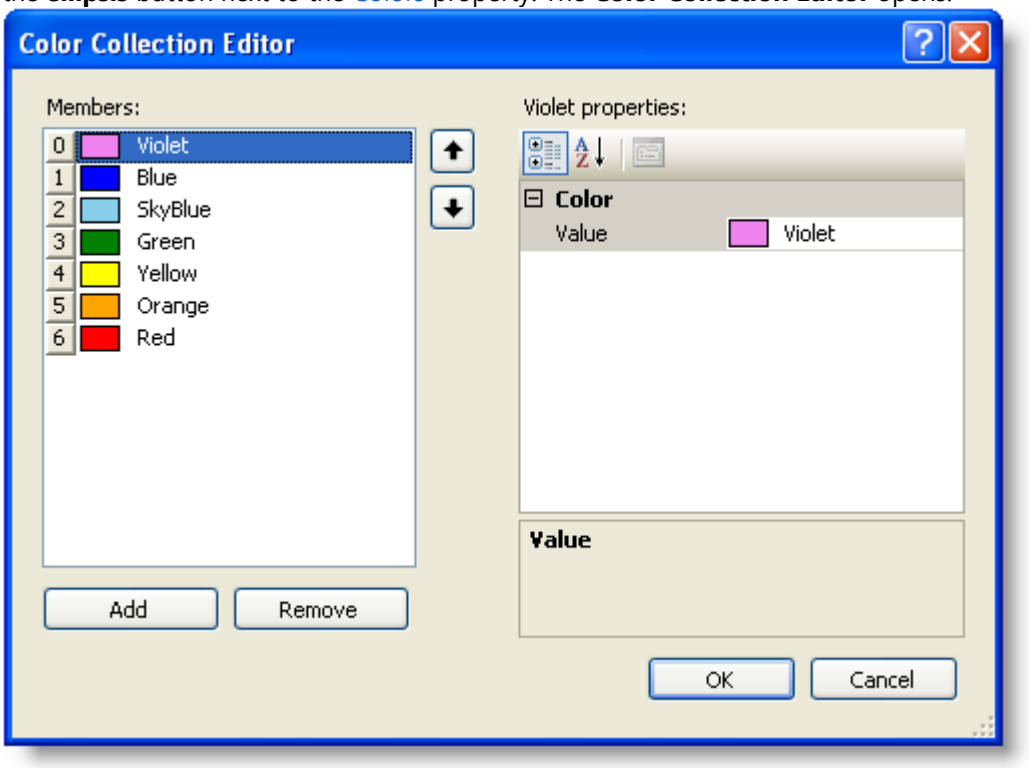
Members	Description
FillColor	Gets or sets zone fill color.
LineStyle	Gets or sets the line style for drawing contours.
Color	Gets or sets the color of a plotted line.
Pattern	Gets or sets the pattern of a plotted line.
Thickness	Gets or sets the thickness of a plotted line.

Color Collection Editor

The **Color Collection Editor** is used for adding or removing colors in the collection.

To access the Color Collection Editor:

1. Right-click on the **C1Chart3D** control and select **Properties** from its context menu.
2. In the Properties window, expand the **ChartGroups** node, then expand the **ContourStyles** node and click on the **ellipsis** button next to the Colors property. The **Color Collection Editor** opens.

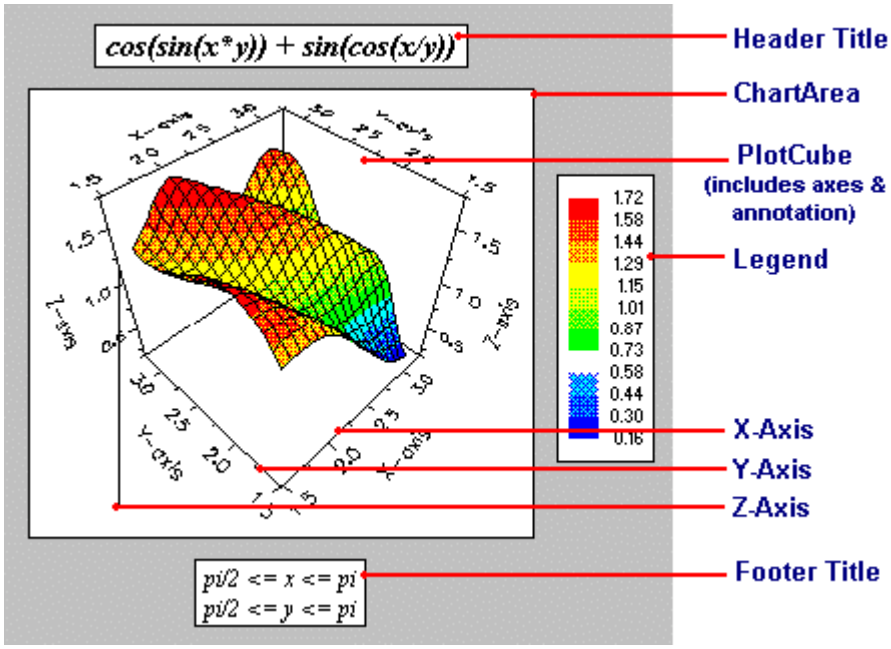


3D Chart Fundamentals

Successful charting requires familiarity with basic chart processes and vocabulary, specifically as they relate to the 3D Chart control.

3D Chart Terminology

The following image shows the terms used to describe chart elements:



The following topics cover basic information that anyone who uses the 3D Chart should be familiar with.

Basic 3D Chart Types

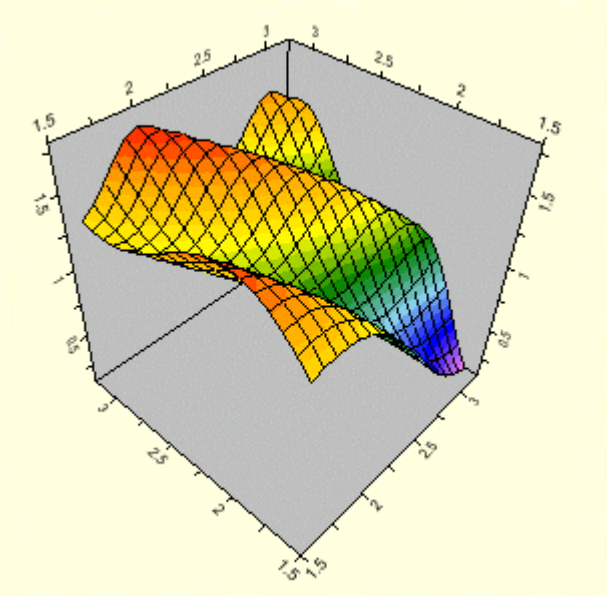
The 3D Chart displays data as a **3D Surface** chart, a **3D Bar** chart, or **3D Scatter Plot**. Use the ChartType property to specify whether to display data as a Surface chart, Bar chart, or Scatter Plot. The ChartType property is located in the **ChartGroup Collection Editor**, which can be accessed through the Chart3DGroups property in the .NET Properties window.

The following topics introduce each chart type.

3D Surface Charts

The 3D Surface charts display the data as three dimensional shaded or meshed surface with a Z-axis. They are based on X, Y, and Z axes, with more variation in the Z variable than the X or Y variables. The shaded or meshed surface is created from the information collected on how the data points are connected. The surface between data points can be estimated through interpolation.

Chart3DTypeEnum.Surface



To set the 3D chart type to Surface at design time:

1. Expand the **ChartGroups** node in the Properties window, then expand **Group0**.
2. Locate the **ChartType** property and select **Surface**.

To programmatically set the 3D chart type to Surface:

To write code in Visual Basic

```
Visual Basic
C1Chart3D1.ChartGroups(0).ChartType = Chart3DTypeEnum.Surface
```

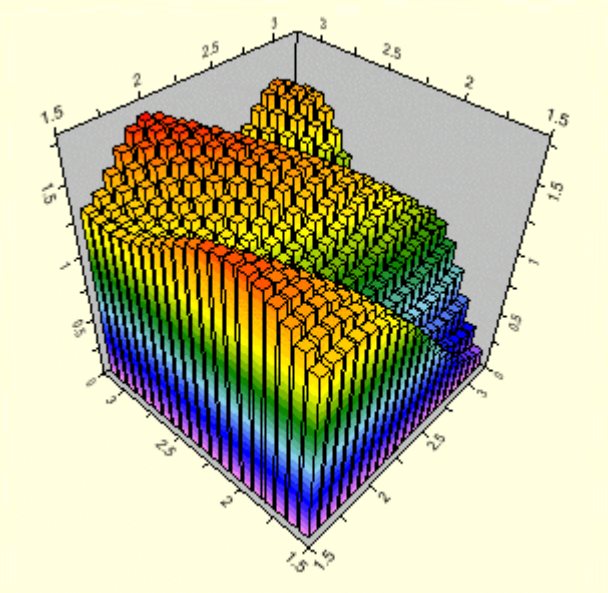
To write code in C#

```
C#
C1Chart3D1.ChartGroups[0].ChartType = Chart3DTypeEnum.Surface;
```

3D Bar Charts

The 3D Bar chart displays each data point as a single bar drawn from the Z-value to the origin. Spacing between the adjacent bars can be added when using either grid or irregular grid data layout. Like the 2D Bar chart it is useful for comparing individual items or groups of items.

Chart3DTypeEnum.Bar



To set the 3D chart type to Bar at design time:

1. Expand the **ChartGroups** node in the Properties window, then expand **Group0**.
2. Locate the **ChartType** property and select **Bar**.

To programmatically set the 3D chart type to Bar:

To write code in Visual Basic

```
Visual Basic
C1Chart3D1.ChartGroups(0).ChartType = Chart3DTypeEnum.Bar
```

To write code in C#

```
C#
C1Chart3D1.ChartGroups[0].ChartType = Chart3DTypeEnum.Bar;
```

Special 3D Bar Chart Properties

In a 3D Bar chart, each data point is displayed as a single bar drawn from the Z-value to the origin. Spacing between adjacent bars is honored when using either grid or irregular grid data layout. You can customize the origin, spacing, and colors for the 3D Bar charts.

Bar Z Origin

Use the axis Origin property to set the origin of the Z-axis. It is only used with Bar charts. **Origin** is located in the Bar node of the **Chart3DGroup Collection Editor**.

Bar Width

Use the bar RowWidth and ColumnWidth properties to set the space used by each bar. The value represents the

percentage of available space, with valid values between 0 and 100. When 100, the bars touch one another. These properties are located in the Bar node of **Chart3DGroup Collection Editor**.

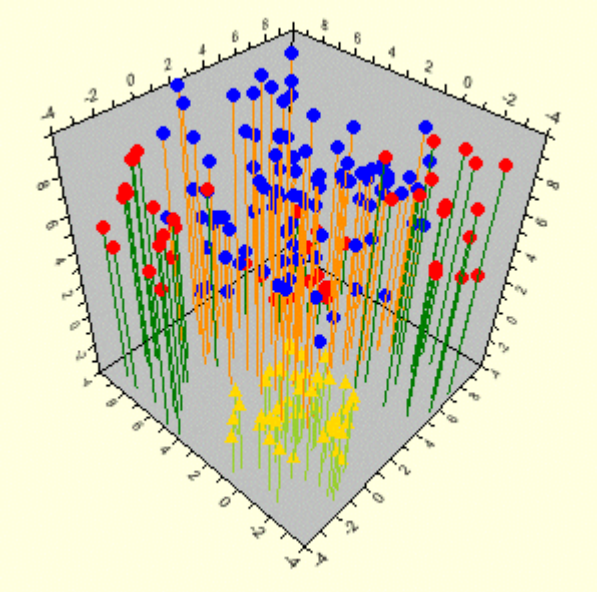
Bar Shading Colors

When a Bar chart is shaded but not zoned, bars drawn above the Z-origin use the surface top color and bars drawn below the Z-origin use the surface bottom color. The color of one bar or entire rows and columns of bars can be arbitrarily set. Create a 4D chart to define colors that appear in the Legend; see Displaying 4D Data for more information.

3D Scatter Plot Charts

The 3D Scatter plots consist of one or more series of individual points. They show a relationship between three or more variables which represent the X, Y, and one or more Z coordinates of each point. To create a better display of the 3D Scatter plot, you can add drop lines between the plotted points and the origin like shown in the following image:

Chart3DTypeEnum.Scatter



To set the 3D chart type to Scatter at design time:

1. Expand the **ChartGroups** node in the Properties window, then expand **Group0**.
2. Locate the **ChartType** property and select **Scatter**.

To programmatically set the 3D chart type to Scatter:

To write code in Visual Basic

Visual Basic
`C1Chart3D1.ChartGroups(0).ChartType = Chart3DTypeEnum.Scatter`

To write code in C#

C#

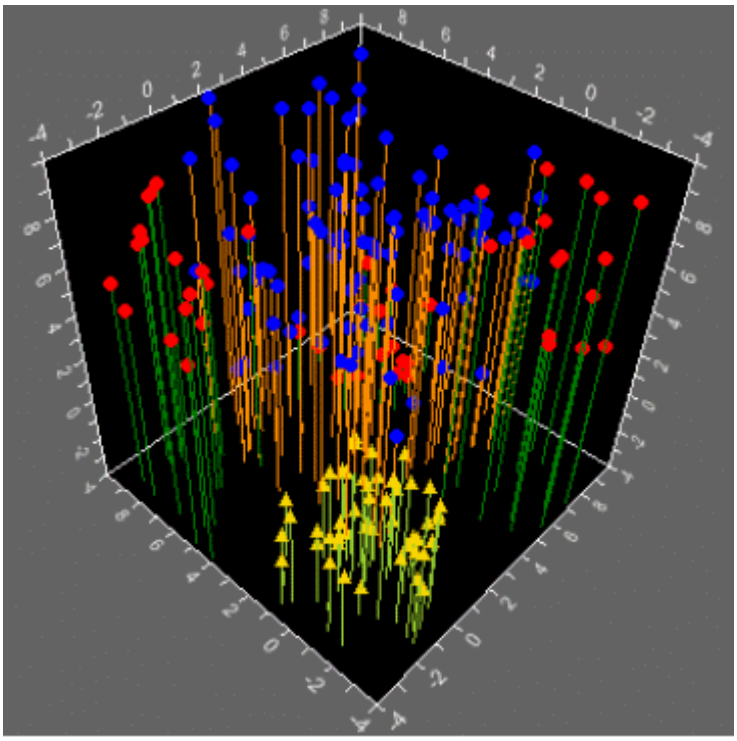
```
C1Chart3D1.ChartGroups[0].ChartType = Chart3DTypeEnum.Scatter;
```

Special Scatter Plot Properties

A scatter plot consists of one or more series of individual points. Customize the presence of drop lines as well as the appearance of the symbol and drop lines for each series. The following topics provide information on the unique properties for scatter plots that are used for creating drop lines, depth, and plot lines for the 3D Scatter Plot charts:

Drop Lines

Use the Chart3DElevation object's DropLines property to specify whether or not drop lines appear between the plotted points and the origin. Each point in a scatter plot has a drop line that connects it to the Z = Zmin plane. The DropLines property can be accessed at design time under the Elevation node of **Chart3DGroup Collection Editor**.



The following example specifies that drop lines appear:

To write code in Visual Basic

Visual Basic

```
C1Chart3D1.ChartGroups(0).Elevation.DropLines = True
```

To write code in C#

C#

```
C1Chart3D1.ChartGroups[0].Elevation.DropLines = true;
```

Formatting Drop Lines

Use the Pattern property to set the line drawing pattern, the Thickness property to set its width, and the Color properties to set the line color for a Chart3DLineStyle. The valid patterns are shown below:

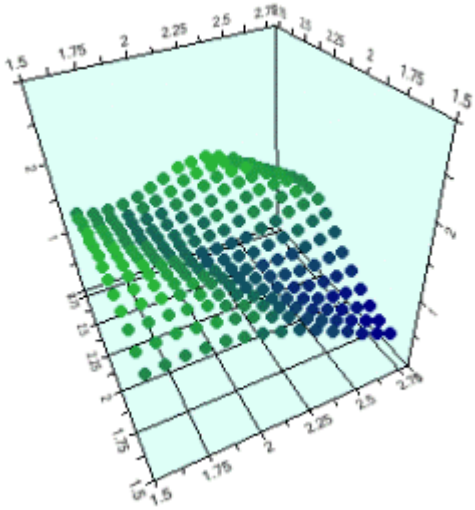
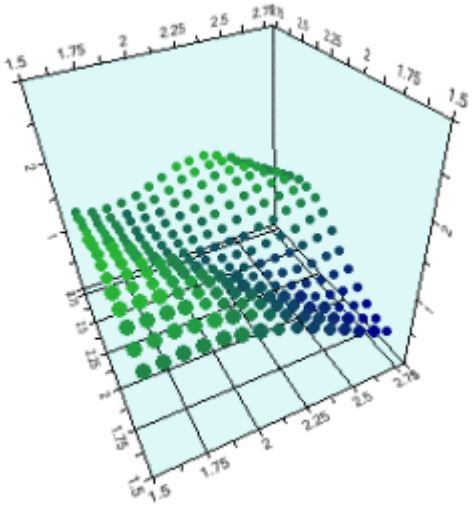
Type	Image
Solid	————
Dash	— — —
Dot	· · · · · · · ·
DashDot	— · — · -
DashDotDot	—··—··—

These properties are available at design time under the **LineStyle** node in the **Chart3DStyle Collection Editor**.

Scatter Plot Depth Cue

Use the DepthCue property of Chart3DElevation object to make emphasis on depth effect. When **DepthCue** is set to **True**, symbol size is dependent on its distance from the eye.

The following table shows how the Scatter Plot appears when the DepthCue is disabled and enabled:

DepthCue = False	**DepthCue = True**
	

Scatter Plot Lines and Symbols

The attributes that define how a series of data looks in the scatter plot are called a Chart3DStyle. Each series has its own Chart3DStyle, which can be customized. The Chart3DStyle object allows the following properties of a series to be changed programmatically:

- The LineStyle property, which defines the line style for the object. This specifies the color, pattern, and thickness of drop lines.
- The SymbolStyle property, which defines the symbol style for the object. This specifies the color, size and shape of points.

The Chart3DStylesCollection contains the Chart3DStyle objects defined for each series. The following methods are defined for the Chart3DStylesCollection:

Method	Description
AddNewStyle()	Add a new **ChartStyle** object to the collection.
Remove(index)	Remove a **ChartStyle** object from the collection.

Normally, manual adding or removing of Chart3DStyle objects from the collection is not needed. If a Chart3DStyle object already exists when its corresponding series is created, the previously created Chart3DStyle object is used to display the data in this series.

Looping through the Chart3DstylesCollection can quickly change the behavior of all of the lines or points in a chart. For example, the following code lightens all of the points in a chart whenever the mouse is clicked:

To write code in Visual Basic

Visual Basic

```
Private Sub C1Chart3D1_Click(ByVal sender As Object,
 ByVal e As System.EventArgs) Handles C1Chart3D1.Click
  Dim Style As Chart3DStyle
  For Each Style in Chart3D1.ChartGroups.ChartStyles
    Style.SymbolStyle.Color = Color.White
  Next Style
End Sub
```

To write code in C#

C#

```
private void c1Chart3D1_Click(object sender, System.EventArgs e)
{
  foreach(Chart3DStyle sty in c1Chart3D1.ChartGroups.ChartStyles)
    sty.SymbolStyle.Color = Color.White;
}
```

Setting the line pattern

The following sets the line pattern for the second series to a dotted line:

To write code in Visual Basic

Visual Basic

```
C1Chart3D1.ChartGroups.ChartStyles(1).LineStyle.Pattern = LinePatternEnum.Dot
```

To write code in C#

C#

```
C1Chart3D1.ChartGroups.ChartStyles[1].LineStyle.Pattern = LinePatternEnum.Dot;
```

Setting the symbol style

And the following statement sets the symbol style for the third series to an unfilled circle:

To write code in Visual Basic

```
Visual Basic
C1Chart3D1.ChartGroups.ChartStyles(2).SymbolStyle.Shape = SymbolShapeEnum.Circle
```

To write code in C#

```
C#
C1Chart3D1.ChartGroups.ChartStyles[2].SymbolStyle.Shape = SymbolShapeEnum.Circle;
```

Formatting data point symbols

You can easily format the Scatter Plot's data point symbols shape, size, color, and width using the properties of the Chart3DSymbolStyle class.

Use the Symbol Shape property to set the symbol type, the Size property to set its size, and the Color properties to set the symbol color for a ChartStyle. The valid types of symbols are shown below:

Type	Image	Type	Image
None		Vertical Line	
Dot	●	Horizontal Line	−
Box	■	Cross	+
Triangle	▲	Circle	○
Diamond	◆	Square	□
Star	*	Inverted Triangle	▼
Diagonal Cross	×	Open Triangle	△
Open Diamond	◇	Open Inverted Triangle	▽

These properties are available at design time under the **LineStyle** node in the **Chart3DStyle Collection Editor**.

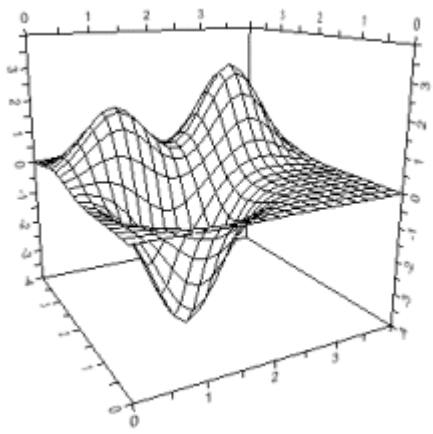
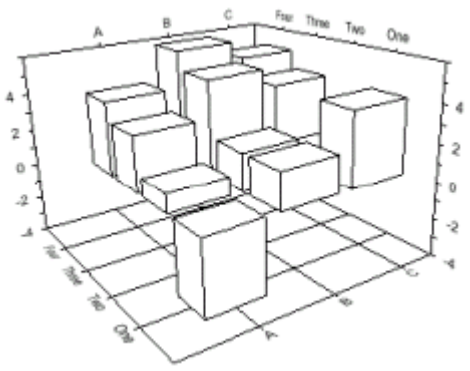
Meshed and Shaded Charts

The basic appearance of the **Surface** and **Bar** chart types is controlled by the combination of two properties: IsMeshed and IsShaded. These properties are located in the **Chart3DGroup Collection Editor** (IsMeshed and IsShaded under the **Elevation** node) that can be accessed through the **ChartGroups** property in the .NET Properties window.

Using IsMeshed

When the IsMeshed property is set to **True** for **Surface** or **Bar** charts it displays each chart type as the following:

- **Surfaces:** The chart displays the X-Y grid projected onto the 3D surface in a 3D view with a Z-axis. The chart honors rotation and perspective control.
- **Bars:** The chart draws the outline of all bars.

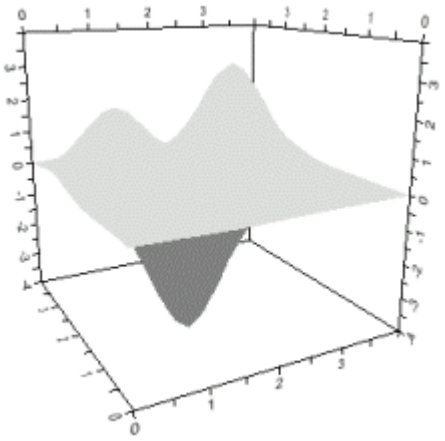
Surface Example	Bar Example
	

Note: The chart's top and bottom mesh color and other mesh properties can be customized.

