ComponentOne

CalendarView 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.

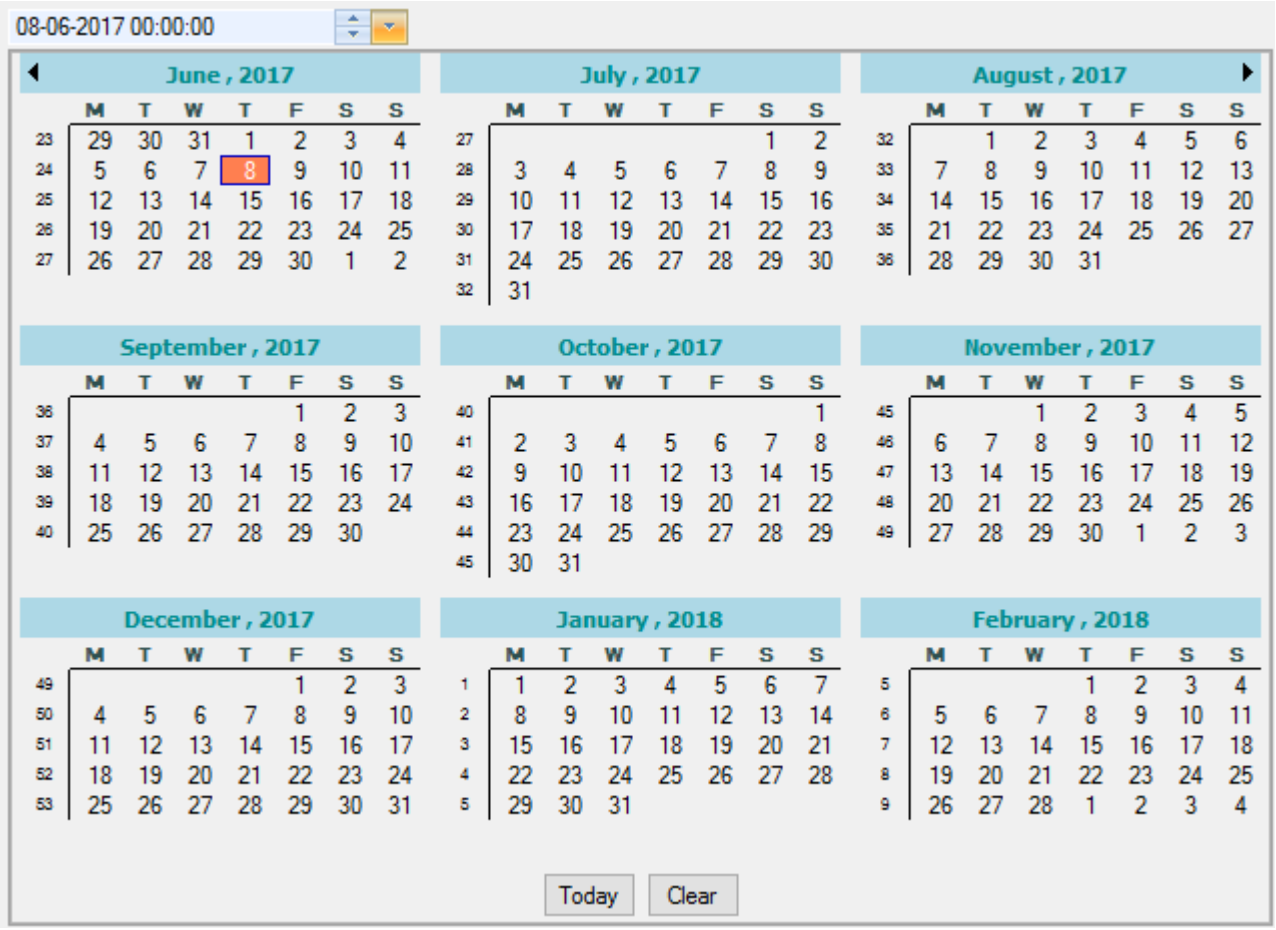
Copying and Distribution

Table of Contents

CalendarView for WinForms

ComponentOne Studio provides **CalendarView for WinForms**, a control that goes beyond navigation and selection of dates. The control provides you with smart tag and design time collection editors so that you can achieve the maximum with the minimum lines of code. Multi-month view, contiguous and non-contiguous selection, keyboard support and theming options make it even more interactive and functional. Furthermore, the control supports internationalization with culture settings combined with right-to-left feature.