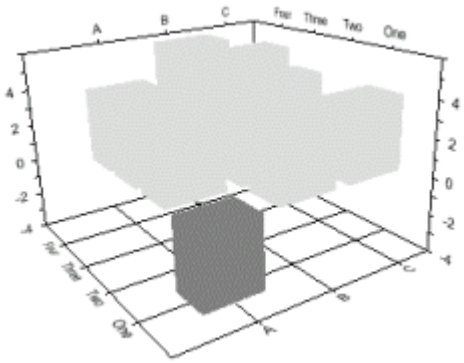
Using IsShaded

When the IsShaded property is set to true for Surface or Bar charts it displays each chart type as the following:

- **Surfaces:** The chart displays the data as a flat shaded surface in a 3D view with a Z-axis. Top and bottom shading colors may be set. The chart honors rotation and perspective control.



- **Bars:** The chart draws the bars as a flat shade.

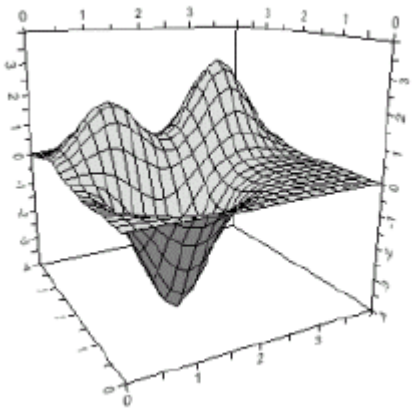
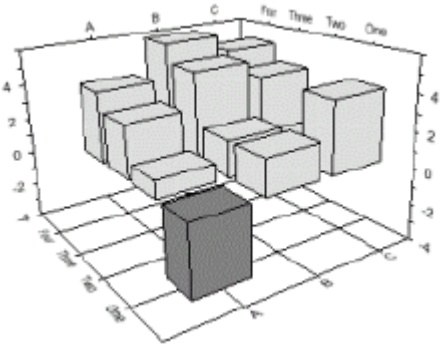


Note: The chart's top and bottom surface color and other surface properties can be customized.

Using IsMeshed and IsShaded

Use a combination of the IsMeshed and IsShaded properties to set the chart's basic appearance.

In this example the IsShaded and the IsMeshed properties are both set to **True** to display a Shaded and Meshed Surface 3D and Bar chart.

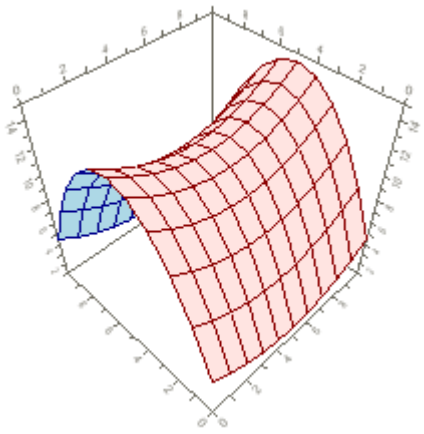
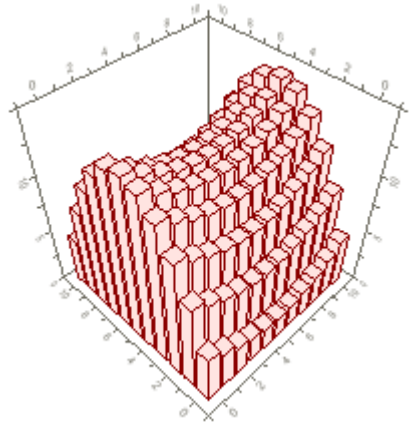
Surface Example	Bar Example
	

Adding Colors to the Meshed and Shaded Charts

When the IsShaded and the IsMeshed properties are both set to **True** you can add distinct colors to the top and bottom for the meshed and shaded charts for the 3D Bar or Surface type.

The MeshTopColor and MeshBottomColor properties set the top and bottom colors for the Surface charts and set the top colors for the Bar charts.

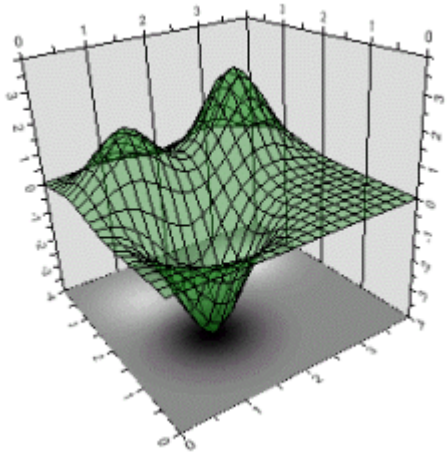
In this example the MeshTopColor and MeshBottomColor are set to DarkRed and DarkBlue colors to distinguish between the top and bottom surface of the meshed part of the Surface and Bar charts. Also the ShadedTopColor and ShadedBottomColor are set to MistyRose and LightBlue to distinguish between the top and bottom surface of the shaded part of the Surface and Bar charts.

Surface Example	Bar Example
	

Note: In the Bar chart above, the colors for the MeshBottomColor and the ShadedBottomColor properties are not visible on the bottom surface because of the Bar chart's present rotation.

Creating Transparency for the Meshed and Shaded Charts

Use the Transparency property to set transparency of data drawing. Valid values of Transparency are 0 through 255. 0 - transparent, 255 - opaque.



Note: The transparency is not applied to **Zoned** and **Contour** charts.

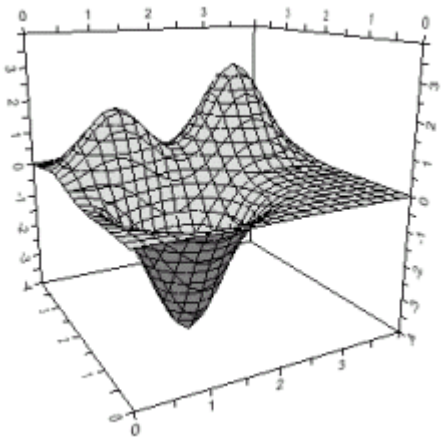
Contour and Zoned Charts

You can easily add contours and zones to 3D charts by enabling the IsContoured and IsZoned properties. These properties are located in the **Chart3DGroup Collection Editor** (IsContoured and IsZoned are under the Contour node) that can be accessed through the ChartGroups property in the .NET Properties window. For more information on the **Chart3DGroup Collection Editor**, see Chart3DGroup Collection Editor.

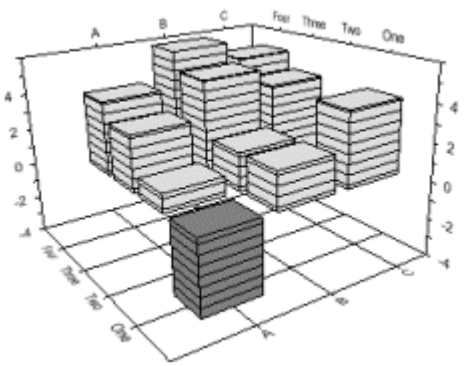
Using IsContoured Property

The Surface and Bar 3D charts behave as the following when they are contoured:

- **Surfaces:** The chart examines the distribution of the data and draws contour lines demarcating each of the contour levels.



- **Bars:** The chart draws contour lines around the bars, demarcating each contour level.

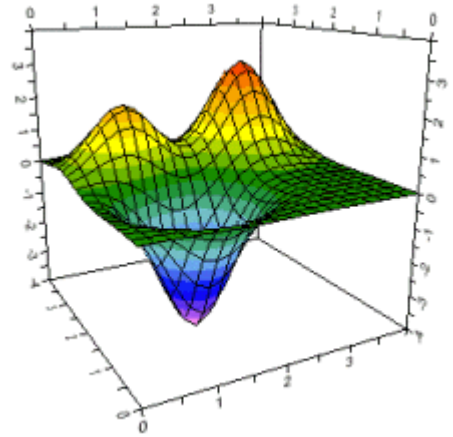


Note: The chart's contour levels, ContourStyles, and other contour properties can be customized.

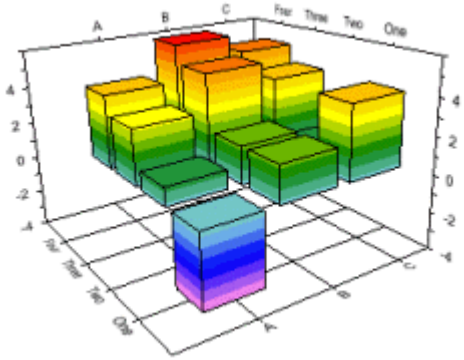
Using IsZoned Property

The Surface, Bar, and Scatter 3D charts behave as the following when they are zoned:

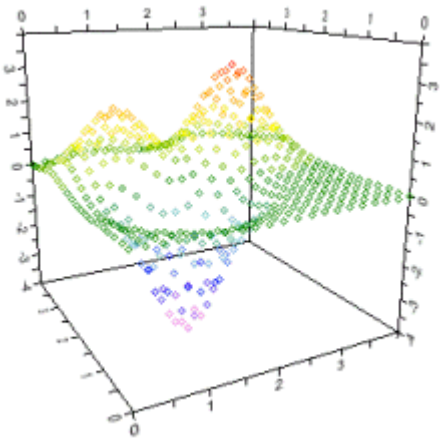
- **Surfaces:** The chart examines the distribution of the data and fills each level with a solid color.



- **Bars:** The chart fills each level within each bar with a solid color.



- **Scatter**: The points are colored according to their z-value.

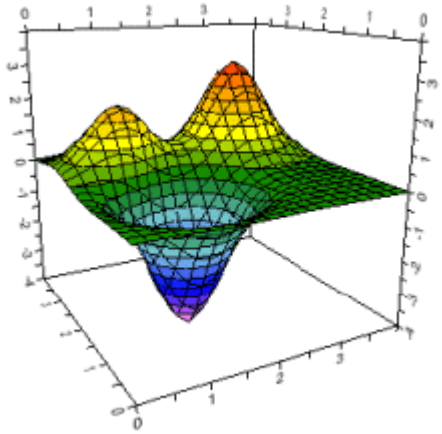
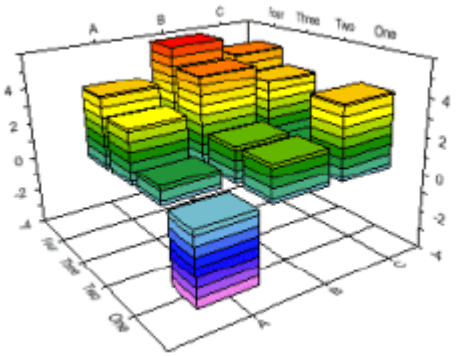


Note: The chart's contour levels, ContourStyles, and other zoning properties can be customized

Using IsContoured and IsZoned Properties

You can combine contours and zones to Surfaces, Bars, and Scatter 3D charts by enabling both IsContoured and IsZoned properties.

The following table illustrates a combination of contours and zones in Surface and Bar 3D charts:

Surface Example	Bar Example
	

Note: The chart's contour levels, ContourStyles, and other contour and zoning properties can be customized.

3D Chart Data Layouts

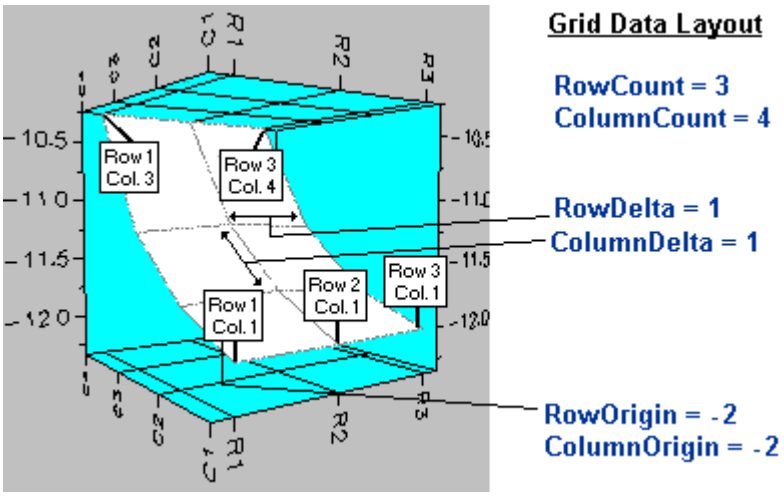
The 3D chart displays three-dimensional data. Each data point consists of an X/Y Cartesian coordinate and a Z elevation value. It charts surfaces that are increasing in X and Y and cannot chart surfaces that fold back in X or Y (such as a sphere).

The data must be supplied in one of three basic layouts, **Grid**, **Irregular Grid**, or **Point**. Data can come from many different sources, such as data files or the results of mathematical formulae. Data files can be loaded and saved, and data itself can be customized using the .NET Properties window.

The following topics explain when to use the appropriate type of **Grid**, **Irregular Grid**, or **Point** layout and how to edit them.

Grid Data Layout

Use Grid layout when the X-coordinates of each point and the Y-coordinates of each point are always the same distance apart. The following diagram illustrates the characteristics of Grid layout.



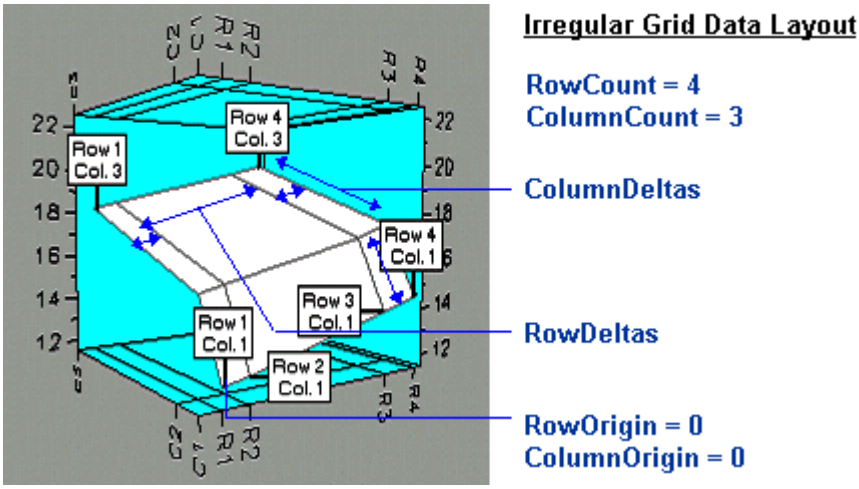
Column 1	Column 2	Column 3	Column 4	Z-Values
–12	–11.75	–11.25	–10.25	Row1
–12	–11.7	–11.2	–10.25	Row2
–12	–11.75	–11.25	–10.25	Row3

The important Grid layout characteristics are:

- The number of rows and columns, the origins, and the single row and column delta value defines the X/Y grid.
- The grid spacing for all rows is set by the RowDelta property. The grid spacing for all columns is set by the ColumnDelta property.

Irregular Grid Data Layout

Use Irregular Grid layout when the X-coordinates of each point or the Y-coordinates of each point are not the same distance apart. The following diagram illustrates the characteristics of Irregular Grid layout:



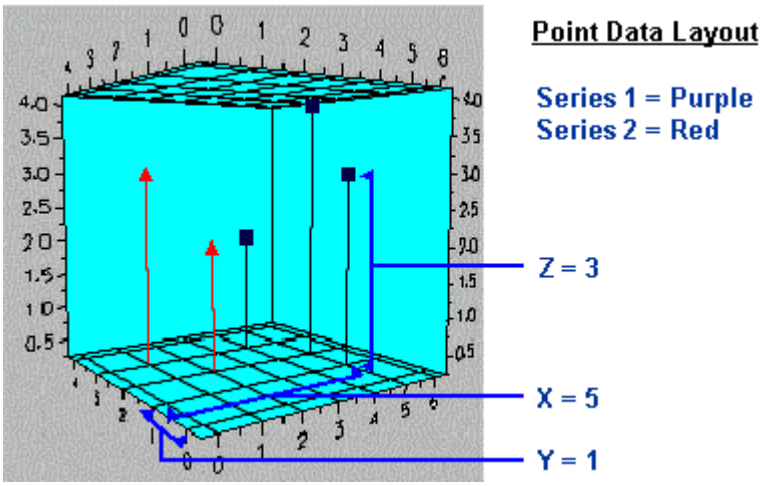
Column 1	Column 2	Column 3	Z-Values
12.25	15.25	18.25	Row 1
12.5	15.5	18.5	Row 2
14	17	20	Row 3
14.5	17.5	20.5	Row 4

The important Irregular Grid layout characteristics are:

- The number of rows and columns, the origins, and the irregular grid spacing deltas defines the X/Y grid.
- Each row and column has its own RowDelta / ColumnDelta value that sets the grid spacing between that row/column and the previous one.

Point Data Layout

Use Point Data Layout for scatter plots when charting multiple series of points. The following diagram illustrates the characteristics of Point layout.



		X	Y	Z
Series 1	Point 1	5	1	3
	Point 2	5	2	4
	Point 3	4	3	2

		X	Y	Z
Series 2	Point 1	2	2	2
	Point 2	1	3	3

The important Point layout characteristics are:

- The points are grouped into one or more series.
- The scatter plots' X, Y, and Z Cartesian coordinates specify each individual point.

3D Axes

The 3D chart has an X, Y, and Z axis that can be controlled by the Chart3DAxis object. You can customize the look and feel of the axes through the Chart3DAxis object and then set the X, Y, or Z axis object through the AxisX, AxisY, and AxisZ properties for the new Axes settings to appear on the X, Y, and Z axes.

This section describes the most common axis configuration scenarios used to make the chart more readable such as labeling, scaling, and formatting.

Axis Appearance

You can modify the axis title's alignment, text, and font.

Alignment

The HorizontalAlignment property can be set to five different settings: **Center**, **Far**, **General**, **Justify** or **Near**. Setting the alignment to **Center** centers the axis title in comparison to the ChartArea. Setting the alignment to **Near** places the axis title to the left side of the Chart3DArea. Setting the alignment to **Far** places the axis title to the right side of the Chart3DArea.

Text

Use the Text property to set the text for the Axes.

Font

Use AxisTitleFont property of Axes collection to specify font for the axis titles. All axes titles use the same font. See 3D Chart Fonts for more information on using Fonts.

> **Note:** The axis title font size is measured in hundredths of the unit cube size.

The font size and style for the axis title can be changed by manipulating the Font object of the axis. To access the font properties at design time click the **ellipsis** next to the **Font** node or expand the **Font** node under the axis object in the Visual Studio Properties window. To programmatically modify the Axis font properties, enter the following:

To write code in Visual Basic

Visual Basic

```
Dim f As Font = New Font("Arial", 8, FontStyle.Bold)
C1Chart1.ChartArea.AxisX.Font = f
```

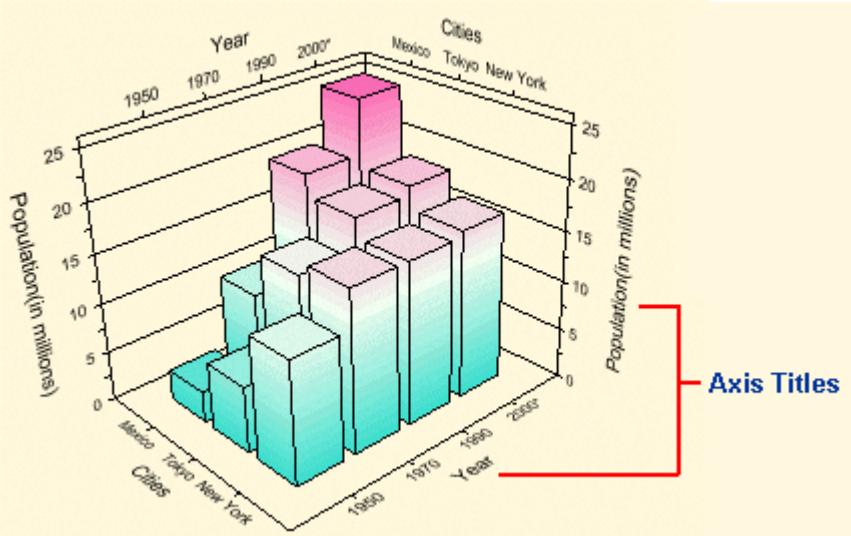
To write code in C#

C#

```
Font f = new Font("Arial", 8, FontStyle.Bold);
C1Chart1.ChartArea.AxisX.Font = f;
```

Axis Title and Rotation

Adding a title to an axis clarifies what is charted along that axis. A title with a specified font can be added to any axis.



To Add an Axis Title:

Use the axis Title property to add a title to an axis. To remove the title, delete the text.

Adding a title to an axis clarifies what is charted along that axis. Axis titles can be added to **Area**, **XY-Plot**, **Bar**, **HiLo**, **HiLoOpenClose** or **Candle** charts. The title or the annotation along the axis can also be rotated.

To Rotate an Axis Title:

Use the axis **Rotation** property to rotate the axis title to 90, 180, or 270 degrees. The 90 and 270-degree rotations are most efficient for vertical axes.

To Rotate the X-Axis:

Use the RotationX property to specify the number of degrees of rotation of the plot cube about the X-axis.

To Rotate the Y-Axis:

Use the RotationY property to specify the number of degrees of rotation of the plot cube about the Y-axis.

Axis Tick Marks

The chart automatically sets up the axis with both major and minor ticks. The UnitMajor and UnitMinor properties set the units by which the ticks will be spaced. When the UnitMajor property is set, the UnitMinor property is automatically set by the chart to half the UnitMajor value. Although the chart automatically sets the UnitMinor property, it also can be manually changed to a different value.

Axis Grid Lines

Grid lines are lines that appear perpendicular with major/minor tick marks at unit major/minor intervals. The lines that are placed coincident with major tickmarks at UnitMajor intervals are controlled by the MajorGrid property. Grid lines can help improve the readability of the Chart when you are looking for exact values. The major grid lines appearances

are controlled by the Chart3DGridLines properties.

To set grid lines on the XY-plane

Set the IsOnXYPlane property to true for the grid lines to appear on the XY-Plane.

To set grid lines on the XZ-plane

Set the IsOnXZPlane property to true for the grid lines to appear on the XZ-Plane.

To set grid lines on the YZ-plane

Set the IsOnYZPlane property to true for the grid lines to appear on the YZ-Plane.

Grid lines appearance

You can determine the style for the major grid lines through the Style property which returns the Chart3DLineStyle object that allows you to customize the grid lines color, pattern, or thickness.

Axis Bounds

The chart usually recalculates and redraws the axes whenever data changes. The bounds of any axis can be fixed so that it is not recalculated when data changes. For the X and Y-axes, a portion of data can be framed by setting axis bounds inside the data bounds.

To Set the Axis Minimum and Maximum:

Use the Axes Min and Max properties to fix the minimum or maximum axis extent at a particular value. AutoMax and AutoMin properties allow the chart to automatically determine axis bounds based on the data bounds. These properties are located on the **Chart3DAxis Collection Editor**. This editor can be accessed by:

1. Expanding the **ChartArea** node in the Visual Studio Properties window.
2. Clicking on the **ellipsis** button adjacent to the **Axes** property.

Z-axis Notes

The Z-axis minimum/maximum cannot be set inside the Z-range of the data. If a fixed extent exists for the Z-axis and a subsequent data change puts the range outside of the axis, the chart sets the Z-axis **AutoMax** and **AutoMin** properties so that its extent is recalculated.

X- and Y-axis Notes

The X-axis and Y-axis minimum/maximum cannot be set to a location that does not coincide with a line of data. The values are always adjusted upwards to coincide.

Axis Scaling

You can determine the amount of scaling in the Y-axis or X-axis direction in relation to the height (Z-axis) of the cube by specifying the amount for the YScale and XScale properties.

You can specify the scaling programmatically through the YScale and XScale properties or you can specify it at design time through the **Chart3D Wizard** or the **Chart3D Properties** designers. For additional information on these designers, see Design-Time Tools for Creating 3D Charts.

Axes Annotation

The annotation along each axis is an extremely important part of any chart. The 3D chart can annotate any axis with numbers based on the data (**Values**) or with text displayed at axis coordinates (**ValueLabels**). The X and Y-axes can also be annotated with text for each point in the data (**DataLabels**). The following topics provide more information about the position, different types of axes annotation (Values annotation, Value Labels annotation, Data Labels annotation), and distinct methods to label axes.

Axes Annotation Position

Axis annotation typically appears beside its axis. This may be a problem on charts with an origin that is not at the axis minimum or maximum. The chart can automatically determine where to place annotation in different situations, depending on the chart type.

3D Values Annotation

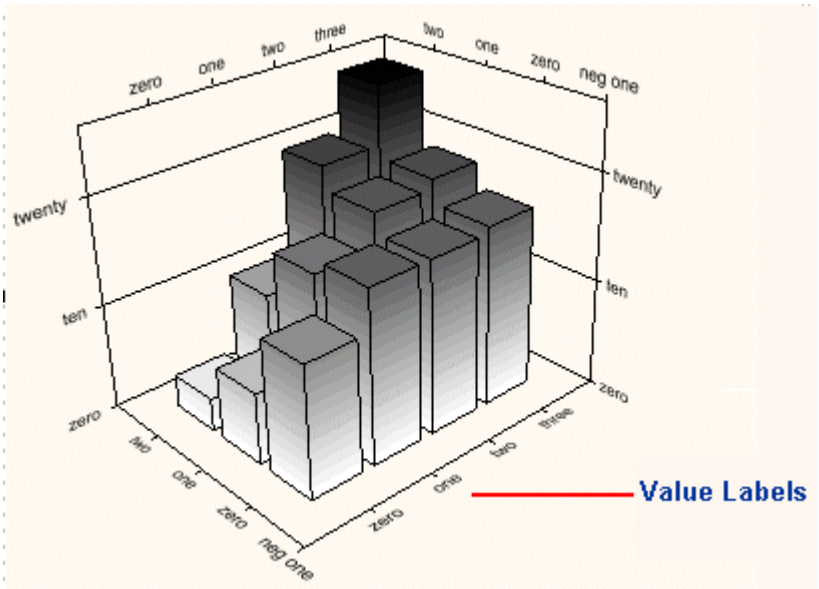
Values annotation automatically generates numeric annotation based on the data itself. **Values** annotation can be used for any axis.

Use the axis AnnoMethod property to specify **Values** annotation for an axis. The AnnoMethod property can be set through the **Chart3D Axis Collection Editor**. This editor can be accessed by:

1. Expanding the **ChartArea** node in the Visual Studio Properties window.
2. Clicking on the **ellipsis** adjacent to the **Axis** property.

3D Value Labels Annotation

A very flexible type of annotation, **ValueLabels** displays text defined at a specific axis coordinate. This is useful to label only specific coordinates, or to produce annotation in a form that the chart does not provide. **ValueLabels** annotation can be used for any axis.



Every label displayed on the axis is one ValueLabel. Each ValueLabel has a Value property and a Text property.

Use the Value property to set the axis coordinate to display the label. Use the Text property to specify the text to display. All **ValueLabels** properties are located on the **Chart3DLabel Collection Editor**, which can be accessed by:

1. Expanding the **Labels** node in the **Chart3Daxis Collection Editor** window.

2. Clicking on the **ellipsis** adjacent to the **Labels** property.

Set **AnnoMethod** to ValueLabels for an axis and use this editor to define the labels.

Data Labels Annotation (X and Y-axes only)

DataLabels are a collection of labels that display text defined at each point along the X or Y-axes. **DataLabels** annotation cannot be used for the Z-axis.

Use the RowLabels and ColumnLabels ChartGroups properties to define and edit DataLabels. These properties can be accessed through the **Chart3DDataLabel Collection Editor**. The respective editors for RowLabels and ColumnLabels can be accessed by:

1. Expanding the **ChartGroups** node in the Visual Studio Properties window.
2. Clicking on the **ellipsis** adjacent to the ColumnLabels or RowLabels properties.

Set AnnoMethod to **DataLabels** for an axis and use this tab to define the labels.

> **Note: DataLabels** annotation can be used only for regular and irregular grid data.

Distinct Methods to Label Axes

There are three distinct ways to label axes when using the 3D Chart control:

- The axis can be automatically labeled based on the range of data.
- Surface lines or bar row/columns can be individually labeled.
- Labels can be placed at explicit locations along an axis.

The axis labeling method in use for a particular axis is specified by the AnnoMethod property. The **C1Chart3D.AnnotationMethodEnum** specifies valid values for this property.

If the AnnoMethod property is set to AnnotationMethodEnum.**Values**, **C1Chart3D** will automatically annotate the axis based on the range of the data. This is most suitable for the Z-axis, and for surface charts.

If the AnnoMethod property is set to AnnotationMethodEnum.**DataLabels**, **C1Chart3D** uses a list of strings to annotate each grid line or bar. This labeling method can only be used on the X and Y-axes. For the X-axis, the labels are supplied by setting the RowLabels property of the Chart3DGroups object for the chart. For example, the following code specifies three data labels for each of the three rows of a chart:

To write code in Visual Basic

Visual Basic

```
With C1Chart3D1
  'assume three rows in chart
  .ChartArea.Axes("X").AnnoMethod = AnnotationMethodEnum.DataLabels
  With .ChartGroups.RowLabels
    .Add(0, "Row 1")
    .Add(1, "Row 2")
    .Add(2, "Row 3")
  End With
End With
```

To write code in C#

C#

```
C1.Win.C1Chart3D.ChartGroups cgs = C1Chart3D1.ChartGroups;

//assume three rows in chart
C1Chart3D1.ChartArea.Axes["X"].AnnoMethod = AnnotationMethodEnum.DataLabels;
cgs.Add(0, "Row 1");
cgs.Add(1, "Row 2");
cgs.Add(2, "Row 3");
```

Similarly, labels for the Y-axis are supplied by setting the ColumnLabels property of the Chart3DGroups object.

If the AnnoMethod property is set to AnnotationMethodEnum.**ValueLabels**, **C1Chart3D** places labels at explicit locations along an axis. The ValueLabels property, which is a ValueLabels **collection**, supplies this list of strings and their locations. For example, the following code sets chart labels at the locations 10, 20, and 30:

To write code in Visual Basic

Visual Basic

```
With C1Chart3D1.ChartArea.Axes("X")
  .AnnoMethod = AnnotationMethodEnum.ValueLabels
  With .ValueLabels
    .Add(10#, "Label 1")
    .Add(20#, "Label 2")
    .Add(30#, "Label 3")
  End With
End With
```

To write code in C#

C#

```
C1.Win.C1Chart3D.Chart3DAxis axis = C1Chart3D1.ChartArea.Axes["X"];
C1.Win.C1Chart3D.Chart3DAxisLabelsCollection valueLabels = axis.ValueLabels;
axis.AnnoMethod = AnnotationMethodEnum.ValueLabels;
valueLabels.Add(10#, "Label 1");
valueLabels.Add(20#, "Label 2");
valueLabels.Add(30#, "Label 3");
```

Design-Time Tools for Creating 3D Charts

The chapter includes three options for creating 3D charts at design time. You can create charts using the **Smart Designer**, **Chart Wizard**, or the **Chart Properties** designer. After reading this chapter you can decide which method for creating a chart is easiest for you.

Working with the Smart Designer

The **Smart Designer** allows you to quickly set Chart properties without leaving the design form. This solves the earlier problem of having to drill down through Chart's properties in the Properties window. You can use the Smart Designer feature to create a functional 3D Chart at design time.

The following section introduces the toolbars included in the Smart Designer for the **C1Chart3D** control.

Primary Toolbars

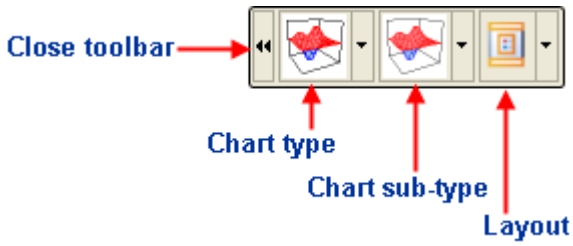
The Smart Designer has three primary toolbars for the **C1Chart3D** control. The primary toolbars include the following:

- C1Chart3D toolbar
- ChartArea toolbar

This section describes the functionality of the buttons in C1Chart's primary toolbars.

Chart3D Toolbar

The primary **C1Chart3D** toolbar for the **C1Chart3D** control includes a close toolbar, chart type, chart sub-type, and layout button. The figure below provides a label for each of the command buttons in the **C1Chart3D** toolbar.



C1Chart3D Toolbar's Command Buttons

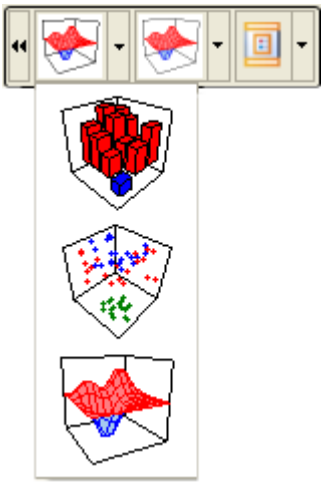
The following command buttons are available in the **C1Chart3D** toolbar:

- **Close Toolbar button**

  The close command button closes the toolbar once it is clicked.

- **Chart type button**

  The Chart type command has a drop-down menu that contains a selection of all the chart types provided by the **C1Chart** control. Hovering over each chart image with the mouse pointer exposes a label with the name of the selected chart type. You can choose from one of the following simple chart types: Bar, Scatter, and Surface.



- **Chart sub-type button**

  The **Chart sub-type** command also has a drop-down menu which contains a selection of all the chart sub-types provided by the **C1Chart3D** control. Hovering over each chart image with the mouse pointer exposes a label with the name of the selected chart sub-type. You can choose from one of the available sub-chart types that corresponds to the Bar, Scatter, or Surface.

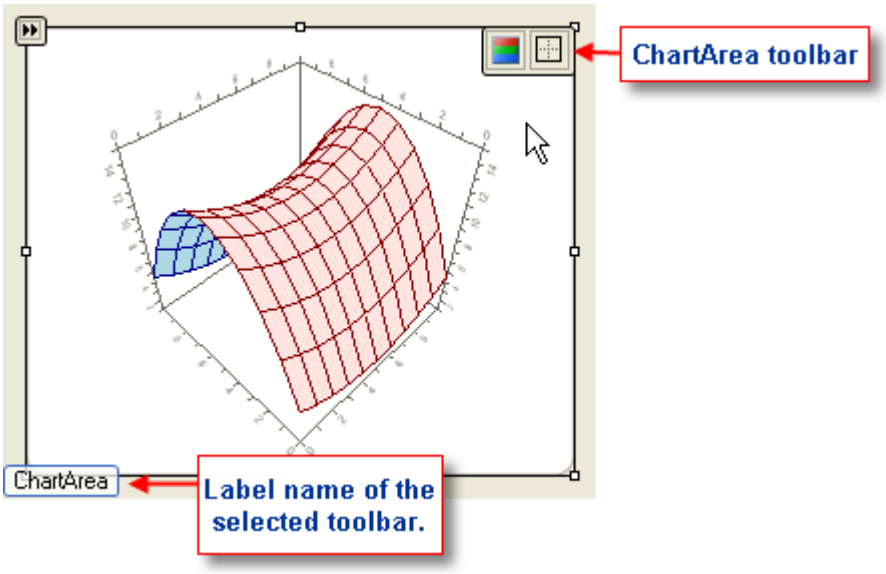
- **Layout button**

  The **Layout** command button has a drop-down menu which contains the Header, Footer, and Legend elements. Selecting one of the elements exposes either an editable Header, Footer, or Legend element directly on the Chart Area.

  Selecting either the Header or Footer element exposes an editable textbox along with a toolbar that provides formatting commands. If the toolbar does not appear instantly then you can click the left mouse button and slide it over the textbox to expose the toolbar. The image below illustrates a Header text box automatically added to the Chart Area on the **C1Chart3D** control.

ChartArea Toolbar

Another primary toolbar, the **ChartArea** toolbar for the **C1Chart3D** control includes a **Background** and a **Border** command button. The figure below shows how the **ChartArea** toolbar appears when the user selects the ChartArea on the **C1Chart3D** control. When a user selects a toolbar a label name is provided for the user's convenience.

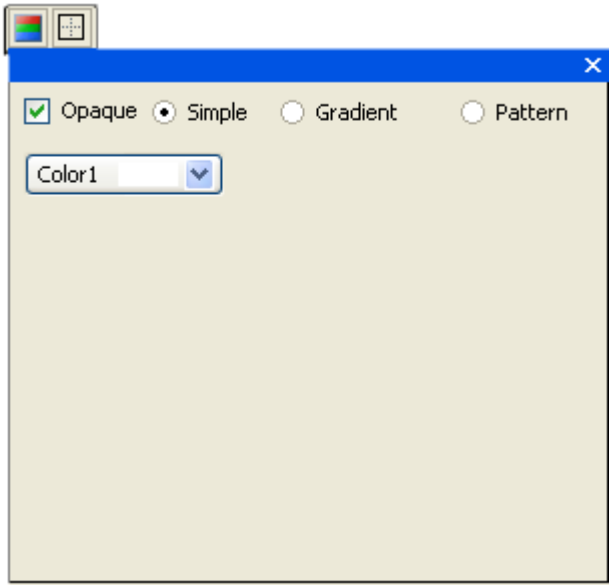


ChartArea Toolbar's Command Buttons

The following command buttons are available in the **ChartArea** toolbar:

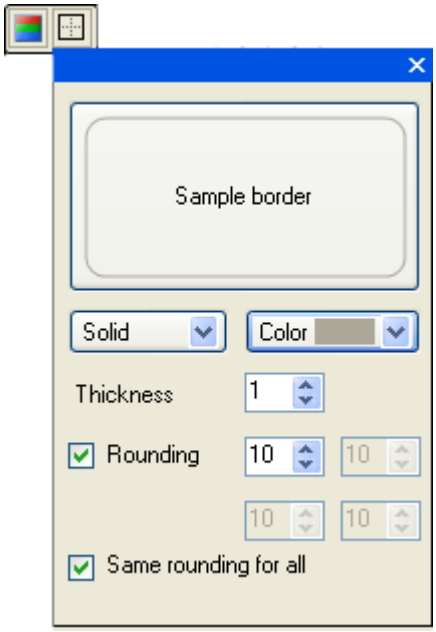
- **BackGround button**

  The **Background** command button has a drop-down box that contains three different types of styles for the background and a color drop-down list box for the user to specify a color for the ChartArea's background.



- **Border button**

  The **Border** command button includes a drop-down box that contains editable Border styles and colors for the ChartArea's border.



Secondary Toolbars

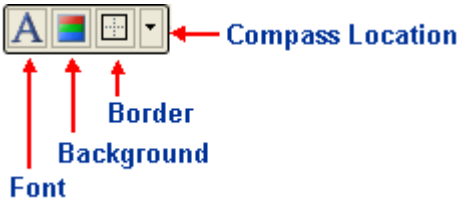
The Smart Designer has four secondary toolbars for C1Chart's additional elements such as the header, footer, label, and legend. The secondary toolbars include the following:

- Header toolbar
- Footer toolbar
- Legend toolbar

This section describes the functionality of the buttons in C1Chart's secondary toolbars.

Header and Footer Toolbars

The toolbar for the **Header** and **Footer** element includes the following command buttons: Font, Background, Border, and a Compass Location for the **Header** and **Footer** elements of the **C1Chart3D** control. The figure below provides a label for each of the command buttons for the Header and Footer toolbar.



Exposing the Header or Footer Toolbar

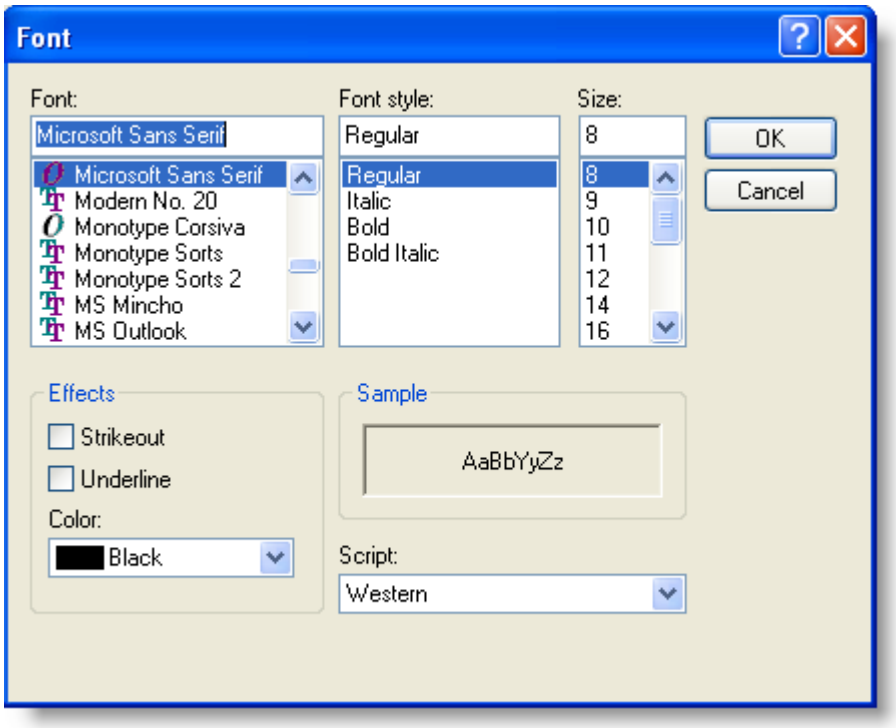
In order for the **Header** or **Footer** toolbar to appear you have to select either the **Header** or **Footer** from **C1Chart3D** toolbar's drop-down menu.

Header/Footer Toolbar's Command Buttons

The following command buttons are available in the **Header/Footer** toolbar:

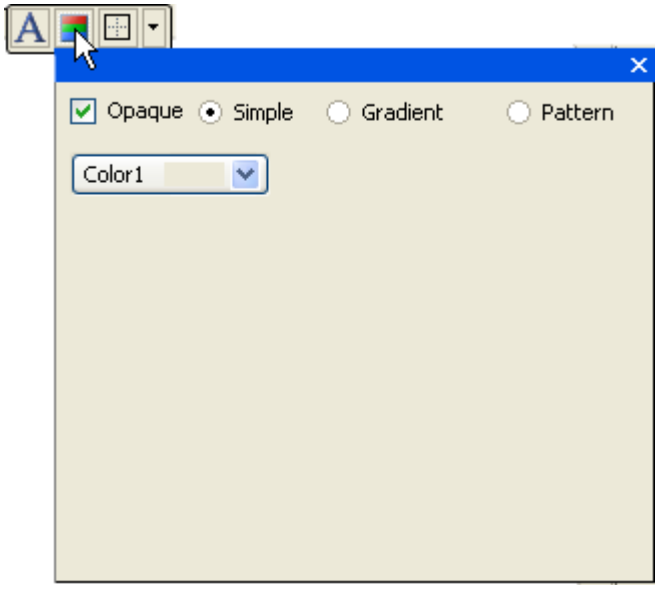
- **Font button**

  The **Font** command button exposes the **Font** dialog box for the **Header** and **Footer** element. Here the Font style can be modified for the Header and Footer's text.



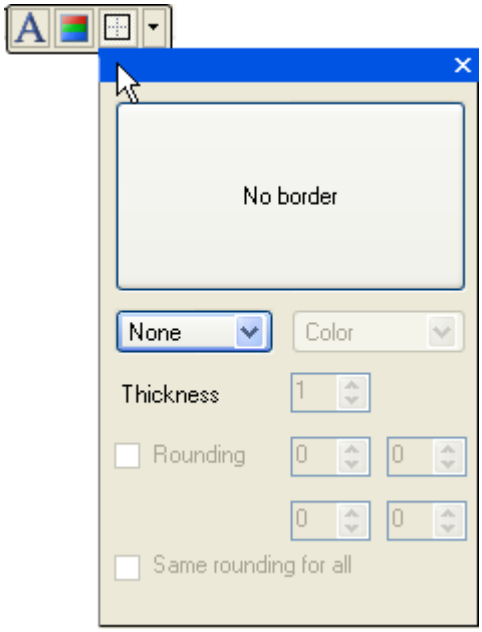
- **BackGround button**

  The **Background** command button has a drop-down box that contains three different types of styles for the background and a color drop-down list box for the user to specify a color for the **Header** and **Footer's** background.



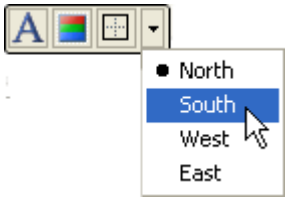
- **Border button**

  The **Border** command button includes a drop-down box that contains editable Border styles and colors for the **Header** and **Footer's** border.



- **Compass Location button**

  The **Compass** command button has a drop-down list box which includes a list of different compass directions (North, South, West, and East) for the user to choose from. The directions will position the **Header** or **Footer** in the North position which is above the Chart, the South position which is below the Chart, the West position which is to the left of the Chart, and the East position which is to the right of the Chart. The default compass position for the Header is north and the default position for the Footer is south.



Legend Toolbar

The toolbar for the Legend element is similar to the Header/Footer toolbars except that it includes an additional command button called Edit text. The Label toolbar also contains this command button.

Exposing the Legend Toolbar

In order for the **Legend** toolbar to appear you have to select the Legend item from the Chart toolbar's drop-down menu.

Legend Toolbar's Command Buttons

The following command buttons are available in the **Legend** toolbar:

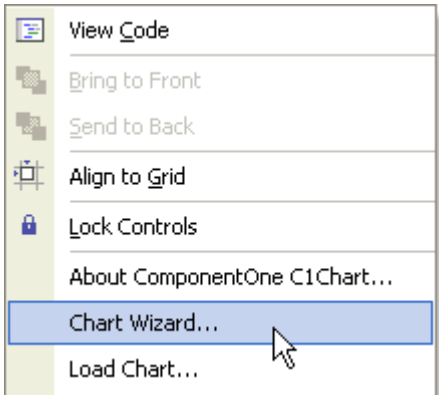
- **Font button**
  The **Properties** button for the Legend toolbar exposes the **Chart Properties** editor for the Legend once it is clicked by the user.
- **Background button**
  The **Background** button functions exactly like the rest of the Border command buttons for the **C1Chart** control toolbars.
- **Border button**
  The **Border** command button functions exactly like the rest of the Border command buttons for the **C1Chart** control toolbars.
- **Compass Location button**
  The **Compass** command button has a drop-down list box which includes a list of different compass directions (North, South, West, and East) for the user to choose from. The directions will position the **Header**, **Footer**, or **Legend** in the North position which is above the Chart, the South position which is below the Chart, the West position which is to the left of the Chart, and the East position which is to the right of the Chart. The default compass position for the Header is north and the default position for the Footer is South.

Working with the Chart3D Wizard

The **Chart Wizard** provides an easy three step process to guide you through the basic steps for creating a 3D chart. You can choose from various chart types, set up the chart's data layout, and edit the view of the chart by rotating or scaling the x or y-axis.