In addition, there is **DateEdit for WinForms**, a control that extends the exisiting CalendarView control. The DateEdit control allows selecting and editing dates with support for masking, data validation, null values, maximum and minimum dates, watermark, custom date formats, and internationalization.



Help with WinForms Edition

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

Key Features

CalendarView and DateEdit controls offer numerous features, such as navigation, selection, validation, date formats, and formatting support to let developers build intuitive, professional-looking applications.

- **CalendarView key features**
- **DateEdit key features**

CalendarView key features

- **Quick Navigation**
  CalendarView allows quick and easy navigation through dates, months, and years in different ways. Navigation buttons allow you to go to the previous or the next month. The month and the year selectors enable you to select any month or year from their popups, respectively. Furthermore, it is possible to jump to a specific year by editing the year field.

- **Selection**
  CalendarView supports selection of single as well as multiple dates. While selecting multiple dates, the control allows both contiguous and non-contiguous modes of selection.

- **Multi-month view**
  CalendarView lets you display more than one month in calendar by setting the CalendarDimensions property.

- **Keyboard support**
  CalendarView provides keyboard support for both navigation and selection. You can use various keys to navigate through dates and months, and select multiple dates in a month.

- **Orientation**
  CalendarView allows displaying months vertically and horizontally in the case of multi-month view.

- **Internationalization**
  CalendarView lets you change current culture settings to display calendar in specific locales. In addition, the control provides right-to-left support for languages that follow right-to-left scripts.

- **Theming**
  CalendarView enables you to customize the look and feel of calendar by using pre-defined themes.

- **Styling**
  CalendarView offers various styling features for styling calendar area and calendar elements, such as titles and navigation buttons.

Back to Top

DateEdit key features

- **Date formats**
  DateEdit allows displaying dates in predefined formats, such as short date, long date, and general date, and custom date formats.

- **Data validation**
  DateEdit supports data validation of two types, pre-validation that validates raw input string and post-validation that validates values entered by end users.

- **Null values**
  DateEdit provides flexible rules for handling null values in both read-only and edit modes.

- **Formatting and parsing**
  DateEdit allows formatting values besides using standard and custom format specifiers.

- **Masking**
  DateEdit provides masking support to restrict user input and avoid invalid characters.

- **Internationalization**

DateEdit supports internationalization, i.e. the control can adapt to different languages and cultures without modifications and also support languages following right-to-left scripts.

- **Styling**
  DateEdit offers customization features for styling its elements, such as titles and navigation buttons.

Back to Top

Object Model Summary

The CalendarView and the DateEdit control have a rich object model, providing various classes, objects, and associated methods and properties. This section covers object model summary of both CalendarView and DateEdit controls separately.