To access **Chart3D Wizard** at design time complete one of the following steps:

1. Right-click on the **C1Chart3D** control. Select **Chart Wizard** from the context menu.

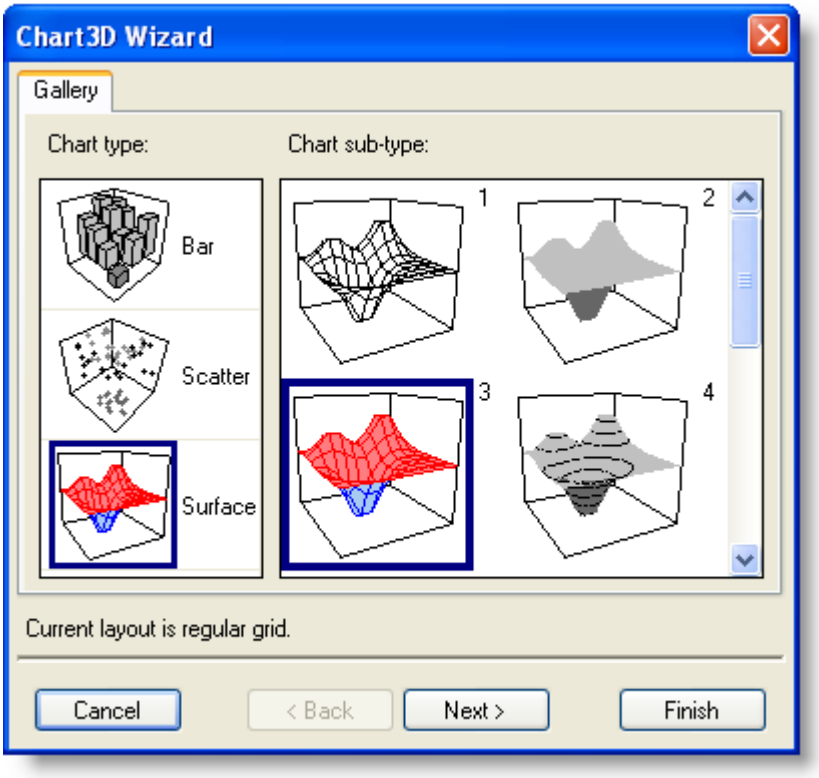


2. Open the **Chart3D Tasks** menu and click on the **Wizard** item to open the **Chart3D Wizard** designer.

The following topics walk you through the three step process for creating a basic 3D chart.

Step 1. Choose Chart Type

There are various chart types to choose from the chart wizard. This offers the ability to quickly create many different types of charts at design time. The chart types include Bar, Scatter, and Surface. In addition each chart type has individual subtypes, which allows further selection of chart types. See Basic 3D Chart Types for more detail about each chart type.

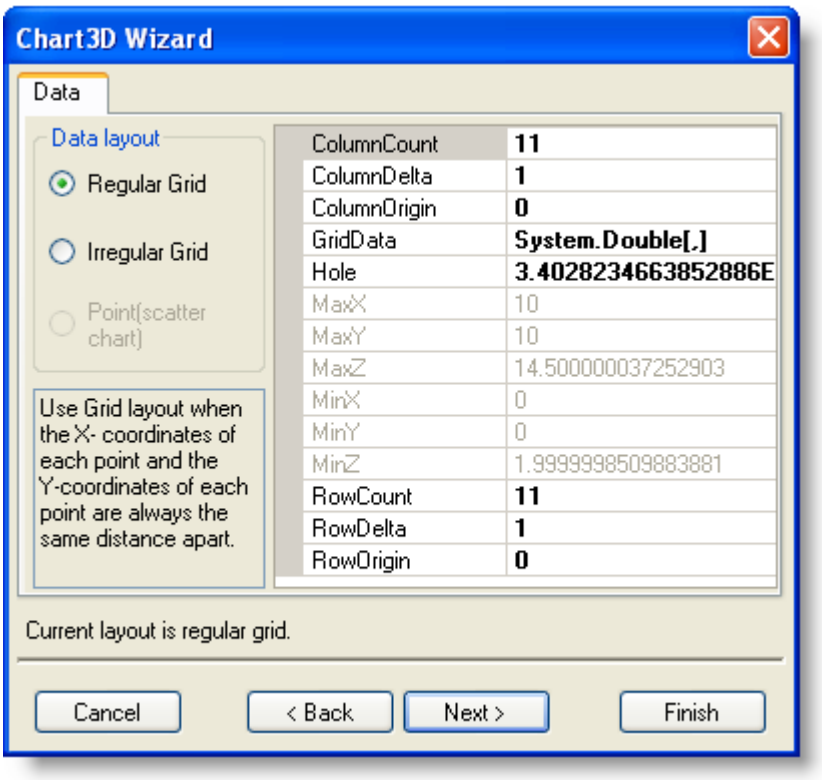


After a chart type is selected, click **Next** to set up specific details for the new chart.

Step 2. Setup Chart

The next step in the chart wizard, you can select the type of data layout and then set up its Chart3DDataSetGrid properties. The Grid layout is used when the X-coordinates of each point and the Y-coordinates of each point are always the same distance apart. For more information about the Grid layout, see Grid Data Layout. The Irregular Grid

layout is used when the X-coordinate of each point or the Y-coordinates of each point are not the same distance apart. Unlike Regular grids that use the **ColumnDelta** property, the Irregular Grid layout uses the **ColumnDeltaArray** property. For more information about the Irregular grid layout, see Irregular Grid Data Layout. The Point layout is used for scatter plots when you want to chart multiple series of points. When the Point layout is selected the **Chart3DpointSeries Collection Editor** is available to add or modify the series in the Chart3D Point chart type. For more information about the Point layout, see Point Data Layout.



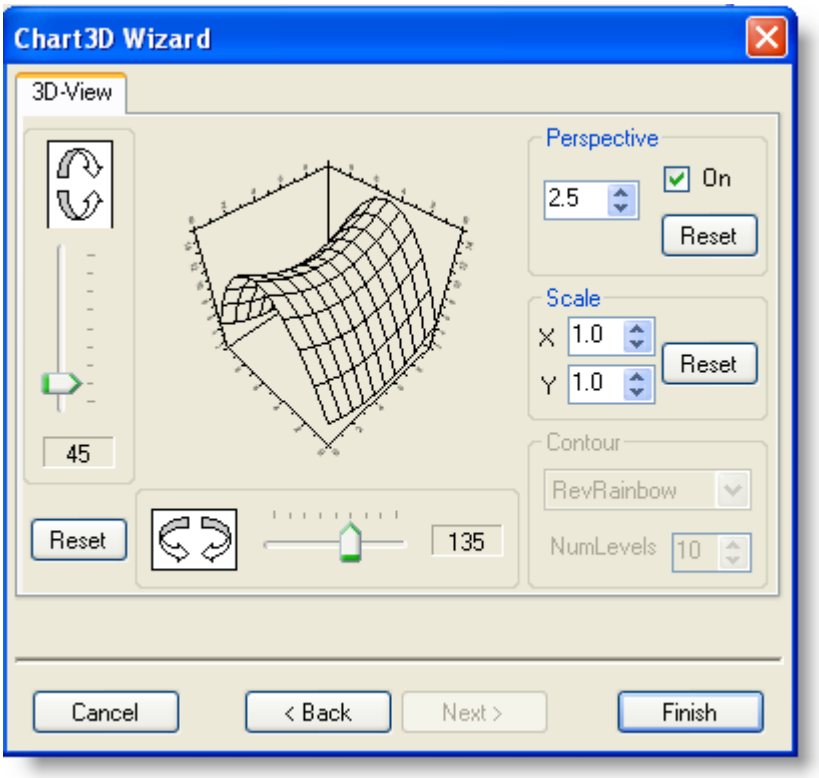
Once the settings for the Chart3DDataSetGrid object are specified, select **Next** to edit the Chart3D view.

Step 3. Edit the Chart3D View

The next step in the chart wizard is to modify the view of the 3D Chart. Under the **3D-View** tab, you can easily set or modify any of the properties in the Chart3DView class. To rotate the x-axis, move the slide up or down and to rotate the y-axis move the slider left to right. You can also change the 3D chart perspective by clicking the up or down arrow or by entering a value. To modify the perspective, enable the Perspective property by checking the **On** check box. To scale the X or Y axis, click on the up and down arrows in the combo box.

If you have a contour or zone chart, you can modify the ColorSet and NumLevels properties located in the Contour group.

A **reset** button is provided for each property in case you need to reset a specific property back to its default value.



One you are done editing the 3D chart you can select **Finish**.

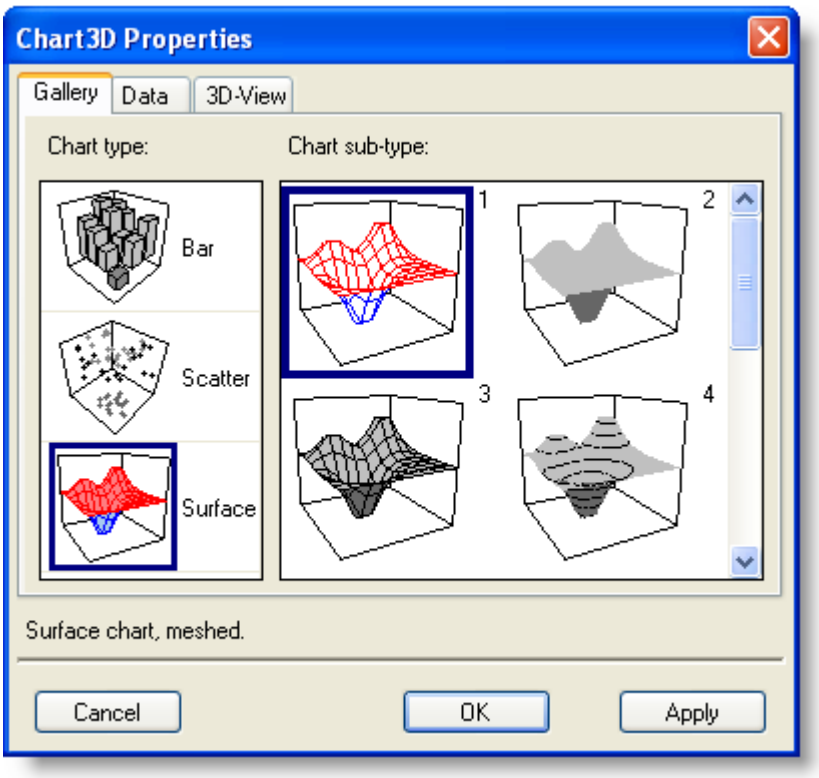
Working with the Chart3D Properties Designer

The **Chart3D Properties** designer provides an easy and interactive way to create and modify a new or existing chart. Like the **Chart3D Wizard**, it contains the same functionality as the **Gallery**, **Data**, and **View** tab. However, it displays all three of the tabs rather than displaying one tab at a time. The **Chart3D Properties** editor is accessible through the C1Chart3D's context menu or its Tasks menu.

The **Chart3D Properties** designer provides more options to address specific details with the design of the chart you are developing.

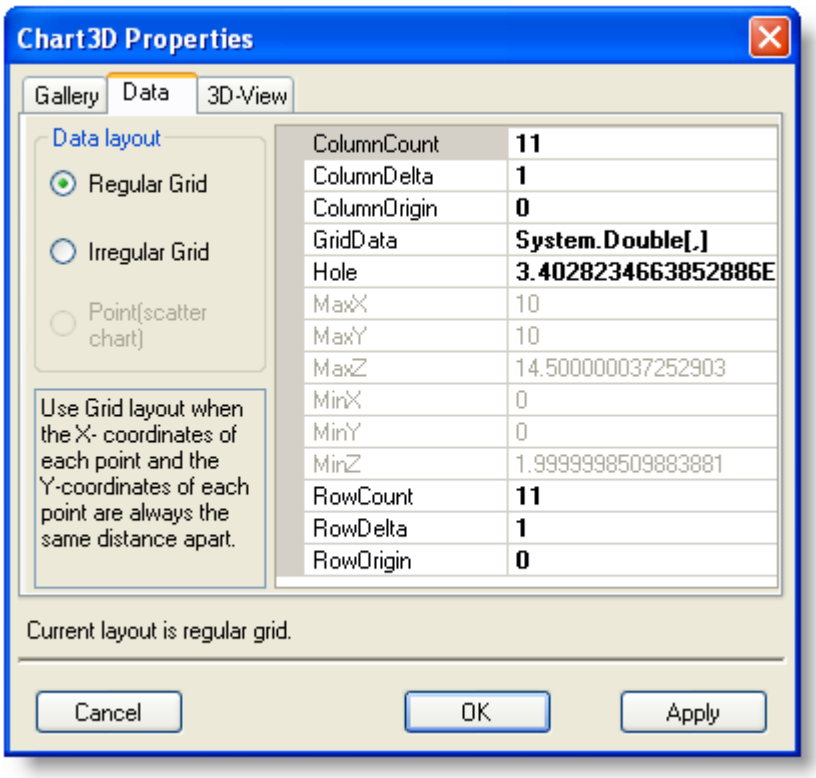
Gallery Tab

The **Gallery** tab provides options for choosing a chart type and/or a sub-type of a chart. The available chart types are located in the left side of the **Gallery** tab page and the available chart sub-types are located on the right side of the **Gallery** tab page. To see a description of all chart type selections, see Basic 3D Chart Types. You can choose from a variety of simple chart types or you could click on complex types to add more functionality to your chart.



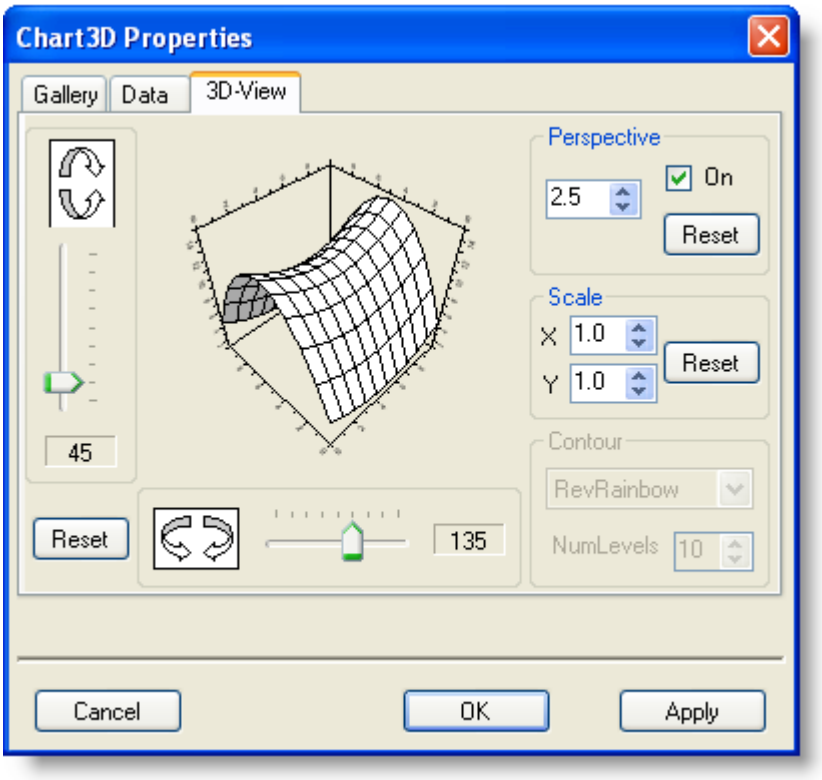
Data Tab

Select the **Data** tab from the **Chart3D Properties** dialog box to select the type of data layout and set its Chart3DDataSetGrid properties.



3D-View Tab

Select **3D-View** tab from **Chart3D Properties** dialog box to set or modify any of the properties in the Chart3DView class.



3D Data

The following topics explain how to plot, setup, copy, and configure data for use with **C1Chart3D**.

Data Organization

Data in a chart is organized into a chart group. In 3D Chart, a chart group is represented by a Chart3DGroups object. At present, only one ChartGroup can be stored in a ChartGroupsCollection.

Within a Chart Group, the three-dimensional data to be displayed is assumed to come either from a grid surface, or from one or more series of points. This data is represented by the Set property of Chart3DData class, which is of Chart3DDataSet type. Chart3DDataSet class is a base abstract class for all data types. The data displayed can be in one of three formats:

- Regularly-grid data, in which the X and Y-grid lines are evenly spaced (Chart3DDataSetGrid).
- Irregularly-grid data, in which the X and Y-grid lines are not evenly spaced (Chart3DDataSetIrGrid).
- Point data, in which the X, Y, and Z-coordinates can be specified without restrictions (Chart3DDataSetPoint).

To set the data format, create one from three datasets and assign it to the Set property of Chart3DData or set the Layout property to the appropriate data layout constant. The second way is useful at design time.

Handling Missing Data

If one coordinate for a data point is set to a special value known as the hole value, the chart treats the data point as a missing value, and does not display it on the chart. This hole value is specified by the Hole property of the Chart3DDataSet object.

For example, the following code treats the point in the third row and first column as a missing value:

To write code in Visual Basic

Visual Basic

```
Dim gridset As Chart3DDataSetGrid
gridset = C1Chart3D1.ChartGroups(0).ChartData.SetGrid
With gridset
  .Hole = 999.1
  .Item(0, 2) = 999.1
End With
```

To write code in C#

C#

```
Chart3DDataSetGrid gridset = C1Chart3D1.ChartGroups[0].ChartData.SetGrid;
gridset.Hole = 999.1f;
gridset[0,2] = 999.1f;
```

Grid and Irregular Grid Data

For grid and irregular grid data, the Chart3DDataSetGrid and Chart3DDataSetIrGrid objects define the following properties that control the data to be displayed in the chart:

- The RowCount and ColumnCount properties define the number of rows and columns of data.
- The RowOrigin and ColumnOrigin properties specify the origins of the X-axis and Y-axis respectively.
- Regular grid: RowDelta and ColumnDelta are properties that specify the space between neighboring rows and columns.
- Irregular grid: RowDeltaArray and ColumnDeltaArray are indexed properties that specify the space between neighboring rows and columns.
- The SetGrid property specifies the Z-coordinate of a particular data point. This property is indexed by row and column.

The following statement assigns the Z-coordinate of the point in the first row and third column to Zval:

To write code in Visual Basic

Visual Basic

```
Zval = C1Chart3D1.ChartGroups(0).ChartData.SetGrid(2, 0)
' assigns the Z-coordinate of the point in the first row and third column to Zval.
```

To write code in C#

C#

```
Zval = C1Chart3D1.ChartGroups(0).ChartData.SetGrid(2, 0);
// assigns the Z-coordinate of the point in the first row and third column to Zval.
```

Changing Data Values

To change a data value displayed in the chart, set its Item property. For example, the following statement changes the value of the point in the third row and second column to 3.14159:

To write code in Visual Basic

Visual Basic

```
C1Chart3D1.ChartGroups(0).ChartData.SetGrid(1, 2) = 3.14159
```

To write code in C#

C#

```
C1Chart3D1.ChartGroups[0].ChartData.SetGrid[1, 2] = 3.14159;
```

Adding Rows and Columns

To add rows to a chart group, do the following:

Set the RowCount property to the new number of rows.

If the data is irregularly-grid, set the RowDeltaArray property for the space between a new row and its predecessor.

Set the Item property for each of the new points.

For example, the following code adds a fourth row of three points to a ChartGroup:

To write code in Visual Basic

Visual Basic

```
With C1Chart3D1.ChartGroups(0).ChartData.SetGrid
  .RowCount = 4
  .Item(0, 3) = 5.17
  .Item(1, 3) = 5.84
  .Item(2, 3) = 6.33
End With
```

To write code in C#

C#

```
C1Chart3D1.ChartGroups[0].ChartData.SetGrid.RowCount = 4;
C1Chart3D1.ChartGroups[0].ChartData.SetGrid[0,3] = 5.17;
C1Chart3D1.ChartGroups[0].ChartData.SetGrid[1,3] = 5.84;
C1Chart3D1.ChartGroups[0].ChartData.SetGrid[2,3] = 6.33;
```

New columns can be added in much the same way:

Set the ColumnCount property to the new number of columns.

If the data is irregularly-grid, set the ColumnDeltaArray property for the space between a new column and its predecessor.

Set the Item property for each of the new points.

Working with Point Data

The data for the point data format is accessed by series. A series can be added or removed using the **Chart3DPointSeries Collection Editor** at design time or programmatically using **AddSeries** and **RemoveSeries** method of the Chart3DDataSetPoint class. For more information on accessing the **Chart3DPointSeries Collection Editor**, see Chart3DPointSeries Collection Editor.

The PointSeries Collection

Individual series are accessed via the SeriesCollection property of the Chart3DDataSetPoint object. The number of points in the series can be determined via the read-only **Count** property.

The Points property of Chart3DPointSeries class gives access to the points array of the series. It returns array of **Chart3DPoint** that represent coordinates of data points. To set new data points for the series, assign Points property to the new array of **Chart3Dpoint**.

Examples

The following creates point dataset and adds two series to it:

To write code in Visual Basic

Visual Basic

```
Dim pointset As Chart3DDataSetPoint
pointset = new Chart3DDataSetPoint()
Dim len As Integer = 20
```

```
Dim i As Integer
Dim points1(len) As Chart3DPoint
Dim points2(len) As Chart3DPoint
Dim c, s As  Double

For i = 0 To 20
  c = Math.Cos(i * Math.PI * 2 / len)
  s = Math.Sin(i * Math.PI * 2 / len)
  points1(i) = New Chart3DPoint(s, c, 0)
  points2(i) = New Chart3DPoint(s, 0, c)
Next

pointset.AddSeries( points1)
pointset.AddSeries( points2)

' transfer dataset to the chart and select
scatterC1Chart3D1.ChartGroups(0).ChartData.Set = pointset
C1Chart3D1.ChartGroups(0).ChartType = Chart3DTypeEnum.Scatter
```

To write code in C#

C#

```
Chart3DDataSetPoint pointset = new Chart3DdataSetPoint();
int len =21;
Chart3DPoint[] points1 = new Chart3DPoint[len];
Chart3DPoint[] points2 = new Chart3DPoint[len];
double c, s;

for( int i=0; i<len; i++)
{
  c = Math.Cos( i * Math.PI*2 / len);
  s = Math.Sin( i * Math.PI*2 / len);
  points1[i]=new Chart3DPoint(s, c, 0);
  points2[i]=new Chart3DPoint(s, 0, c);
}

pointset.AddSeries( points1);
pointset.AddSeries( points2);

// transfer dataset to the chart and select scatter
C1Chart3D1.ChartGroups[0].ChartData.Set = pointset;
C1Chart3D1.ChartGroups[0].ChartType = Chart3DTypeEnum.Scatter;
```

The following sets up a new array of data points to the first series:

To write code in Visual Basic

Visual Basic

```
Dim pts(1) As Chart3DPoint
pts(0) = New Chart3DPoint( 0, 0, -1)
pts(1) = New Chart3DPoint( 0, 0, 1)
```

```
pointset.SeriesCollection(0).Points = pts
```

To write code in C#

C#

```
Chart3DPoint[] pts = new Chart3DPoint[2];
pts[0] = new Chart3DPoint( 0, 0, -1);
pts[1] = new Chart3DPoint( 0, 0, 1);
pointset.SeriesCollection[0].Points = pts;
```

Data Points

The coordinates of specific points are accessible via the properties of their respective **Chart3DPoint** structure. To retrieve the **Chart3DPoint** for a coordinate, use the **Item** method of the Chart3DPointSeries. The following code sets the coordinates of the third point of the second series to (1, 1, 1):

To write code in Visual Basic

Visual Basic

```
C1Chart3D1.ChartGroups(0).ChartData.SetPoint(1,2) = New Chart3DPoint(1,1,1)
```

To write code in C#

C#

```
C1Chart3D1.ChartGroups[0].ChartData.SetPoint[1,2] = new Chart3DPoint(1,1,1);
```

Loading Data from a File

A common task in any **C1Chart3D** program is to load the chart data (called a *dataset*) from a file into a format that the chart can use.