- **CalendarView**
- **DateEdit**

CalendarView

C1CalendarView
Properties: AnnuallyBoldedDates, BackColor, BackgroundImage, BackgroundImageLayout, BoldedDates, CalendarDimensions, CalendarWeekRule, CurrentMonthDisplayOffset, DayTitlePosition, DisabledDates, FirstDayOfWeek, ForeColor, MaxColumns, MaxDate, MaxSelectionCount, MinDate, MonthTitlePosition, PeriodSelectionType, RightToLeftLayout, SelectedDates, ShowArrowButtons, ShowToday, ShowToolTips, ShowWeekNumbers, Theme, VerticalOrientationLayout, WorkDays **Events:** RightToLeftLayoutChanged, SelectionChanged, StylesChanged
BaseArrowStyle
Property: BackImage, BackImageAlignment, BackImageScaling, ForeColor
BaseStyle
Property: BackColor, Border, BorderColor, HorizontalAlignment, Name, VerticalAlignment
CalendarTheme
Properties: Common, Day, NavigationButtons, Titles
CommonStyle
Properties: BackImage, BackImageAlignment, BackImageScaling, Font, VerticalAlignment
DayStyle
Properties: Font, ForeColor
DayTheme
Properties: Bolded, Disabled, Ordinary, Selected, Today, Trail, Weekend
DayTitleStyle
Properties: BackImage, BackImageAlignment, BackImageScaling, ForeColor
MonthTitleStyle
Properties: BackImage, BackImageAlignment, BackImageScaling, ForeColor, Padding, Trimming
NavigationButtonsTheme
Properties: ArrowNext, ArrowPrevious, ImageArrowNext, ImageArrowPrevious
TitleTheme
Properties: Day, Month, Week, Weekend

WeekTitleStyle
Properties: Font, ForeColor

Back to Top

DateEdit

C1DateEdit
Properties: AllowSpinLoop, Calendar, FormatType
CalendarSettings
Property: AnnuallyBoldedDates, ArrowColor, BackColor, BoldedDates, Calendar, CalendarDimensions, CalendarWeekRule, CaptionFormat, ClearText, CurrentCulture, CurrentMonthDisplayOffset, DateIsNull, DayNameLength, DayNamesColor, DayNamesFont, DisabledDates, FirstDayOfWeek, FirstMonth, Font, ForeColor, LastMonth, LineColor, MaxDate, MinDate, RightToLeft, RightToLeftLayout, SelectedDate, SelectionBackColor, SelectionForeColor, ShowClearButton, ShowToday, ShowTodayButton, ShowTodayCircle, ShowWeekNumbers, TitleBackColor, TitleFont, TitleForeColor, TitleHeight, TitleNavigation, TodayBorderColor, TodayText, TrailingForeColor, VisualStyle **Events:** ClearButtonClick, ClearButtonVisibilityChanged, DateValueChanged, DateValueSelected, MonthChanged, RightToLeftLayoutChanged, TodayButtonClick, TodayButtonVisibilityChanged, VisualStyleChanged
DropDownCalendar
Property: BackColor, Font, ForeColor

Back to Top

Quick Start

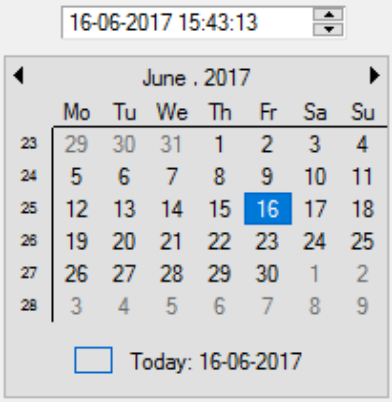
This quick start gets you started with the CalendarView and the DateEdit control by letting you create a WinForms application, add CalendarView and DateEdit controls to it, and input a date from CalendarView into DateEdit.

Note that the quick start binds CalendarView to DateEdit to input a date. However, you can input a date into DateEdit directly from DateEdit's calendar pop-up.

To quickly get started using the controls, follow these steps:

1. **Add the CalendarView and the DateEdit control to the application**
2. **Customize the Calendarview and the DateEdit control**
3. **Input date from CalendarView into DateEdit**

The following image shows the DateEdit contol displaying today's date selected from the CalendarView control.



Step 1: Add the CalendarView and the DateEdit control to the application

1. Create a **Windows Forms Application** in Visual Studio.
2. Drag and drop the **C1CalendarView** control from the Toolbox to the application.
3. Drag and drop the **C1DateEdit** control from the Toolbox to the application.

Step 2: Customize the CalendarView and the DateEdit control

1. Switch to the code view.
2. Add the following code to customize CalendarView and DateEdit with respect to location, size, and back color.
   - **Visual Basic**

```
' set CalendarView location
C1CalendarView1.Location = New System.Drawing.Point(300, 152)

' set CalendarView backcolor
C1CalendarView1.Theme.Common.BackColor =
    System.Drawing.Color.FromArgb(CInt(CByte(224)),
                                  CInt(CByte(224)),
                                  CInt(CByte(224)))

' hide CalendarView
C1CalendarView1.Visible = False

' set DateEdit size and location
C1DateEdit1.Size = New System.Drawing.Size(150, 20)
C1DateEdit1.Location = New System.Drawing.Point(330, 126)
```

   - **C#**

```
// set CalendarView location
c1CalendarView1.Location = new System.Drawing.Point(300, 152);

// set CalendarView backcolor
c1CalendarView1.Theme.Common.BackColor = System.Drawing.Color.FromArgb
                                            (((int)(((byte)(224)))),
                                            ((int)(((byte)(224)))),
                                            ((int)(((byte)(224)))));
// hide CalendarView
c1CalendarView1.Visible = false;

// set DateEdit size and location
c1DateEdit1.Size = new System.Drawing.Size(150, 20);
c1DateEdit1.Location = new System.Drawing.Point(330, 126);
```

Back to Top

Step 3: Input date from CalendarView into DateEdit

Add the following code to pop up CalendarView by clicking into DateEdit, and then select and input a date from CalendarView into the DateEdit control.

- **Visual Basic**

```
Private Sub Form1_Load(sender As Object, e As EventArgs)
    ' handle the textbox Click event
    AddHandler c1DateEdit1.Click, AddressOf c1DateEdit1_Click

    ' handle the CalendarView SelectionChanged event
    AddHandler c1CalendarView1.SelectionChanged, AddressOf CalendarView_SelectionChanged

    c1CalendarView1.Visible = False

    ' hide DateEdit's calendar pop-up
    c1DateEdit1.ShowDropDownButton = False
End Sub
Private Sub c1DateEdit1_Click(sender As Object, e As EventArgs)
    c1CalendarView1.Visible = True
    AddHandler c1CalendarView1.SelectionChanged, AddressOf CalendarView_SelectionChanged
End Sub

Private Sub CalendarView_SelectionChanged(sender As Object, e As EventArgs)
    ' input selected date into textbox from CalendarView
    If c1CalendarView1.Visible Then
        c1DateEdit1.Text = c1CalendarView1.SelectedDates(0).ToShortDateString()
    End If

    ' hide CalendarView
    c1CalendarView1.Hide()
End Sub
```

- **C#**

```
private void Form1_Load(object sender, EventArgs e)
{
    // handle the textbox Click event
    c1DateEdit1.Click += c1DateEdit1_Click;

    // handle the CalendarView SelectionChanged event
    c1CalendarView1.SelectionChanged += CalendarView_SelectionChanged;

    c1CalendarView1.Visible = false;

    // hide DateEdit's calendar pop-up
    c1DateEdit1.ShowDropDownButton = false;
}
private void c1DateEdit1_Click(object sender, EventArgs e)
{
    c1CalendarView1.Visible = true;
    c1CalendarView1.SelectionChanged += CalendarView_SelectionChanged;
}

private void CalendarView_SelectionChanged(object sender, EventArgs e)
{
    // input selected date into textbox from CalendarView
    if (c1CalendarView1.Visible)
        c1DateEdit1.Text = c1CalendarView1.SelectedDates[0].ToShortDateString();

    // hide CalendarView
    c1CalendarView1.Hide();
}
```

Back to Top

Elements

CalendarView consists of various elements that help in navigation of days, months, and years, provide titles in calendar, and allow creating custom dates.