The Chart3DData's LoadDataFromFile method can be used to allocate and load data from a file. The following example loads the data from a .dat file:

To write code in Visual Basic

Visual Basic

```
' Load the data from a file MMF95.DAT
C1Chart3D1.ChartGroups(0).ChartData.LoadDataFromFile("mmf95.dat")
```

To write code in C#

C#

```
// Load the data from a file MMF95.DAT
C1Chart3D1.ChartGroups[0].ChartData.LoadDataFromFile("mmf95.dat");
```

In order for **C1Chart3D** to allocate and load the data from a file, it must be in one of three basic layouts: *Grid*, *Irregular Grid*, or *Point*. Use the Grid format for Surface and Bar charts when the X-coordinates of each point and the Y-coordinates of each point are always the same distance apart. Use the Irregular Grid format for Surface and Bar charts when the X-coordinates of each point or the Y-coordinates of each point are not the same distance apart. Use

the Point format for multiple series of points in scatter plots.

See 3D Chart Data Layouts for more information on Grid, Irregular Grid, and Point layout.

The following topics provide example formats for the Grid, Irregular Grid, and Point data layout.

Grid File Format

The following code shows the format for a Grid layout that has 50 by 30 data points:

```
! Grid has 50 by 30 points
! Holes have a value of 100.0
! Grid increases in X steps of 1.0 and Y steps of 2.0
!Origin of grid is at X = -20, Y = 50
GRID 50 30 100.0 1.0 2.0 -20.0 50.0
! 1500 data values follow, one for each grid point
49.875000 43.765625 38.500000 33.984375 30.12400
26.828125 24.000000 21.235610 48.877940 17.39770
. . .
```

Line 5 indicates that the file format is Grid, and the file contains 50 by 30 data points. Line 6 indicates the hole value (100.0), the X and Y increments (1.0 and 2.0), and the coordinates for the origin.

Note: Lines beginning with a "!" are comments and are ignored by the control.

Irregular Grid Format

The following code shows the format for an Irregular Grid layout that has 10 by 5 points:

```
! Irregular grid has 10 by 5 points
! Holes have a value of 100.0
! Ten X values are given, and then five Y values
IRGRID 10 5
100.0
20 21.1 23.4 24.4 25.0 27.8 29.9 31.0 32.6 33.2
50.3 51.3 52.6 54.8 59.6
! 50 data points follow
23.34563 12.89239 11.99423 15.781212 18.81988
. . .
```

Line 4 indicates that the file is in the Irregular Grid format, and has 10 by 5 points. Line 5 shows the hole value (100.0). Line 6 contains the 10 X values, and line 7 contains the 5 Y values.

Note: Lines beginning with a "!" are comments and are ignored by the control.

Point File Format

The following code shows the format for a point data layout that has 2 series:

```
! Point has 2 series.
! Holes have a value of 100.0
POINT 2
100.0
! The first series has 5 points
5
```

```
!  X    Y    Z
  5.65 6.24 1.78
  7.41 7.26 4.21
  5.45 5.44 1.43
  0.97 9.66 3.41
  3.86 1.42 0.20
! The second series has 3 points
3
  8.49 -1.28 8.76
  8.14 0.42 6.06
  7.17 -3.80 4.11
```

Line 3 indicates that the file Is in Point format, and has 2 series. Line 6 shows the hole value (100.0). Line 8 indicates that the first series has 5 points. Lines 10-14 contain the X, Y, and Z-coordinates of the 5 points in the first series. Line 16 indicates that the second series has 3 points, and the three following lines specify the coordinates of the 3 points.

Note: Lines beginning with a "!" are comments and are ignored by the control.

Charting Data from a Mathematical Calculation

Often it may not be convenient to load chart data from a file. For example, if the data is created from within the program using a mathematical calculation, it makes more sense to read it into the chart directly instead of saving it to a file and then reading it again. Not only can this method be quicker, it can also be quite easy to program.

The following example creates a regular grid dataset using a precalculated two-dimensional array and then transfers it to the chart:

To write code in Visual Basic

Visual Basic

```
' Calculate array
Dim Rnd As Random = New Random()
Dim m,n As Integer
Dim z(20,30) As Double
For m = 0 To 30
  For n = 0 To 20
   If m Mod 2 = 0 Then
     Z(n, m) = m * 5 * Rnd.NextDouble() + Math.Sqrt(2 * m * m * n)
   Else
     Z(n, m) = m * -5 * Rnd.NextDouble() + Math.Sqrt(2 * m * n)
   End If
  Next n
Next m

' create dataset and put it to the chart
Dim gridset As Chart3DDataSetGrid
gridset=New Chart3DDataSetGrid( 0, 0, 1, 1, z)
C1Chart3D1.ChartGroups(0).ChartData.Set = gridset
```

To write code in C#

C#

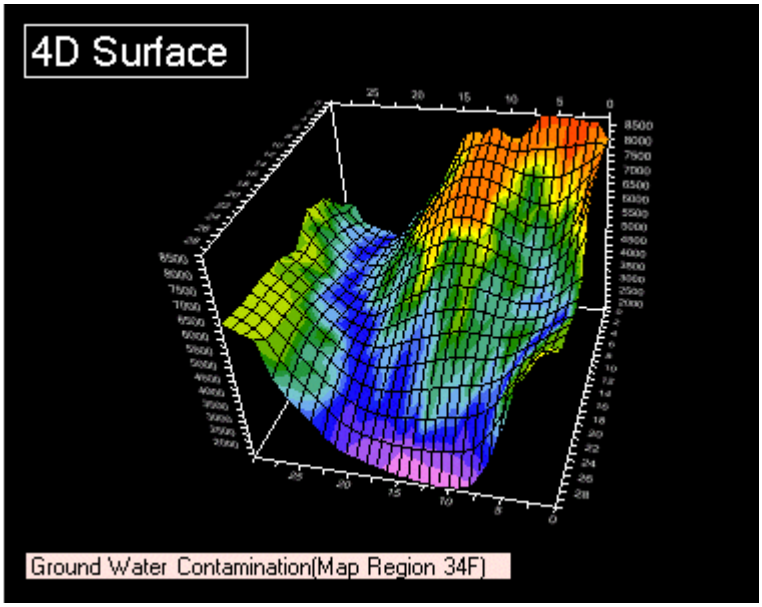
```
// Calculate array
Random rnd = new Random();
double[,] z = new  double[21, 31];
for( int m = 0; m<31; m++)
  for( int n = 0; n<21; n++)
 {
  if(m%2==0)
   z[n,m]= m * 5 * rnd.NextDouble() + Math.Sqrt(2 * m * m * n);
  else
   z[n,m]= m * -5 * rnd.NextDouble() + Math.Sqrt(2 * m * n);
 }

// create dataset and put it to the chart
Chart3DDataSetGrid gridset=new Chart3DDataSetGrid(0,0,1,1,z);
C1Chart3D1.ChartGroups[0].ChartData.Set = gridset;
```

Displaying 4D Data

3D Chart enables developers to represent four-dimensional plots and Bar charts using color as the fourth dimension.

A 4D chart uses color to display a fourth dimension of data on a 3D Surface or Bar chart. The colors come from a second set of data that the chart uses for the zoning and contouring colors. 4D charts are useful for visualizing two sets of identically sized data in one chart.



Creating 4D Charts

To create a 4D chart, complete the following steps:

1. Add the grid dataset (regular or irregular) data to the chart. See 3D Data for information on adding data to a chart.
2. Add the contour data to the chart. This data will be used to derive the zoning and contouring colors for the fourth dimension of chart information. The contour data is defines by **ContourData** property of Chart3DData object.

3. Set the IsShaded and/or IsZoned Contour object properties of Chart3DGroup.

To write code in Visual Basic

Visual Basic

```
' 1 step
' Create dataset with sample data
Dim rnd As Random = New Random()
Dim i, j As Integer
Dim z(20, 20) As Double
For i = 0 To 20
  For j = 0 To 20
    z(i,j) = 200 - ((i - 10) * (i - 10) + (j - 10) * (j - 10))
  Next j
Next i
Dim gridset As Chart3DDataSetGrid = New Chart3DDataSetGrid(0, 0, 1, 1, z)
C1Chart3D1.ChartGroups(0).ChartData.Set = gridset

' 2 step
' Create 4-dimension values array and put it to the chart
Dim contour(20, 20) As Double
For i = 0 To 20
  For j = 0 To 20
    contour(i, j) = rnd.NextDouble()
  Next j
Next i
C1Chart3D1.ChartGroups(0).ChartData.ContourData = contour

' 3 step
' Set zoned chart
C1Chart3D1.ChartGroups(0).Contour.IsZoned = True
C1Chart3D1.ChartGroups(0).Contour.NumLevels = 20
C1Chart3D1.ChartGroups.ContourStyles.ColorSet = ColorSetEnum.RevRainbow
```

To write code in C#

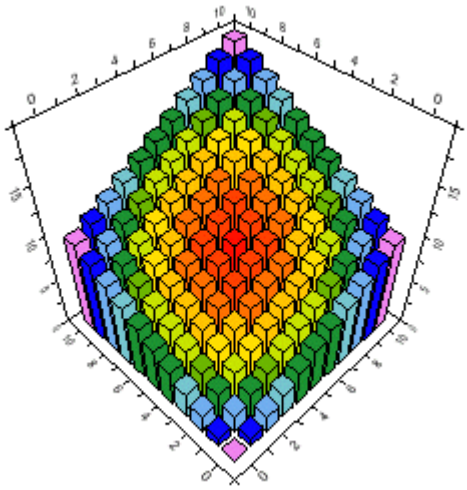
C#

```
// 1 step
// Create dataset with sample data
Random rnd = new Random();
int i, j;
double[,] z = new double[21, 21];
for( i=0; i<z.GetLength(0); i++)
{
  for( j=0; j<z.GetLength(1); j++)
  {
    z[ i, j] = 200 - (( i-10)*(i-10) +  (j-10)*(j-10));
  }
}
Chart3DDataSetGrid gridset=new Chart3DDataSetGrid(0,0,1,1,z);
C1Chart3D1.ChartGroups[0].ChartData.Set = gridset;
```

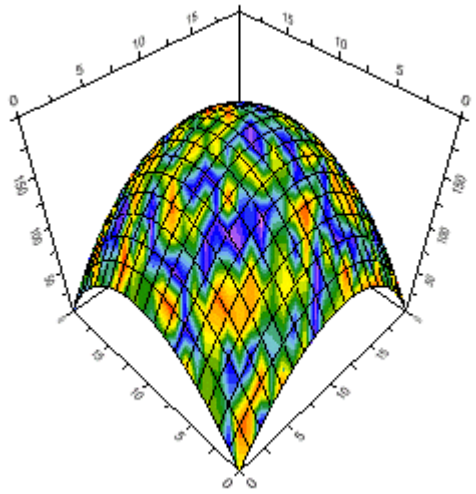
```
// 2 step
// Create 4-dimension values array and put it to the chart
double[,] contour = new double[21, 21];
for( i=0; i<contour.GetLength(0); i++)
{
  for( j=0; j<contour.GetLength(1); j++)
  {
    contour[ i, j] = rnd.NextDouble();
  }
}
C1Chart3D1.ChartGroups[0].ChartData.ContourData = contour;

// 3 step
// Set zoned chart
C1Chart3D1.ChartGroups[0].Contour.IsZoned = true;
C1Chart3D1.ChartGroups[0].Contour.NumLevels = 20;
C1Chart3D1.ChartGroups.ContourStyles.ColorSet = ColorSetEnum.RevRainbow;
```

In Bar charts, each bar is displayed as a single solid color according to the zoned height of the bar:



In Surface charts, the fourth dimension is displayed as zoning and contouring colors.

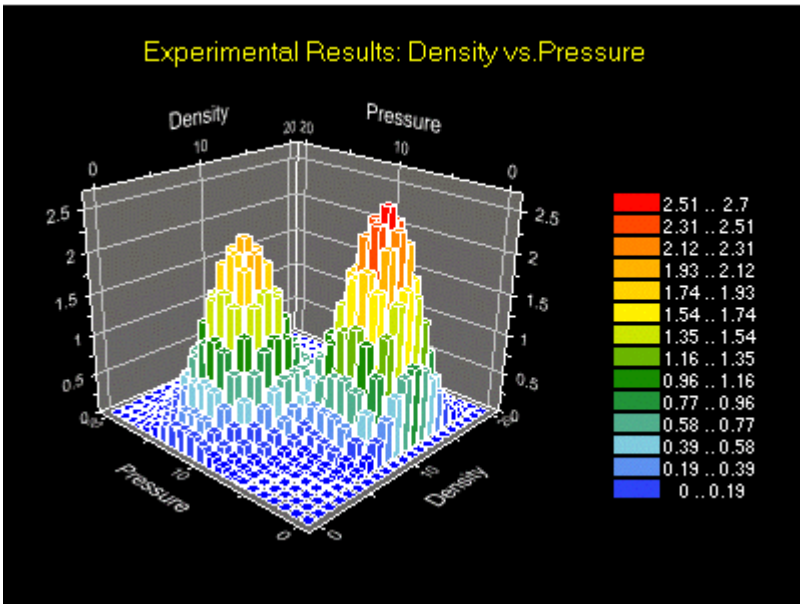


Note: The grid dataset and contour data array must have the same dimensions. If these conditions are not met, a 4D chart is not displayed.

Using Contour Data with 4D Bars Charts

When you are using contour data with 4D Bar chart, the following occurs:

- When a set of contour data is added to a Bar chart and the IsZoned Contour property is set, each bar is drawn with one color (determined by the zoned height of the bar) instead of being drawn as separate colored segments.
- Contours are never drawn when contour data is added.



3D Labels

A Chart3DLabel is an independent label that can be displayed inside or outside the PlotCube. ChartLabels are used to highlight an important data point, but can also be used generally to provide information on data or on the chart.

There is no limit to the number of ChartLabels a chart can contain. Each Chart3DLabel has a foreground and background color, border, fonts, and attachment attributes that can be customized.

The Chart3DLabel object provides a number of properties that help define and position the chart label. The most important of these properties are the following:

- The Text property specifies the text to appear in the chart label, and is of type Label.
- The View3D property specifies the position of the chart label in 3D space. The position is specified as LabelView3DEnum.
- The Connected property is a Boolean that specifies whether a line is to be drawn from the chart label to its attached location. If **True**, the line is drawn.

The following code sets the above properties:

To write code in Visual Basic

Visual Basic

```
With C1Chart3D1.ChartLabels(1)
  .Text = "Here is my chart label"
  .View3D = LabelView3DEnum.YZ
  .Connected = True
End With
```

To write code in C#

C#

```
Chart3DLabel lab = C1Chart3D1.ChartLabels[1];
lab.Text = "Here is my chart label";
lab.View3D = LabelView3DEnum.YZ;
lab.Connected = true;
```

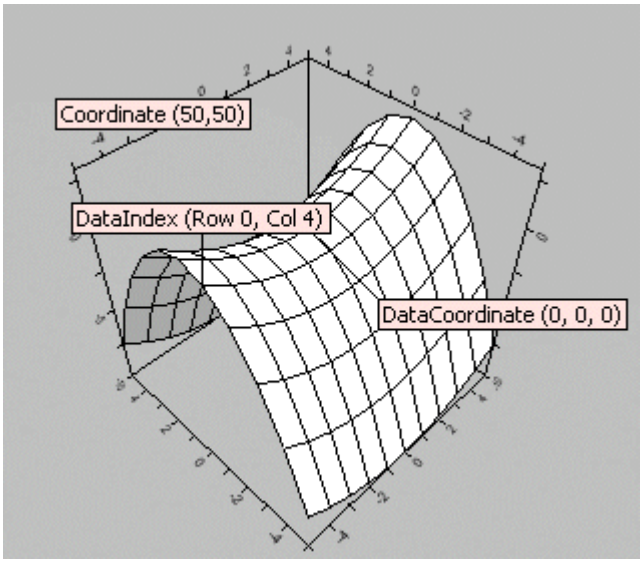
Attaching and Positioning 3D Chart Labels

When defining a 3D Chart label, specify how and where to attach it to the chart.

The attachment method determines the basic appearance of the Chart3DLabel. The best method depends on what the Chart3DLabel is used for. **C1Chart3D** includes the following types of attachment methods:

- **Coordinates** attaches the label outside the PlotCube and it can appear anywhere on the chart. The number of pixels from the top-left corner of the chart to the center of the Chart3DLabel can be specified. When attached this way, customizing the facing plane and cube font has no effect.
- **DataCoordinates** attaches the label inside the PlotCube to any point in 3D space within the data range. Specify the data X, Y, and Z-coordinates. The ChartLabels must fall within the maximum and minimum data coordinates for the axes, otherwise the label is not drawn.
- **DataIndex** attaches the label inside the PlotCube to a specific data point on the chart. Specify the row and column indices.

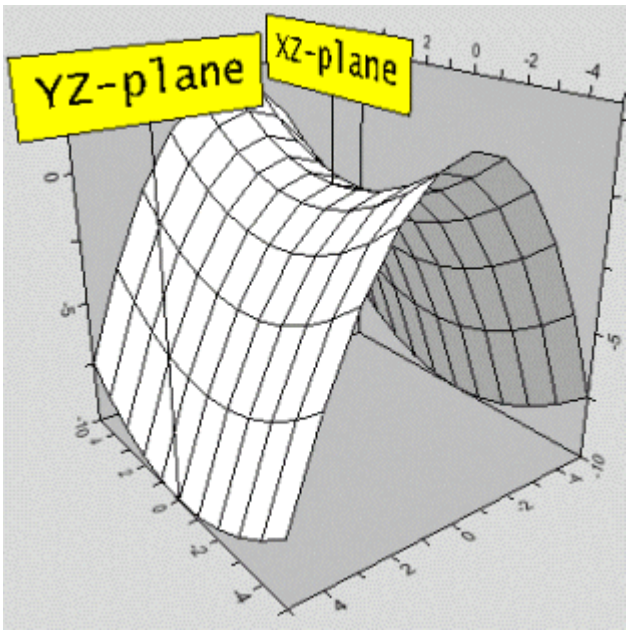
The following chart illustrates the attachment methods:



Use the AttachMethod of the Chart3DLabel property to set the attachment method, and the properties of Chart3DAttachMethodData class to set the attachment point. At design time, these properties are located under **AttachMethodData** node in the **Chart3DLabel Collection Editor**.

Determining which 3D Plane to Face the Label

When attached by **DataCoordinates** or **DataIndex**, use the View3D property to set which plane - **None**, **XZ**, or **YZ** to face the Chart3DLabel. View3D is located in the **Chart3DLabel Collection Editor**. For more information on how to access the **Chart3DLabel Collection Editor**, see Chart3DLabel Collection Editor.



3D Chart Labels Programming Considerations

A Chart3DLabel object defines an independent rectangular region that can be attached to a chart. The Chart3DLabels collection contains all the chart labels defined for a particular chart.

Specify the index number in the Chart3DLabels collection in order to access a ChartLabel individually:

To write code in Visual Basic

```
Visual Basic
C1Chart3D1.ChartLabels(0).Text = "This is the first label in the collection"
```

To write code in C#

```
C#
C1Chart3D1.ChartLabels[0].Text = "This is the first label in the collection";
```

To create a Chart3DLabel object, call the AddNewLabel method:

To write code in Visual Basic

```
Visual Basic
Dim label As Chart3DLabel
label = C1Chart3D1.ChartLabels.AddNewLabel()
```

To write code in C#

```
C#
Chart3DLabel label;
label = C1Chart3D1.ChartLabels.AddNewLabel();
```

The **Remove** method removes a Chart3DLabel from the collection:

To write code in Visual Basic

```
Visual Basic
C1Chart3D1.ChartLabels.Remove(label)
```

To write code in C#

```
C#
C1Chart3D1.ChartLabels.Remove(label);
```

Customizing 3D Chart Labels

The Chart3DLabels object has a DefaultStyle that controls attributes of all labels (font, colors, border). Each of the labels also has its own individual Style inherited from this Default style. Setting attributes of the individual style overrides the attributes of the DefaultStyle.

For instance, in the following code the background color of the default style can be set to gray:

To write code in Visual Basic

```
Visual Basic
C1Chart3D1.ChartLabels.DefaultStyle.BackColor = Drawing.Color.Gray
```

To write code in C#

```
C#
```

```
C1Chart3D1.ChartLabels.DefaultStyle.BackColor = Drawing.Color.Gray;
```

All existing labels will have a gray background (if not deliberately set otherwise), and all new labels created will also adopt a gray background.

3D Chart Label Connecting Lines and Offset

Use the Connected property to draw a line connecting the Chart3DLabel to its attachment point. Connected is located in the **Chart3DLabel Collection Editor**. Use the Offset property to set the distance between the ChartLabel and its attachment point. This property is also located in the **Chart3DLabel Collection Editor**.

3D Chart Label Text and Position

Use the Text property of the Chart3DLabel class to set or change a Chart3DLabel's text. The Text property can be accessed at design time through the **Chart3DLabel Collection Editor**.

Use the LabelCompass property to specify label position relative to the attachment point. For instance, if the label is to be to the left of the point specified, setting this property to East will deliver the desired effect. LabelCompass can be accessed at design time in the **Chart3DLabel Collection Editor**.

3D Chart Label Border

When View3D is set to **XZ** or **YZ**, the border is a fixed thin line and cannot be customized. WhenView3D is set to none, use the **Type** and **Width** border properties to customize the Chart3DLabel's border. These properties can be accessed under the Style node of the **Chart3DLabel Collection Editor**. See 3D Chart Borders for more information.

3D Chart Label Colors

Use the **ForeColor** and **BackColor** properties to customize background and text colors of a Chart3DLabel. These properties can be accessed under the **Style** node in **Chart3DLabel Collection Editor**. See 3D Chart Colors for more information.

3D Chart Label Fonts

When View3D is set to XZ or YZ, the label font is scalable and its size is changing with size of PlotCube. When View3D is set to none, the label font is non-scalable and its size is fixed. The font properties can be accessed under the **Style** node in the **Chart3DLabel Collection Editor**. See 3D Chart Fonts for more information.

3D Chart Elements

When the chart data and axes are formatted properly, its elements can be customized to make it look clearer and more professional. The following topics cover the elements that can be used to customize the appearance of the 3D Charts.

3D Chart Titles

A chart can have two titles, one called the Header and one called the Footer. A title consists of one or more lines of text with an optional border. Both the title and border can be customized. Because each title can be positioned above, below, and to the right or left of the chart, they do not adhere to the traditional concept of Header and Footer at the top and bottom of an object. In addition, the text alignment, positioning, colors, and font used for the Header or Footer can be modified.

Title Text and Alignment

Use the Text property of the Title class to add, change, or remove text for a title. Text is located in the Properties window under the corresponding title node (Footer and Header).

Use the **HorizontalAlignment** and **VerticalAlignment** of the Title's Style property to specify whether to center, left-justify, or right-justify. These properties can be accessed through the **Style** node in either the Header or Footer node in the Visual Studio Properties window.

Title Border

Use the BorderStyle and Thickness border properties to customize the title's border style and width. These properties can be accessed through the **Style** node in either the **Header** or **Footer** node in the **Visual Studio Properties** window. See 3D Chart Borders for more information.

Title Positioning

Use the Compass property of the Title class to specify where to position the Titles relative to the ChartArea. Select from four compass points around the ChartArea.

Use the **X** and **Y** location properties to customize the location of the title. To restore auto location, set left or top to -1.

Use the **Width** and **Height** properties to customize the size of the title. To restore auto size selection set width or height to -1. These properties can be accessed through the **Size** node in either the Header or Footer node in the Visual Studio Properties window.

Title Colors

Use the **ForeColor** and **BackColor** properties to customize background and text colors of a title. These are located under the **Style** node of the Properties window. See 3D Chart Colors for more information.

Title Font

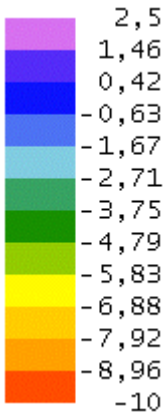
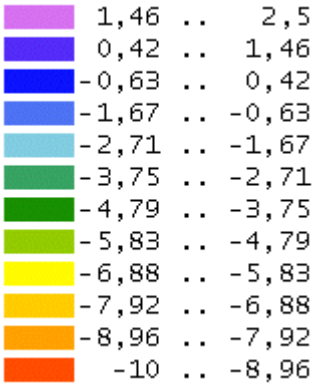
Use the font properties to customize the font used for a title. These are located on the **Style** node of Properties window. See 3D Chart Fonts for more information.

3D Chart Legend

The chart automatically generates a Legend whenever contours or zones are drawn. When using contours only, the Legend shows the value represented by each contour line. When using zones, the Legend shows the value represented by each zone level. The layout, positioning, border, colors and font used for the Legend can be customized through Chart3DLegend properties.

Legend Types and Orientation

Use the Type property of the Chart3DLegend class to specify whether to draw the legend in a continuous or stepped style. Continuous legends take less space on the screen than stepped. The Type property can be accessed through the **Legend** node of the Visual Studio Properties window.

Continuous Legend	Stepped Legend
	

Legend Positioning

Use the legend's Compass property to specify where to position the legend relative to the ChartArea. Select from four compass points around the ChartArea. Compass can be accessed under the Legend node of the Visual Studio Properties window.

By default the chart automatically positions the legend. Use the **X** and **Y** location properties to fine-tune the positioning. These properties can be accessed through the **Location** node that is under the **Legend** node of the Visual Studio Properties window.

By default the chart automatically calculates size of the legend. Use the **Width** and **Height** size properties to fine-tune the size of legend. These properties can be accessed under the **Size** node that is under the **Legend** node of the Visual Studio Properties window.

Legend Title

Use the **Chart3DLegend.Text** property to specify the legend title. The legend title appears centered at the top of the Legend. The **Chart3DLegend.Text** property can be accessed at design time under the **Legend** node in the Visual Studio Properties window.

Legend Border

Use the BorderStyle, Rounding, Color, and Thickness properties to customize the Legend's border style, color, rounding, and width. These properties can be accessed under the **LegendStyle** node that is in the **Visual Studio Properties** window. See 3D Chart Borders for more information.

Legend Colors

Use the **BackColor** and **ForeColor** properties to customize background and text colors of the Legend. These can be accessed through the **LegendStyle** node of the **Visual Studio Properties** window. See 3D Chart Colors for more information.

Legend Font

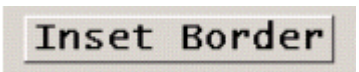
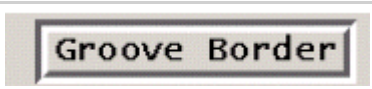
Use the font properties to customize the font used for the Legend. These can be accessed through the **LegendStyle** node of the **Visual Studio Properties** window. See 3D Chart Fonts for more information.

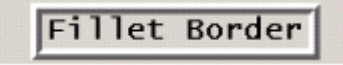
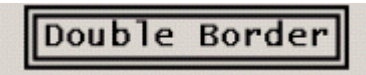
3D Chart Borders

Adding a border to part of the chart can help highlight important information, or simply make the chart look more attractive. The border style, color, rounding, and width can be set using the respective properties: BorderStyle, Color, Rounding, and Thickness. Using these properties you can create customized borders for any of the following chart elements:

- Header and Footer titles
- Legend
- ChartArea
- The entire chart

The following table defines and illustrates the effect of each value in the BorderStyleEnum:

Member name	Description	Effect
NotSet	Border style is not set and is inherited from C1Chart class.	
None	No border.	
Empty	Empty border.	
Solid	Solid line border.	
Raised	Raised 3D border, drawn using system colors.	
Inset	Inset 3D border with bevel.	
RaisedBevel	Raised 3D border with bevel.	
InsetBevel	Inset 3D border with bevel.	
Groove	Compound border (inset+raised).	

Member name	Description	Effect
Fillet	Compound border (raised+inset).	
Double	Double solid line border.	
Dashed	Dashed line border.	
Opaque	The opaque border style ensures that anti-aliasing is turned off when drawing the border. Opaque borders ignore **Rounding** settings. This border style can be useful for generating chart images to be use with transparent backgrounds.	

The border properties can be modified at design time using the Chart Properties, Properties window, or Chart Smart Designer. These properties are located under the **Style** nodes in the Visual Studio Properties window, which can be found on the Control, ChartArea, Header and Footer, Legend, and ChartLabels objects.

To change the border style of the ChartArea at design time

To change the border style of the ChartArea element at design time using the Visual Studio Properties window, complete the following:

1. Expand the **ChartArea** node in the c1Chart3D properties window.
2. Expand the **Style - Border** property.
3. Click on the dropdown arrow next to the **BorderStyle** property and select a border style, for example **Dashed**.

To change the border style of the ChartArea programatically

To write code in Visual Basic

```
Visual Basic
c1Chart3D1.ChartArea.Style.Border.BorderStyle =
C1.Win.C1Chart3D.BorderStyleEnum.Dashed
```

To write code in C#

```
C#
c1Chart3D1.ChartArea.Style.Border.BorderStyle =
C1.Win.C1Chart3D.BorderStyleEnum.Dashed;
```

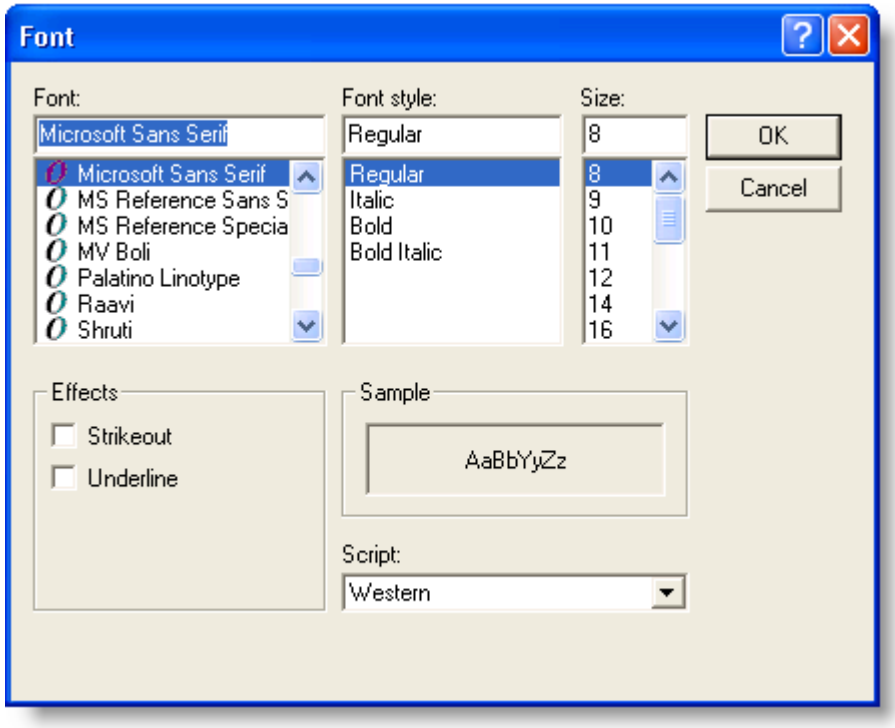
3D Chart Fonts

A chart can have more impact when customized fonts are used for different chart elements. The font size can be adjusted to make an element better fit the overall size of the chart. The chart uses two kinds of fonts sizing: non-scalable fonts and scalable fonts. Non-scalable fonts are always the same size. The scalable fonts is scaling when PlotCube size is changing.

The following table lists the type of font used by each chart element:

Chart Element	Font Type
Axis Annotation	Scalable
Axis Title	Scalable
Footer Title	Non-Scalable
Header Title	Non-Scalable
Legend	Non-Scalable

Use the standard .NET Font property editor to set the font, style, and size attributes. Font properties are located under the Stylenodes of the Visual Studio Properties window.



Note: For scalable fonts the **Size** property is measured in hundredths of the PlotCube length, for example, a value of 8 means the characters are 8% of the length of the PlotCube height.

3D Chart Colors

Color can powerfully enhance a chart's visual impact. Colors can be customized using color names, RGB values, or interactively using a color chooser. Each of the following visual elements in the chart has a background and foreground color that can be customized:

- The entire chart
- Header and Footer titles
- Legend
- ChartArea
- PlotCube (background only)
- Each ChartLabel added to the chart

The mesh, contour lines, surface shading, and zone fills also have color properties that can be customized.

Choosing Colors Interactively

Choose colors interactively using a color dialog box that works just like the standard Windows color dialog box. Select from Windows basic colors, custom colors, or interactively choose from a full color spectrum.

Specifying RGB Colors

Alternately, a color can be specified by its RGB components, useful for matching another RGB color. RGB color values combine hexadecimal values for the red, green, and blue components of a color. "00" is the smallest value a component can have; "ff" is the largest value. For example, "#ff00ff" specifies magenta (the maximum value of red and blue combined with no green).

Using Transparent Colors

The background and foreground of all elements except the chart itself can be "Transparent".

When a background or foreground is transparent, the chart uses the color of the element outside it for the background. For example, the Header would have the background of the chart itself when its background is set to Transparent.

In other words, if the background color of the element is transparent then its background is not drawn. If the foreground color of the element is transparent, then the foreground (for example the text of a title) is not drawn.

The transparent color options are on the **Web** tab of the .NET color dialog box.

Changing Colors

Use the **BackgroundColor** and **ForegroundColor** properties to set background and foreground colors. These properties are located under **Style** node, found on the **Control**, **Titles**, **Legend**, **ChartArea**, **PlotCube**, and **ChartLabels** nodes of the Visual Studio Properties window.

3D Chart Surface Appearance

The shaded surface (drawn when the **IsShaded** property of the Chart3DElevation class is used) has color and appearance properties that can be customized. The zoning method (drawn when the IsZoned property of the Chart3DContour class is used) can also be customized.

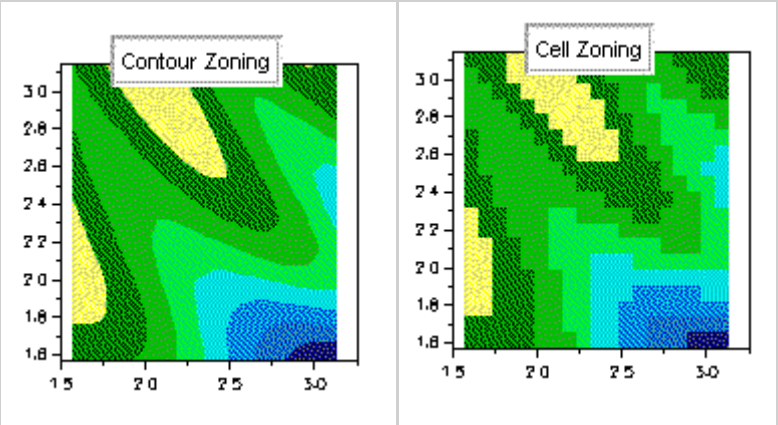
Surface Colors

Use the ShadedBottomColor and ShadedTopColor properties of the Chart3DElevation class to set the shading colors. These properties can be accessed at design time under the **Elevation** node, which is located in the **ChartGroup Collection Editor** in the Visual Studio Properties window.

Zoning Method

Use the ZoneMethod property of the Chart3DContour object to specify how to determine each zone region. Contour zoning uses the contour intervals for each zone region, while cell zoning uses the rectangular block formed by the X/Y grid. Cell zoning produces a coarser-looking surface but redraws significantly faster than contour zoning. **ZoneMethod** can be accessed at design time through the **Contour** node, which is located in the **ChartGroup**

Collection Editor in the Visual Studio Properties window.



3D Mesh Formats

The mesh (drawn when the IsMeshed property of the Chart3DElevation class is used) has display, color, and filtering properties that can be customized. Mesh filtering is particularly useful when displaying a large amount of data in a small space.

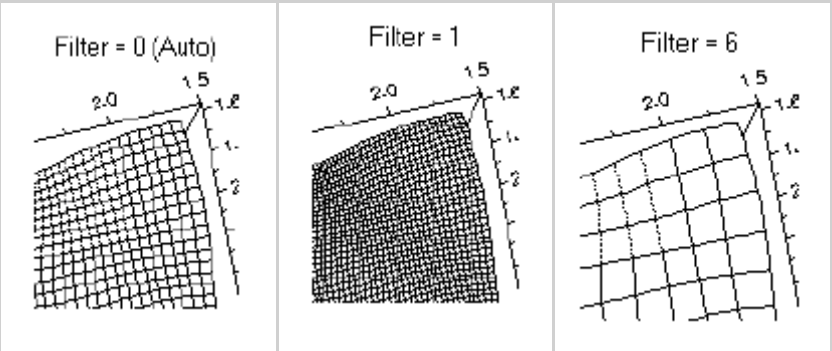
Mesh Filtering

Use the ColumnMeshFilter and RowMeshFilter properties of the Chart3DSurface class to define how to filter the display of mesh lines. The chart can automatically determine the best filter, or specify a positive integer to use.

- When these properties are set to 0, the chart automatically determines the best filter based on the density of the mesh. As the chart data increases or the size of the chart control decreases, the chart performs more mesh filtering.
- When these properties are greater than 0, the chart provides fixed mesh filtering. Higher values cause fewer mesh lines to be drawn. For example, a value of 5 filters the mesh so that every fifth mesh line is drawn.

The ColumnMeshFilter and RowMeshFilter properties can be accessed at design time under the **Surface** node of **ChartGroup Collection Editor**.

The first Surface chart in the following table represents the default value setting, zero, for the ColumnMeshFilter and RowMeshFilter, the second Surface chart represents the ColumnMeshFilter and RowMeshFilter properties set to 1, and the last Surface chart represents the ColumnMeshFilter and RowMeshFilter properties set to 6.



Mesh Display

Use the IsColumnMeshShowing and IsRowMeshShowing properties to display or hide the mesh lines. These properties can be accessed under **Surface** node of the **ChartGroup Collection Editor**.

Mesh Colors

Use the MeshBottomColor and MeshTopColor properties of the Chart3DElevation class to set the mesh colors. These properties can be accessed under the **Elevation** node of the **ChartGroup Collection Editor**.

3D Chart Elements Position and Size

Each of the main chart elements (Header, Footer, Legend, ChartArea, and ChartLabels) has properties that control its position and size. While the chart can automatically control these properties, the following can also be customized:

- Positioning of any element except ChartLabels
- Size of any elements

When the chart controls positioning, it first allows space for the Header, Footer, and Legend, if they exist (size is determined by contents, border, and font). The ChartArea is sized and positioned to fit into the largest remaining rectangular area. Positioning adjusts when other chart properties change.

Changing Location

Use the **X** location property to specify the number of pixels from the edge of the chart to the left edge of the chart element. Use the **Y** location property to specify the number of pixels from the edge of the chart to the top of the chart element. Set **X** and **Y** to -1 to allow the chart to automatically position the element.

These properties are located on the Location node, found on the **ChartArea**, **Titles**, **Legend**, and **ChartLabels** nodes of the Visual Studio Properties window.

Changing Width and Height

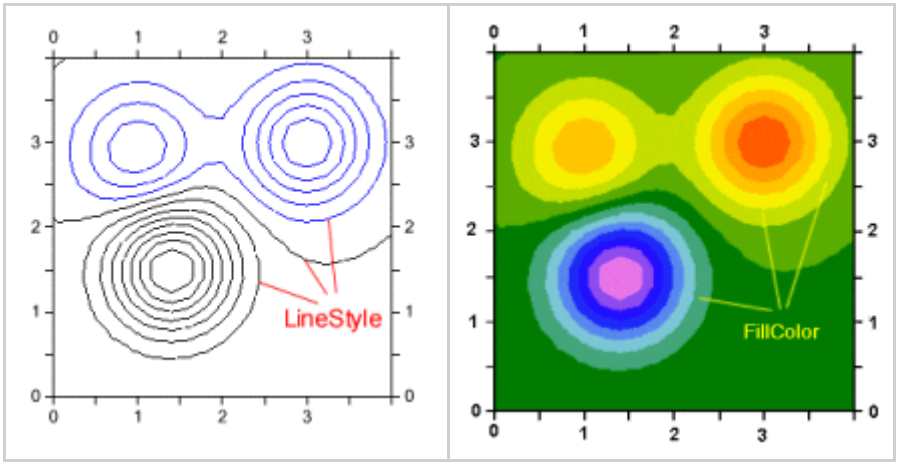
Use the **Width** and **Height** location properties to specify the width and height of the chart elements. Set these properties to -1 to allow the chart to automatically size the chart elements.

These properties are located and enabled under the **Location** nodes of the Visual Studio Properties window.

3D Contour Styles

The attributes that define how contour lines and zoning colors look in the chart are called a ContourStyle. The chart has a built-in set of one hundred styles. A subset of these styles is used for the chart display, based on the number of contour levels in the chart. Individual styles can be customized.

The following two images display the LineStyle for the contour lines and the FillColor for the zoning colors:



Every ContourStyle has a FillColor and a LineStyle. FillColors are used for zoning colors. LineStyles are used for contour lines. Most charts will not need to customize ContourStyles. Situations where customization is needed include:

- To control the precise ContourStyle for any particular level
- To display more than one hundred levels
- To uniquely identify contour lines (the built-in styles use the same LineStyle for every ContourStyle)

Contour Style Appearance Properties

By default, the chart defines an array of contour styles for each chart group. These contour styles are specified as a Chart3DContourStylesCollection of objects.

Each Chart3DContourStyle object in this collection defines two properties that specify the appearance of a particular contour line or zone in a chart:

- The FillColor property specifies the color of the zone associated with this contour style. The FillColor property can be accessed at design time in the **Chart3DStyle Collection Editor** of the ContourStyles node. The **ContourStyles** node can be accessed through the **ChartGroups** node in the Visual Studio Properties window.
- The LineStyle property is a LineStyle object, and specifies the width and color of the contour line associated with this contour style. By default, each line is one pixel wide and is a solid black line. The **LineStyle** property can be accessed at design time in the **Chart3DStyle Collection Editor** of the **ContourStyles** node. The ContourStyles node can be accessed through the **ChartGroups** node in the Visual Studio Properties window.

A total of one hundred contour styles are defined by default. To add additional contour styles, use the Add method.

To access the **ContourStyle** object associated with a particular contour line or zone, use the Style property of Chart3DContourLevel object. For example, the following retrieves the Chart3DContourStyle object for the third contour line:

To write code in Visual Basic

Visual Basic
`Dim cstyle As Chart3DContourStyle`

```
cstyle = C1Chart3D1.ChartGroups(0).Contour.Levels(2).Style
```

To write code in C#

C#

```
Chart3DContourStyle cstyle = C1Chart3D1.ChartGroups[0].Contour.Levels[2].Style;
```

Use this approach to change the attributes of individual zones or contour lines. For example, to change the color for a particular zone, change the **FillColor** property of the **Chart3DContourStyle** object. The following statements change the first zone color to green:

To write code in Visual Basic

Visual Basic

```
C1Chart3D1.ChartGroups(0).Contour.Levels(0).Style.FillColor = Color.Green
```

To write code in C#

C#

```
C1Chart3D1.ChartGroups[0].Contour.Levels[0].Style.FillColor = Color.Green;
```

Contour Styles and Distribution Levels

The *n*th contour line or zone in a chart is normally **not** associated with the *n*th contour style in the Chart3DContourStyles collection. The relationship between distribution levels (contour lines/zones) and the Chart3DContourStyles collection is as follows:

- If *nstyles* contour styles are provided, and the number of distribution levels is *nlevels*, the index of the contour style associated with the *n*th distribution level is the largest integer less than or equal to the following:
  `(n * (nstyles - 1) / nlevels) + 1`

For example, if one hundred contour styles and six distribution levels are defined, the distribution levels use the 1st, 21st, 41st, 60th, 80th, and 100th elements of the Chart3DContourStyles collection, respectively. (The number of distribution levels is specified by the NumLevels property of the Chart3DContour object.)

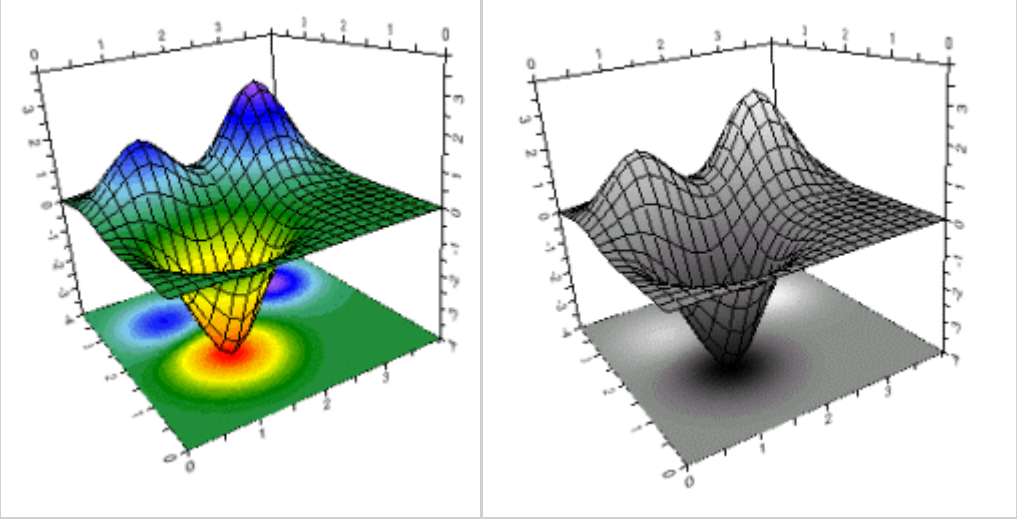
To force a one-to-one match up between contour styles and distribution levels create an array of contour styles with the same number of elements as there are distribution levels. (If it is specified that zone regions be drawn, the array of contour styles must contain one more element than the number of distribution levels. Setting the IsZoned property of the Chart3DContour object to True specifies zone regions. The extra contour style is used to specify the last zone's fill color.)

Contour Styles Fill Colors

The Chart3DContourStyles object has two properties that allow for easy manipulation of the contour styles fill color: the Colors property, and the ColorSet property. When the ColorSet property is set to **Custom** the fill colors for all contour styles is generated automatically using smooth transitions through colors defined in the **Colors** collection property. When the ColorSet property is set to one of the predefined values (Rainbow, RevRainbow, BlackWhite, WhiteBlack) the fill colors are generated using one of these predefined colors collections.

The following table displays the ColorSet values for Rainbow and BlackWhite in the Surface 3D Chart:

ColorSet is Rainbow	ColorSet is BlackWhite



Changing the Contour's Line Thickness and Color

To change the line thickness or color for a contour line, change the Thickness and Color properties of the Chart3DLineStyle object. For example, the following statements change the second contour line to a green line:

To write code in Visual Basic

```
Visual Basic
C1Chart3D1.ChartGroups(0).Contour.Levels(1).Style.LineStyle.Color = Color.Green
```

To write code in C#

```
C#
C1Chart3D1.ChartGroups[0].Contour.Levels[1].Style.LineStyle.Color = Color.Green;
```

To revert to the default contour style behavior, invoke the **Reset** method. All contour styles will be reset to their default values.

Displaying Contours and Zones On the Ceiling or Floor

The contours and zones determined for the chart can be displayed on the ceiling or the floor of the PlotCube. To do this, set the IsContoured and/or the IsZoned property for the ceiling or the floor.