Navigation and Title Elements
: Learn about navigation and title elements of CalendarView.

Month View Elements
: Learn about month view elements of CalendarView.

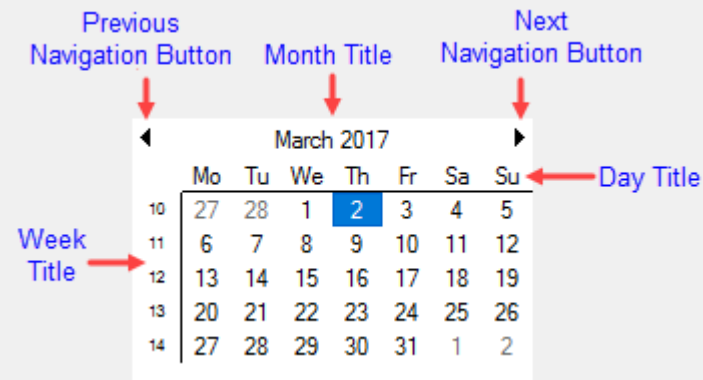
DateEdit Elements
: Learn about dropdown, updown spin buttons, combo box, and calendar pop-up in DateEdit.

Navigation and Title Elements

CalendarView comprises navigation and title elements, as follows:

- **Navigation buttons**: CalendarView consists of Previous and Next navigation buttons that allow you to go to the previous or the next month respectively.
- **Popup month and year selectors**: CalendarView includes popup month and year selectors that appear when you click on the month title. These selectors allow you to jump to a specific month or year by selecting the same from the list.
- **Titles**: CalendarView includes the following titles:
  - **Month title**: The month title, also known as calendar title appears at the top of calendar by default and displays the month and year.
  - **Week title**: The week title appears to the left of calendar and displays the week numbers.
  - **Day title**: The day title appears beneath the month title and displays the names of the week days.

The following image labels the elements that make up the calendar navigation system.

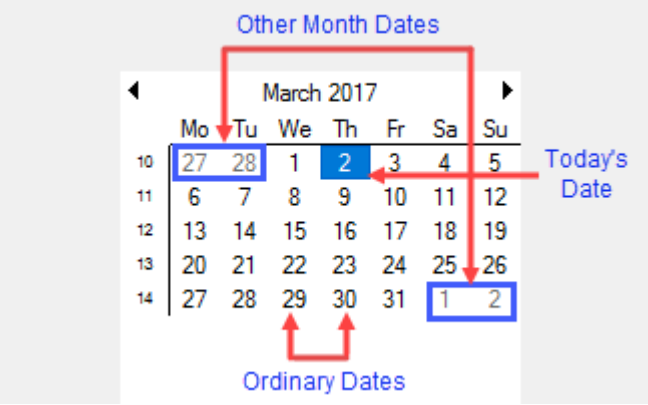


Month View Elements

CalendarView consists of the following Month View elements:

- **Ordinary dates**: Indicates the dates that belong to the current month.
- **Selected dates**: Indicates the dates selected either at design-time or at run-time.
- **Today's dates**: Indicates today's date with a filled rectangular box around the date.
- **Disabled dates**: Indicates the dates that are unselectable.
- **Other month dates**: Indicates the dates that belong to next and previous months, but lie in the same week as that of first or last date of the current month.

The following image labels the elements that make up the CalendarView's Month View Area:

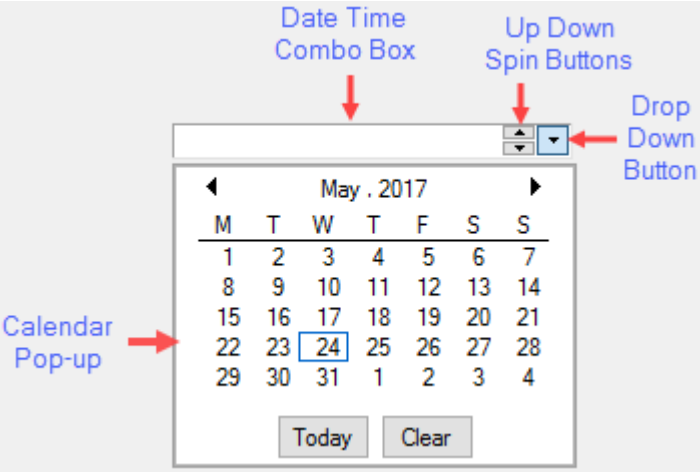


DateEdit Elements

DateEdit consists of the following elements:

- **Date time combo box**: DateEdit includes a date time combo box that allows entering date manually as well as inputting a selected date from the calendar pop-up.
- **Up down spin buttons**: DateEdit includes up down spin buttons that spin the selected part, namely day, month, year, hour, minute, and second one step up or down.
- **Dropdown button**: DateEdit includes a dropdown button that displays calendar pop-up when clicked.
- **Calendar pop-up**: DateEdit includes calendar pop-up that comprises titles, navigation buttons, pop-up month and year selectors, month view elements, and Today and Clear buttons.

The following image labels the elements that make up the DateEdit control.



Design-Time Support

CalendarView and DateEdit provide design-time support to simplify working with the object model. While both controls include a smart tag to readily access properties, CalendarView also comprises collection editors to let you quickly add and remove dates.

Smart Tag
: Learn how to use Smart Tag to access common properties of CalendarView and DateEdit.

Collection Editors
: Learn how to use collection editors to add or remove various types of dates.

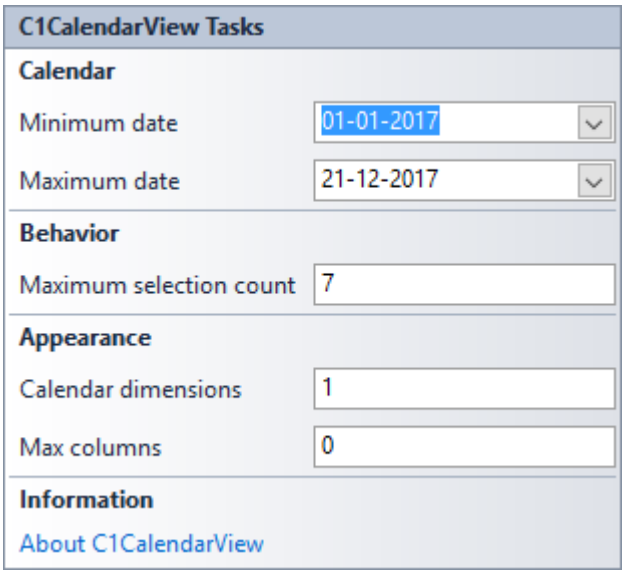
Smart Tag

CalendarView and DateEdit provide design-time support to simplify working with their object model. The controls include smart tags to readily access common properties and other options. Smart tag is a short-cut to the C1CalendarView and the C1DateEdit Tasks menu that provide a quick access to common properties of the controls.

This section dicusses smart tags used to access Tasks menu of both CalendarView and DateEdit controls, respectively.

- **CalendarView tasks menu**
- **DateEdit tasks menu**

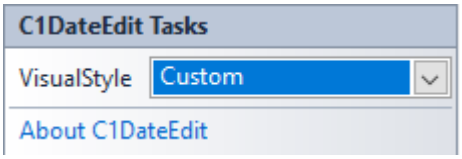
CalendarView tasks menu



The C1CalendarView Tasks menu provides options, as follows:

- **Minimum date**
  Allows you to set the minimum allowable date.
- **Maximum date**
  Enables you to set the maximum allowable date.
- **Maximum selection count**
  Allows you to set the maximum number of dates that can be selected.
- **Calendar dimensions**
  Enables you to set the number of months to be shown.
- **Max columns**
  Allows you to set the maximum number of columns for calendar dimensions.
- **About C1CalendarView**
  Enables you to find the version number of the control and online resources.