For example, the following statement projects zones onto the ceiling of the PlotCube:

To write code in Visual Basic

```
Visual Basic
C1Chart3D1.ChartArea.View.Ceiling.IsZoned = True
```

To write code in C#

```
C#
C1Chart3D1.ChartArea.View.Ceiling.IsZoned = true;
```

The Chart3DContour object defined for that chart group sets the contour and zone settings used in these projections.

Contour and zone projection are ignored in two-dimensional charts, Scatter, and Bar charts.

Customizing the Distribution Table

The chart's ContourLevels determine how many contour lines and zoning colors are used in the chart, and the Z-values that mark each level. The number of levels or the distribution table itself can be customized.

To generate contours and zones, the chart analyzes the data and based on the number of levels specified, creates the following:

- A linear distribution table that contains the Z-value that marks each level.
- An assignment of ContourStyles from the chart's one hundred built-in styles. The chart determines which styles to use by evenly distributing its one hundred styles through the number of levels.

Customizing the Number of Levels

Use the NumLevels property to set the number of contour levels to use. This has no effect when the chart is using a custom distribution table. You can specify up to one hundred levels. The more levels used, the finer the distribution table. **NumLevels** is located under **Contour** node in the **ChartGroup Collection Editor**. For information on how to access the **ChartGroup Collection Editor**, see Chart3DGroup Collection Editor.

Creating a Custom Distribution Table

When a linear distribution table does not suit the needs of the chart, create a custom table that sets the number of levels and the value used to mark each level. To use custom distribution table, set the AutoDistribution property of Chart3DContour object to **False**.

Resetting to Linear Distribution Table

Use the AutoDistribution property to use linear distribution. Set it to **True** and any customized levels are removed. AutoDistribution is located under **Contour** node in the **ChartGroup Collection Editor**.

Distribution Table Programming Considerations

The Chart3DContourLevel object controls the behavior of the distribution levels used to create contour levels and zones for a chart. One Chart3DContourLevel object is defined for each chart group.

The NumLevels property of the Chart3DContour object specifies the number of distribution levels to use in the chart. Any number between 2 and 100 is valid. If two are specified, no contour levels are drawn, and the entire chart is displayed as one zone.

By default, distribution levels are evenly spaced. To define distribution levels and spacing, use the **Add** method of the Chart3DContourLevel to create them. For example, the following code defines distribution levels at 1000, 2500 and 4200:

To write code in Visual Basic

Visual Basic

```
With C1Chart3D1.ChartGroups(0).Contour.Levels
  .Add( 1000)
```

```
  .Add( 2500)
  .Add( 4200)
End With
```

To write code in C#

```
C#
Chart3DContourLevelsCollection levs;
levs = C1Chart3D1.ChartGroups[0].Contour.Levels;
levs.Add(1000);
levs.Add(2500);
levs.Add(4200);
```

To remove a distribution level, use the **Remove** method:

To write code in Visual Basic

```
Visual Basic
C1Chart3D1.ChartGroups(1).Contour.Levels.Remove( 1000)
```

To write code in C#

```
C#
C1Chart3D1.ChartGroups[1].Contour.Levels.Remove( 1000);
```

At any time, revert to evenly spaced distribution levels by setting the AutoDistribution property to **True**. If this is done, the number of distribution levels in the chart will not change.

3D Chart End-User Interaction

Interact with the chart as it is running to examine data more closely or visually isolate a part of the chart. The interactions described here affect the chart displayed inside the ChartArea; other chart elements like the Header are not affected.

The interactivity of the chart is controlled at run time by the IsInteractive property. The default setting for this property is **True**.

Special Notes

The keyboard/mouse combinations that perform the different interactions can be changed or removed. The interactions described here may not be enabled.

Rotation

Hold down both left mouse button.

To rotate freely, move mouse in the desired direction.

Or, to constrain rotation along an axis, press the X, Y, or Z key and move mouse perpendicular to axis.

Translation

Press SHIFT, hold down the left mouse button.

Move mouse to change the positioning of the chart inside the ChartArea.

Scaling

Press CTRL, hold down left mouse button.

Move mouse down to increase chart size, or up to decrease chart size.

Zooming

Press ALT, hold down left mouse button.

Drag mouse to select zoom area and release mouse button.

Reset to Automatic Scale and Position

Press the R key to remove all rotation, scaling, translation, and zooming effects.

Returning Coordinate Values

The Chart3DData object provides methods that enable the following:

- Determine the pixel coordinates of a given data point, or the closest point to a given set of pixel coordinates.
- Convert from data coordinates to screen pixel coordinates and vice versa.

Determining Coordinate Values

To determine the pixel coordinates of a given data point, call the DataIndexToCoord method. For example, the following code obtains the pixel coordinates of the point in the second row and third column:

To write code in Visual Basic

Visual Basic

```
Dim PixelX, PixelY As Integer
C1Chart3D1.ChartGroups(0).ChartData.DataIndexToCoord(2, 1, PixelX, PixelY)
' PixelX and PixelY now contain the pixel coordinate value.
```

To write code in C#

C#

```
int PixelX=0, PixelY=0;
C1Chart3D1.ChartGroups[0].ChartData.DataIndexToCoord(2,1,ref PixelX,ref PixelY);
// PixelX and PixelY now contain the pixel coordinate value.
```

Converting Data Coordinates to Pixel Coordinates

To convert from data coordinates to screen pixel coordinates, call the DataCoordToCoord method. For example, the following code obtains the pixel coordinates corresponding to the data coordinates (5.1, 10.2, 8.4):

To write code in Visual Basic

Visual Basic

```
Dim PixelX, PixelY As Integer
C1Chart3D1.ChartGroups(0).ChartData. _DataCoordToCoord(5.1, 10.2, 8.4, PixelX,
PixelY)
' PixelX and PixelY now contain the pixel coordinate value.
```

To write code in C#

C#

```
int PixelX=0, PixelY=0;
C1Chart3D1.ChartGroups[0].ChartData.
DataCoordToCoord(5.1, 10.2, 8.4, ref PixelX, ref PixelY);
// PixelX and PixelY now contain the pixel coordinate value.
```

If the data coordinate is out of the visible range of the chart, this method returns false.

Converting Pixel Coordinates to Data Coordinates

To convert from pixel coordinates to data coordinates, call the CoordToDataCoord method. For example, the following converts the pixel coordinates (225, 92) to their equivalent data coordinates:

To write code in Visual Basic

Visual Basic

```
Dim DataX As Double
Dim DataY As Double
Dim DataZ As Double

C1Chart3D1.ChartGroups(0).ChartData._CoordToDataCoord(225, 92, DataX, DataY, DataZ)
' DataX, DataY and DataZ now contain the data coordinate value.
```

To write code in C#

C#

```
double DataX=0, DataY=0, DataZ=0;

C1Chart3D1.ChartGroups[0].ChartData.CoordToDataCoord(225, 92, ref DataX, ref DataY,
ref DataZ);
// DataX, DataY and DataZ now contain the data coordinate value.
```

CoordToDataCoord returns **False** if there is no data corresponding these pixel coordinates.

Determining the Closest Data Point

To determine the closest data point to a set of pixel coordinates, call CoordToDataIndex:

To write code in Visual Basic

Visual Basic

```
Dim Row, Column As Integer

C1Chart3D1.ChartGroups(0).ChartData.CoordToDataIndex( 225, 92, Column, Row)
' CoordToDataIndex returns the row and column of the closest data point.
```

To write code in C#

C#

```
int Row=0, Column=0;

C1Chart3D1.ChartGroups[0].ChartData.CoordToDataIndex(225, 92, ref Column, ref Row);
// CoordToDataIndex returns the row and column of the closest data point.
```

Chart 3D for WinForms Samples

Please be advised that this ComponentOne software tool is accompanied by various sample projects and/or demos which may make use of other development tools included with ComponentOne Studio.

Please refer to the pre-installed product samples through the following path:

Documents\ComponentOne Samples\WinForms

The following table provides a description for each sample.

Visual Basic and C# Samples

The following samples are provided for the **C1Chart3D** control:

Sample	Description
Chart4D	Shows a 4 dimensional chart using Chart3D and color. This sample uses the **C1Chart3D** control.
ChartLoader	Shows various load and save operations for both 2D and 3D charts using XML and allows editing using the Chart Properties dialog box, the Chart Wizard, or a basic Property Grid. After editing, the chart can be saved as XML to files or clipboard, or chart images can be generated and saved. This sample is also a useful utility for manipulating chart XML files at run time and storing the chart state.
CoordMapping3D	Shows each available chart type and a variety of 3D chart options. This sample uses the **C1Chart3D** control.
Demo3D	Shows various 3D charts and explores their properties. This sample uses the **C1Chart3D** control.
Function3D	Shows a 3D surface chart based on various analytical functions. This sample uses the **C1Chart3D** control.
PrintIt2D	Loads both 2D and 3D persisted from and generates images or prints them. This sample uses the **C1Chart** and **C1Chart3D** controls.
RuntimeLocalization	This samples demonstrates the steps to provide runtime localizations for the charts.
Scatter3D	Show a 3D scatter chart and demonstrates various effects. This sample uses the **C1Chart3D** control.

Chart 3D for WinForms Task-Based Help

The task-based help assumes that you are familiar with programming in Visual Studio. By following the steps outlined in the help, you get a better feel of the capabilities of the **C1Chart3D** product through exploring its well designed interface and using its editors. A few of the task-based help topics also show how to access C1Chart3D's properties programmatically. However, most of the topics focus on using C1Chart3D's properties at design time.

Changing the Axis Label Color

To change the axis label color to green, use the following code:

```
c1Chart3D1.ChartArea.Style.ForeColor = System.Drawing.Color.Green;
```

or

```
c1Chart3D1.ChartArea.View.ForeColor = System.Drawing.Color.Green;
```

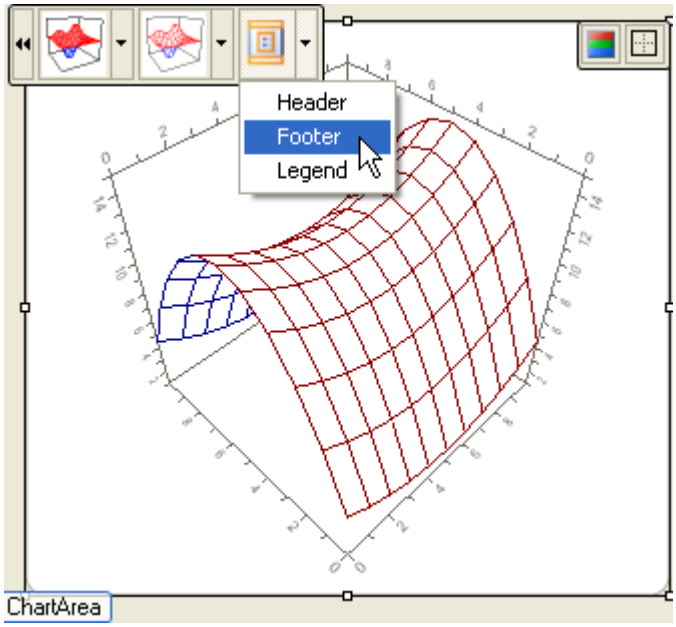
Creating Chart Elements Using the Smart Designer

This section shows how to use the **SmartDesigner** to create and format chart elements, or edit existing elements directly on the form. For example, rather than drilling down through the C1Chart3D's objects in the Properties window to set properties for each of the chart elements you can simply modify the chart elements directly on the form. A simple or complex chart can be created without using any code. It can all be done at design time through the use of C1Chart3D's editors.

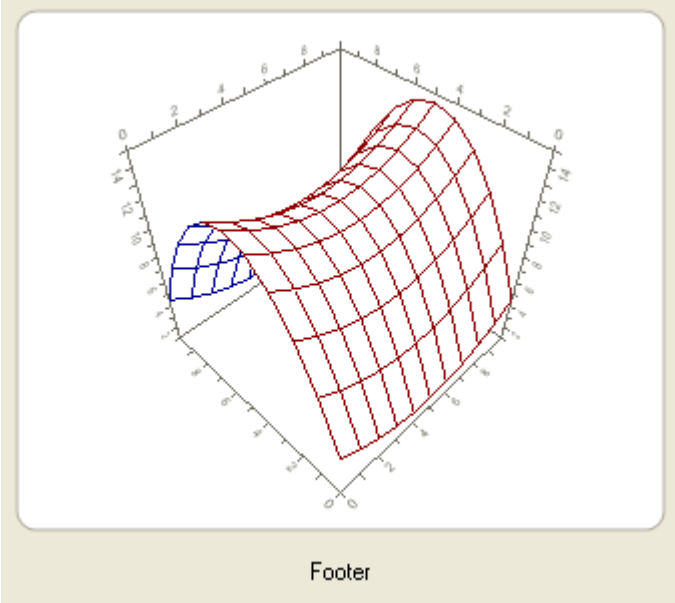
Add a Chart Footer

To add a 3D Chart Footer using the **C1Chart3D** toolbar, complete the following steps:

1. Select the **C1Chart3D** control and click on the  open button to open the **C1Chart3D** toolbar if it is not already open.
2. Select the drop-down arrow from the **C1Chart3D** toolbar drop-down menu and choose the **Footer** item.



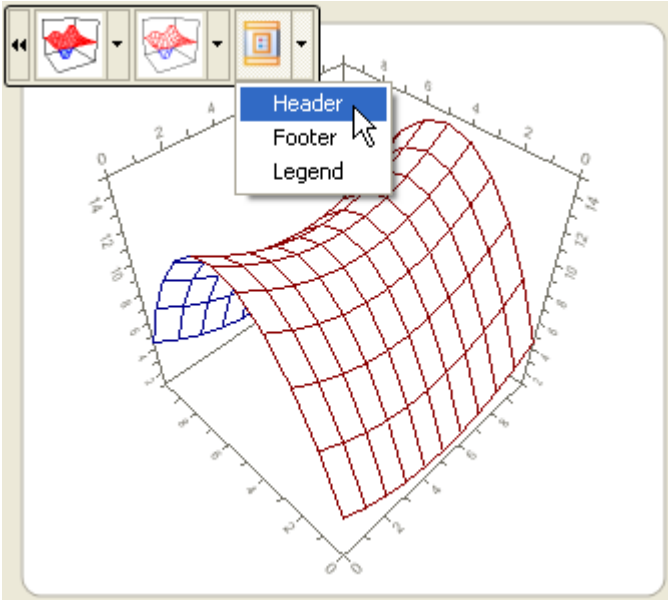
The **Footer** element appears below the chart area. This is the default position for the chart **Footer** element.



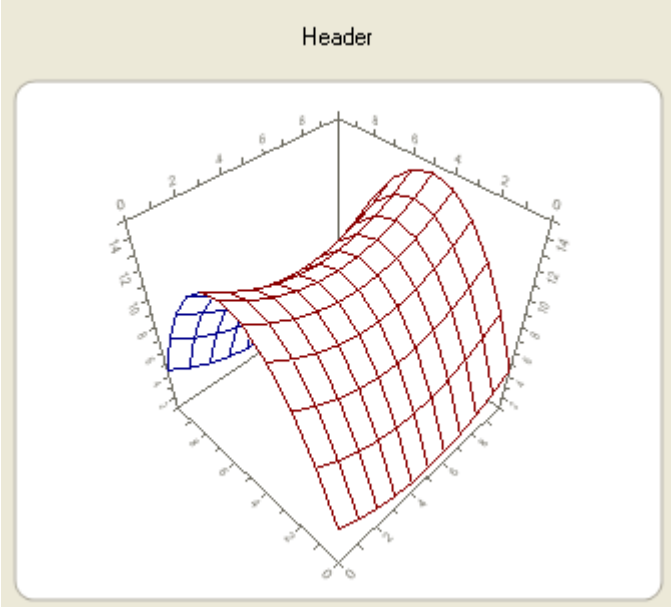
Add a Chart Header

To add a 3D Chart header using the **C1Chart3D** toolbar, complete the following steps:

1. Select the **C1Chart3D** control and click on the open button to open the **C1Chart3D** toolbar if it is not already open.
2. Select the drop-down arrow from the **C1Chart3D** toolbar drop-down menu and choose the Header item.



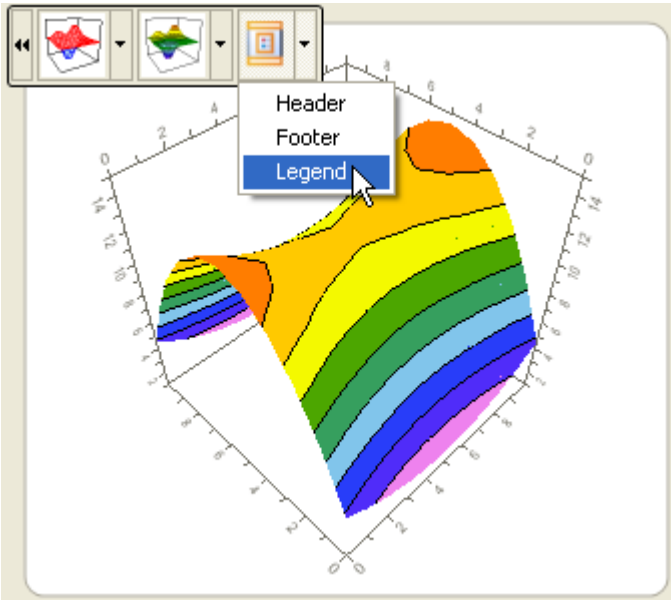
The Header appears above the chart area. This is the default position for the chart Header element.



Add a Chart Legend

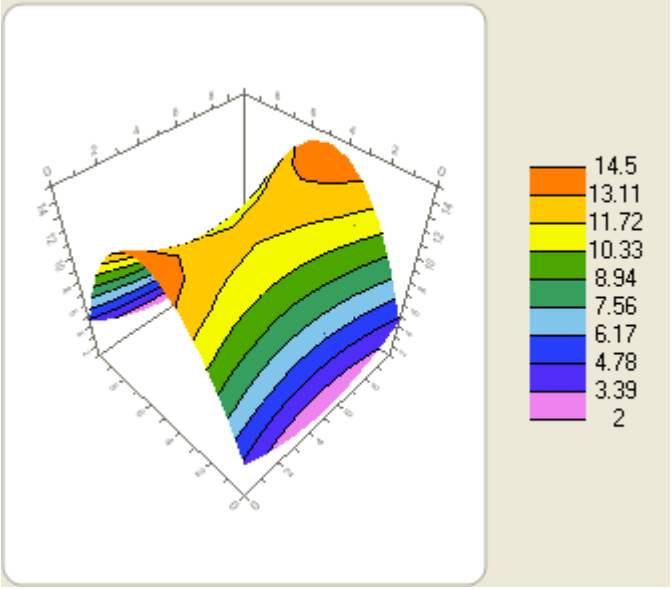
To add a chart **Legend** to the 3D Chart, complete the following steps:

1. Select the **C1Chart3D** control and click on the open button to open the **C1Chart3D** toolbar if it is not already open.
2. Select the drop-down arrow from the **C1Chart3D** toolbar drop-down menu and choose the **Legend** item.



Note: For the Legend to appear, you must have contours and lines in your chart since the legend shows contours and lines.

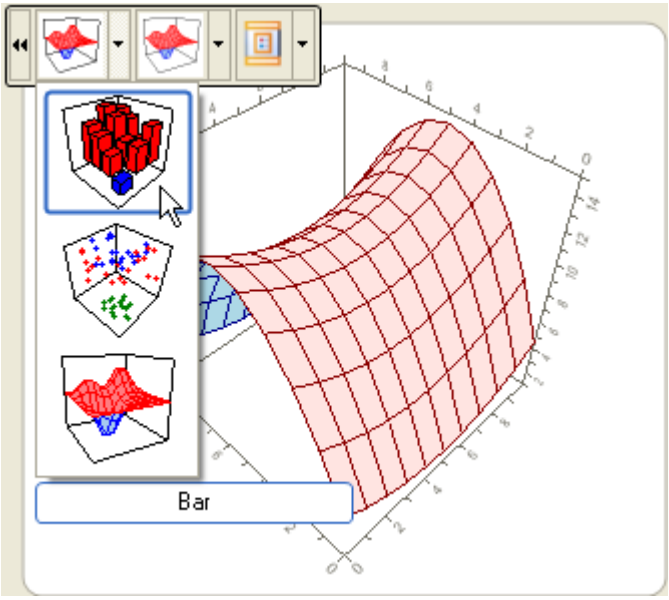
The **Legend** element appears to the right of the chart area or to the east of the chart area. This is the default position for the chart Legend element.



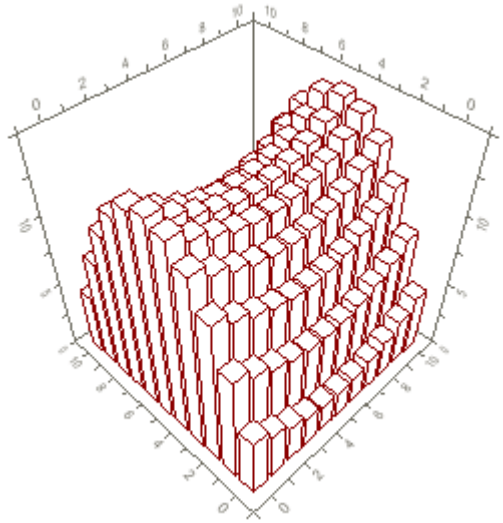
Choose a 3D Chart Type

To select a chart type through the **C1Chart3D** toolbar, complete the following steps:

1. Select the **C1Chart3D** control and click on the open button to open the **C1Chart3D** toolbar.
2. Select the drop-down arrow from the **Chart type** button and choose the Line chart type.



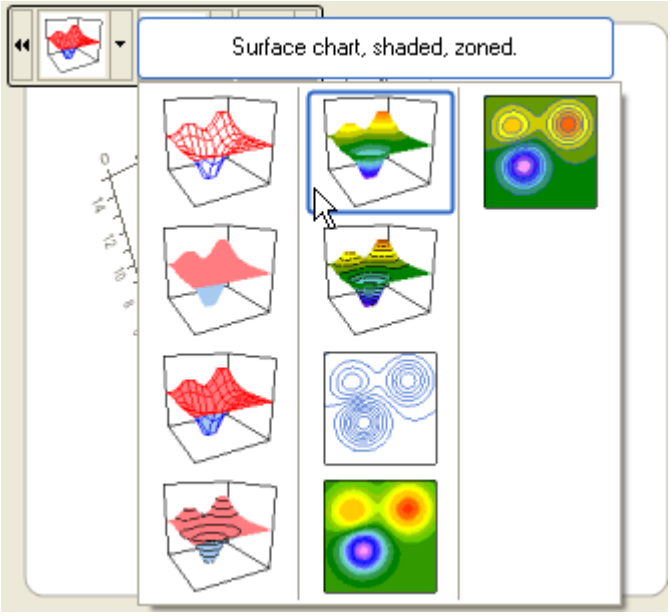
The Bar chart type appears on the **C1Chart3D** control.



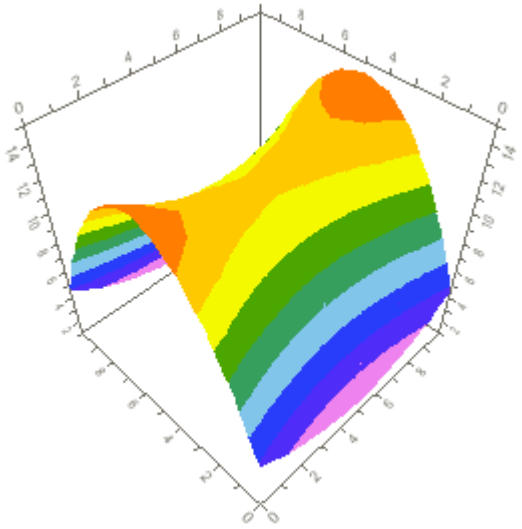
Choose a Chart sub-type

To select a chart sub-type through the **C1Chart** toolbar, complete the following steps:

1. Select the **C1Chart3D** control and click on the open button to open the **C1Chart3D** toolbar if it is not already open.
2. Select the drop-down arrow from the **Chart sub-type** button and choose the Surface chart, shaded, zoned chart sub-type.



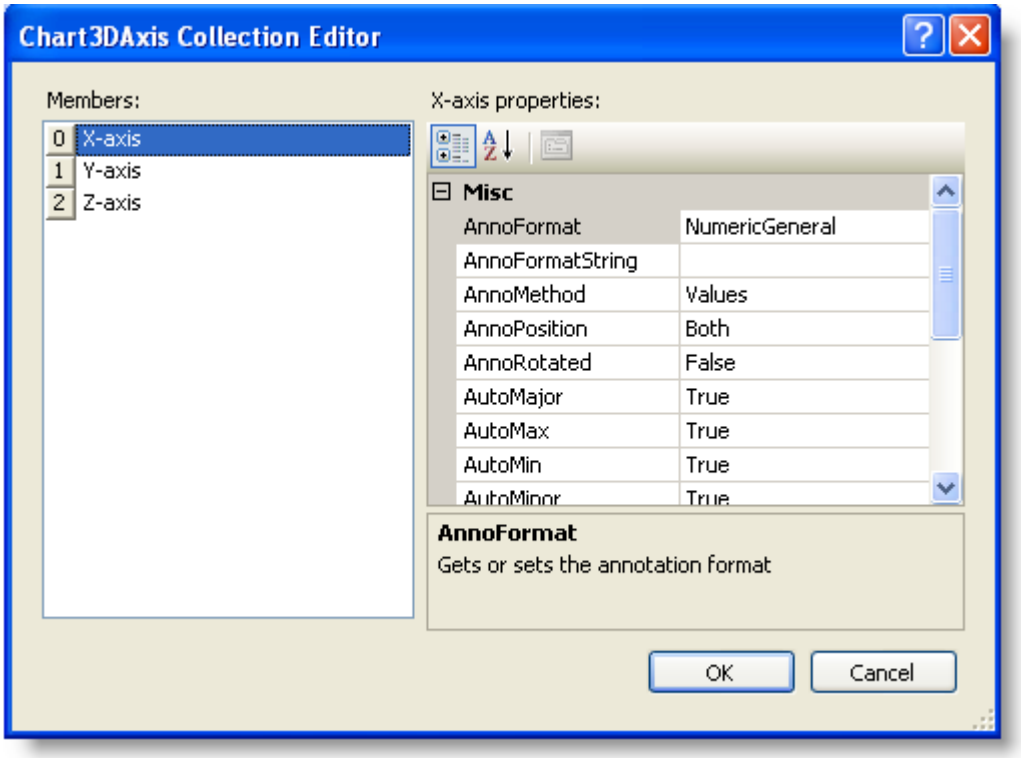
The Surface chart, shaded, zoned chart sub-type appears on the chart control.



Accessing Chart3DAxis Collection Editor

To access the **Chart3DAxis Collection Editor**, complete the following steps:

1. In the Properties window, expand the **ChartArea** node.
2. Press the **ellipsis** next to the **Axes** node to display the **Chart3DAxis Collection Editor**.



3. Modify the properties as desired. For more information, see 3D Axes.

Accessing ChartGroups

ChartGroups are organized into the ChartGroupsCollection, which is accessed through the ChartGroups object.

To access ChartGroups programmatically:

To write code in Visual Basic

Visual Basic
`c1Chart3D1.ChartGroups.ChartGroupsCollection[0]`

To write code in C#

C#
`c1Chart3D1.ChartGroups.ChartGroupsCollection[0];`

Also, as with the 2D chart, the access path can be shortened.

To write code in Visual Basic

Visual Basic
`c1Chart3D1.ChartGroups.Group0`

To write code in C#

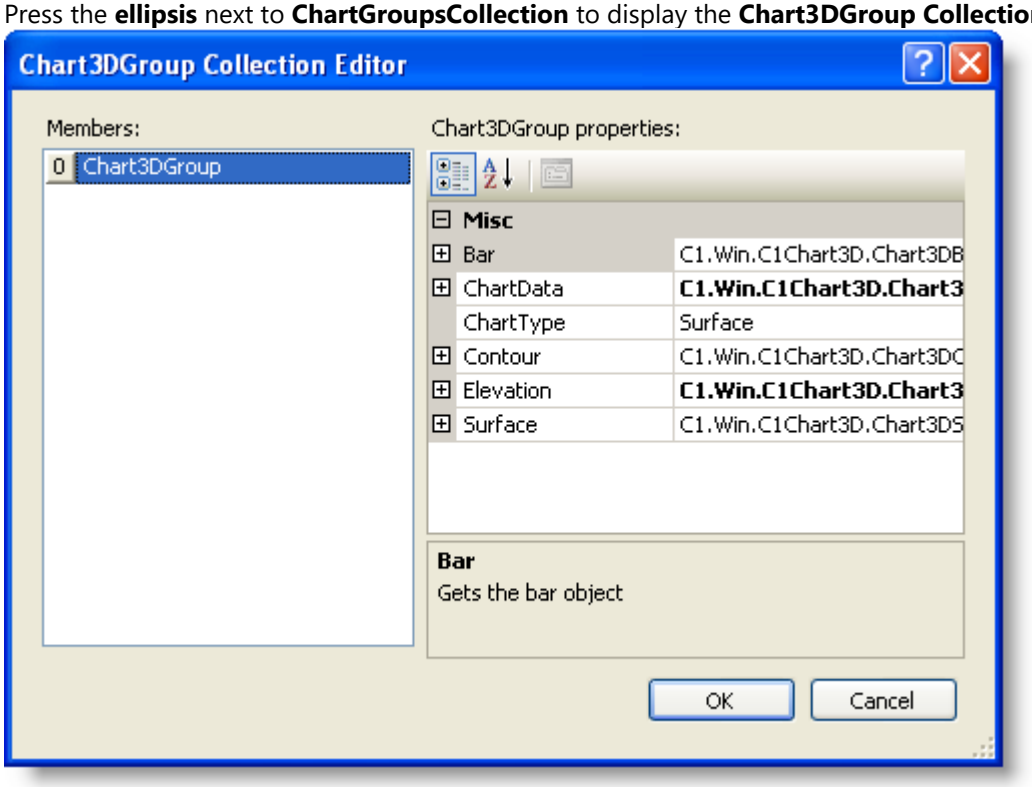
C#

```
c1Chart3D1.ChartGroups.Group0;
```

To access the ChartGroups through the Chart3DGroup Collection Editor:

Properties can be accessed directly from the group as with 2D chart.

1. In the Properties window, expand the **ChartGroups** node.
2. Press the **ellipsis** next to **ChartGroupsCollection** to display the **Chart3DGroup Collection Editor**.



3. Modify the properties as desired. For more information, see 3D Chart Elements.

Modifying Chart Labels

The 3D Chart Labels are used to highlight an important data point, but can also be used generally to provide information on data or on the chart.

The Chart3DLabel object provides a number of properties that help define and position the chart label. The most important of these properties are the following:

- The Text property specifies the text to appear in the chart label, and is of type Label.
- The **View3D** property specifies the position of the chart label in 3D space. The position is specified as LabelView3DEnum.
- The **Connected** property is a Boolean that specifies whether a line is to be drawn from the chart label to its attached location. If **True**, the line is drawn.

To programmatically modify the properties for the Chart Labels:

To write code in Visual Basic

Visual Basic

```
With c1Chart3D1.ChartLabels(1)
  .Text = "Here is my chart label"
  .View3D = LabelView3DEnum.XY
  .IsConnected = True
End With
```

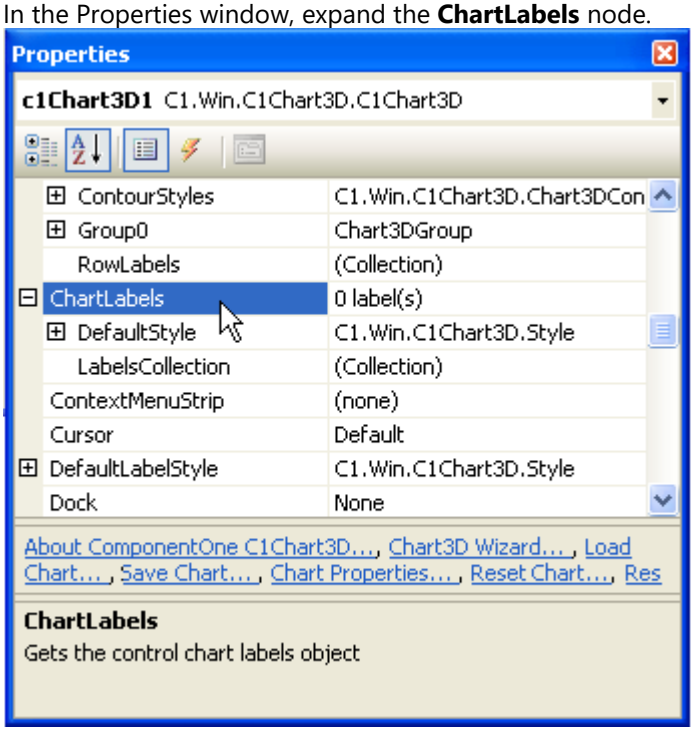
To write code in C#

C#

```
C1.Win.Chart3D.Chart3DLabel lab = C1Chart3D1.ChartLabels[1];
lab.Text = "Here is my chart label";
lab.View3D = LabelView3DEnum.XY;
lab.IsConnect = True;
```

To modify the properties for Chart Labels through the Properties window:

1. In the Properties window, expand the **ChartLabels** node.



2. Modify the properties as desired. For more information, see 3D Labels.

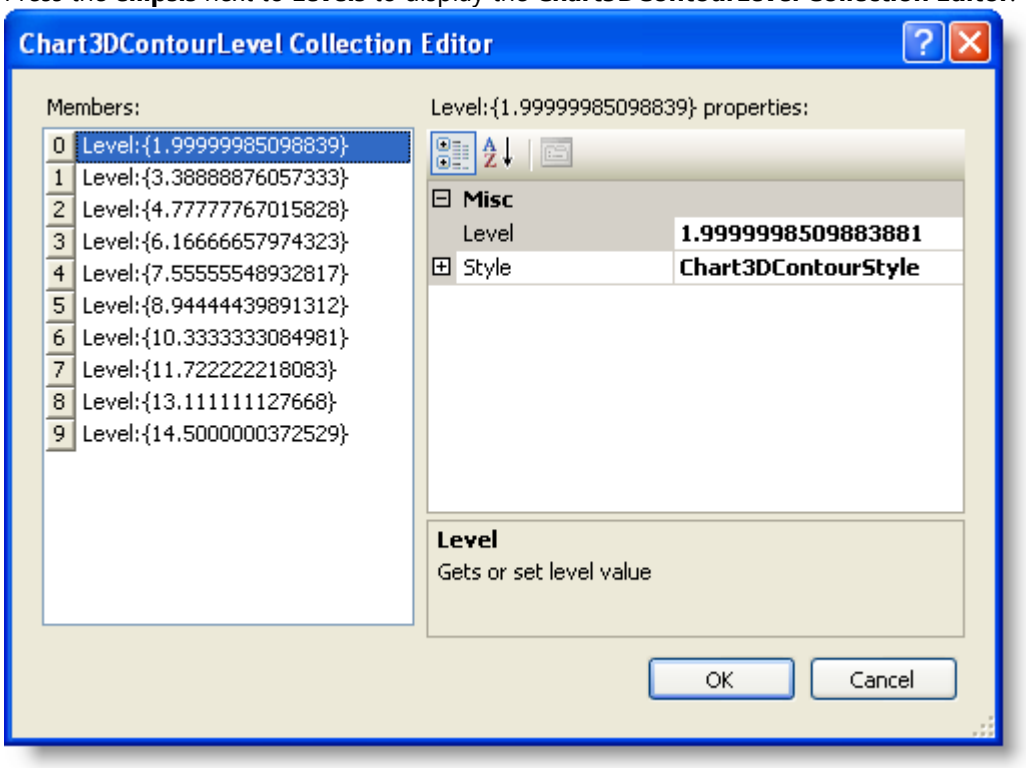
Modifying Contour Levels

The relationship between distribution levels (contour lines/zones) and the Chart3DContourStyles collection is as follows: if *nstyles* contour styles are provided, and the number of distribution levels is *nlevels*, the index of the contour style associated with the *n*th distribution level is the largest integer less than or equal to the following:

```
(n * (nstyles - 1) / nlevels) + 1
```

To modify the Contour Levels through the Chart3DContourLevel Collection Editor:

1. In the Properties window, expand the **ChartGroups** node.
2. Press the **ellipsis** next to **ChartGroupsCollection** to display the **Chart3DGroup Collection Editor**.
3. Expand the **Contour** node.
4. Press the **ellipsis** next to **Levels** to display the **Chart3DContourLevel Collection Editor**.



5. Add, remove, or modify levels as desired. For more information, see Contour Style Appearance Properties.

Modifying Header and Footer Titles

A chart can have two titles, one called the **Header** and one called the **Footer**. Because each title can be positioned above, below, and to the right or left of the chart, it is not necessary to adhere to the traditional concept of **Header** and **Footer** at the top and bottom of an object.

To programmatically modify the 3D Chart Header and Footer Titles:

To write code in Visual Basic

Visual Basic

```
c1Chart3D1.Header.Text = "Sample Header Text"
c1Chart3D1.Header.Compass = C1.Win.C1ChartBase.CompassEnum.East
c1Chart3D1.Header.Location = New Point(20, -1)

c1Chart3D1.Footer.Text = "Sample Footer Text"
c1Chart3D1.Footer.Compass = C1.Win.C1ChartBase.CompassEnum.South
c1Chart3D1.Footer.Location = New Point((C1Chart3D1.Size.Width - 120), -1)
```

To write code in C#

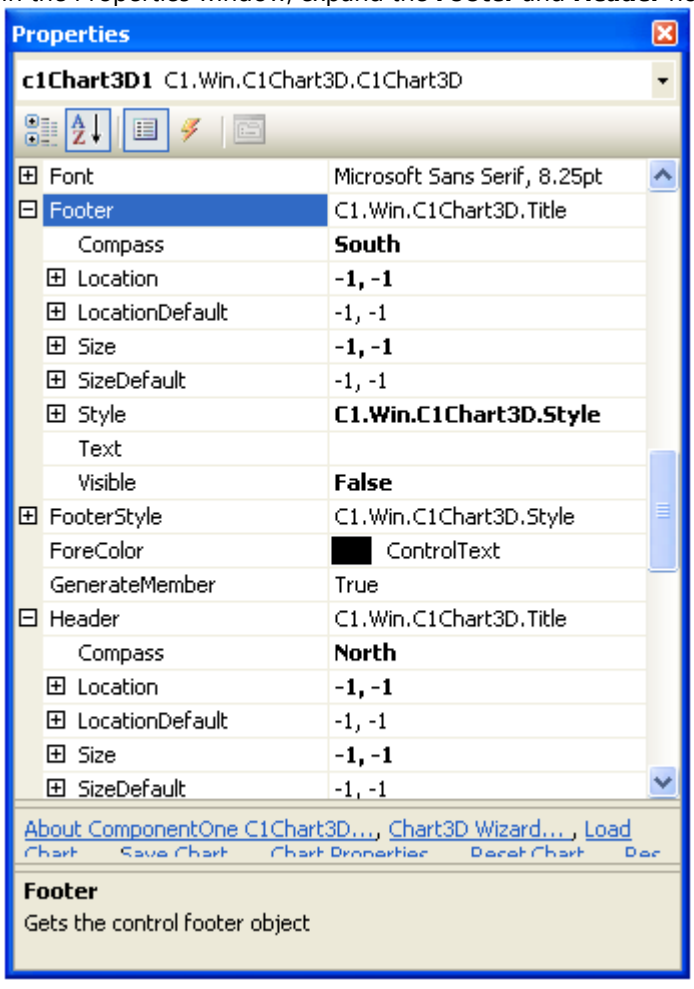
C#

```
c1Chart3D1.Header.Text = "Sample Header Text";
c1Chart3D1.Header.Compass = C1.Win.C1ChartBase.CompassEnum.East;
c1Chart3D1.Header.Location = new Point(20, -1);

c1Chart3D1.Footer.Text = "Sample Footer Text";
c1Chart3D1.Footer.Compass = C1.Win.C1ChartBase.CompassEnum.South;
c1Chart3D1.Footer.Location = new Point((C1Chart3D1.Size.Width - 120), -1);
```

To modify the 3D Chart Header and Footer Titles through the Properties window:

1. In the Properties window, expand the **Footer** and **Header** nodes.



2. Modify the properties as desired. For more information, see 3D Chart Elements.

Modifying the Legend

Whenever data exists in the chart, a **Legend** is automatically generated. The chart assigns the name specified in the ChartDataSeries object for the series as the series identifier. **LineStyle** and **SymbolStyle** determine the symbols that accompany the series name in the **Legend**. The positioning, border, colors and font used for the **Legend** can be

customized.

To programmatically modify the Legend properties:

Sample property settings are listed below:

To write code in Visual Basic

Visual Basic

```
C1Chart3D1.Legend.Compass = CompassEnum.East
C1Chart3D1.Legend.Style.Border.BorderStyle = BorderStyleEnum.Solid
C1Chart3D1.Legend.Style.Border.Thickness = 3
C1Chart3D1.Legend.Style.Border.Color = Color.Black
C1Chart3D1.Legend.Style.BackColor = Color.Gray
C1Chart3D1.Legend.Text = "Legend Text"
```

To write code in C#

C#

```
 c1Chart3D1.Legend.Compass = CompassEnum.East;
 c1Chart3D1.Legend.Style.Border.BorderStyle = BorderStyleEnum.Solid;
 c1Chart3D1.Legend.Style.Border.Thickness = 3;
 c1Chart3D1.Legend.Style.Border.Color = Color.Black;
 c1Chart3D1.Legend.Style.BackColor = Color.Gray;
 c1Chart3D1.Legend.Text = "Legend Text";
```

Note: In VB.NET, it is still possible to use the With and End With statement which may make it easier to show repeated use of the same object.

The code above would look as follows using this method:

To write code in Visual Basic

Visual Basic

```
With C1Chart3D1.Legend
   .Compass = CompassEnum.East
   With .Style
       With.Border
          .BorderStyle = BorderStyleEnum.Solid
          .Thickness = 3
          .Color = Color.Black
       End With
       .BackColor = Color.Gray
   End With
.Text = "Legend Text"
End With
```

To write code in C#

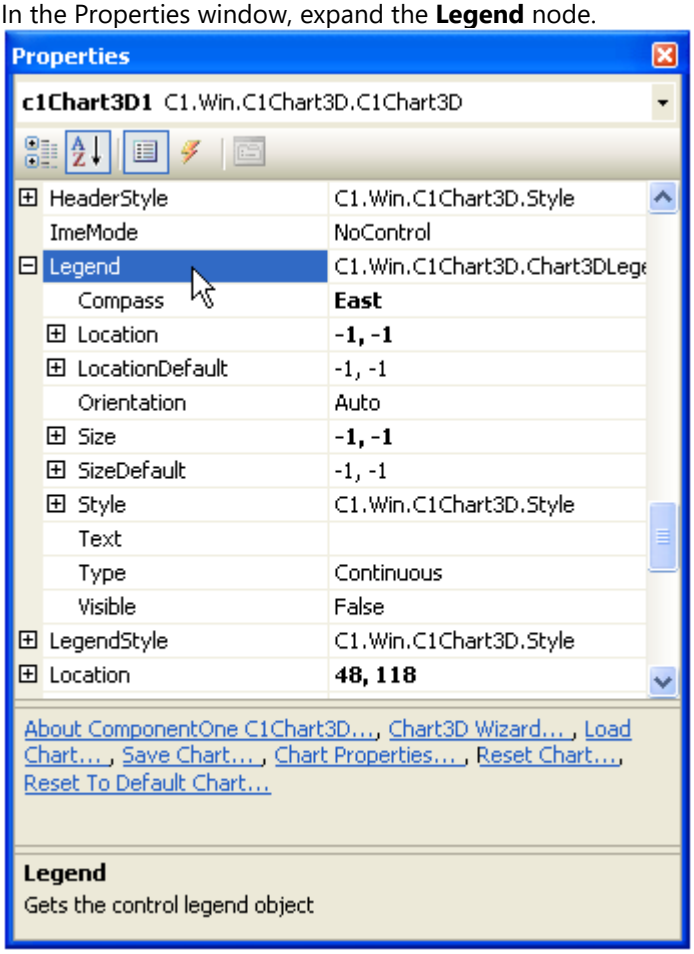
C#

```
C1.Win.C1Chart3D.Legend legend = C1Chart3D1.Legend;
C1.Win.C1Chart3D.Style style = legend.Style;
C1.Win.C1Chart3D.Border border = style.Border;

legend.Compass = CompassEnum.East;
border.BorderStyle = BorderStyleEnum.Solid;
border.Thickness = 3;
border.Color = Color.Black;
style.BackColor = Color.Grey;
legend.Text = "Legend Text";
```

To modify the Legend properties through the Properties window:

1. In the Properties window, expand the **Legend** node.



2. Modify the properties as desired. For more information, see 3D Chart Legend.

Chart 3D for WinForms Frequently Asked Questions

Below are some frequently asked questions (FAQs) about **C1Chart3D**:

How do I change chart type?

1. In the Properties window, expand the **ChartGroups** node.
2. Press the **ellipsis** button next to the **ChartGroupsCollection** to display the **ChartGroups Collection Editor**.
3. Select the desired chart type from the **ChartType** property drop-down list box.

> **Note**: When using 3D Chart, the following chart types will be available:
>
> - Scatter
> - Bar
> - Surface
>
> For more information about chart types, see Basic 3D Chart Types.

How do I change the way chart data is plotted?

1. In the Properties window, expand the **ChartArea** node in Properties.
2. Expand the node of appropriate axis.
3. In Grid Minor, set **Max** and **Min** (maximum and minimum interval to display on axis) and set **UnitMajor** (numerical interval on axis).

For example: If **Max** is set to 20, **Min** is set to 0, and **UnitMajor** is set to 5, every fifth number will display 0, 5, 10, 15, and 20 on the axis. For more information, see Axis Bounds.

How do I change colors displayed in the chart?

To change a series color, complete the following:

1. Right-click on the chart control.
2. Select **Chart Properties**.
3. Choose Data.
4. Expand **SymbolStyle** node.
5. Select desired color from **Color** drop-down list box.

To change chart area color, complete the following:

1. In the Properties window, expand the **ChartArea** node.
2. Expand **Style** node.
3. Select desired color from **BackColor** drop-down list box.

To change axis color, complete the following:

1. In the Properties window, expand the **ChartArea** node.
2. Expand **Style** node.
3. Expand **Style.Font** node.
4. Select desired color from **ForeColor** drop-down list box.

How do I change placement of the Legend?

1. In the Properties window, expand the **Legend** node.
2. Select East, West, North, or South from **Compass** drop-down list box.

For more information, see Legend Positioning.

How do I add or modify a Border?

1. In the Properties window, expand the **ChartArea** node.
2. Expand **Style** node.
3. Expand **Border** node.
4. Select style type from the **BorderStyle** drop-down list box.
5. Select desired color from the **Color** drop-down list box.
6. Increase number of **Thickness** property to make border more prominent or decrease number to make border less prominent.

For more information, see 3D Chart Borders.