DateEdit tasks menu



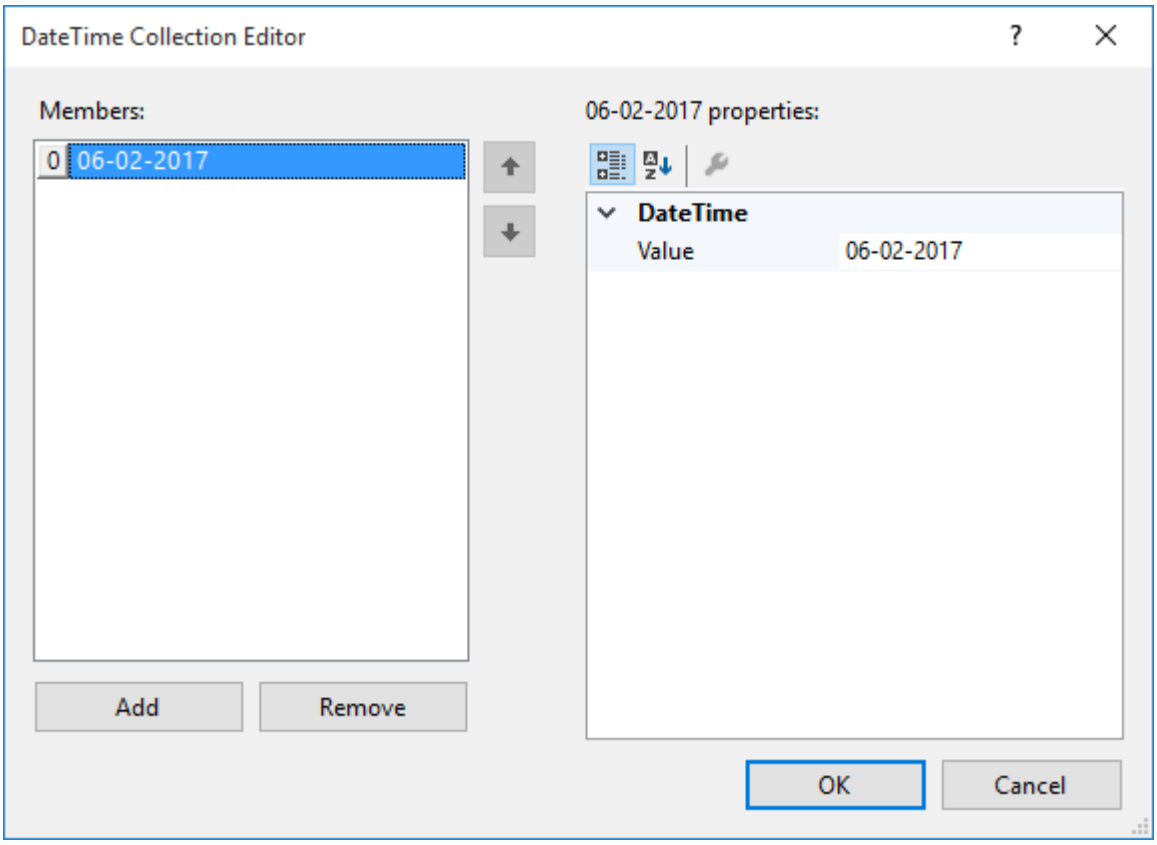
The C1DateEdit Tasks menu provides the following options:

- **VisualStyle**
  Allows you to set the desired theme from pre-defined themes of the DateEdit control.

- **About C1DateEdit**
  Allows you to find the version number of the control and online resources.

Collection Editors

CalendarView provides DateTime Collection Editor to let you add or remove annually bolded, bolded, and disabled dates.



To access DateTime Collection Editor to add or remove annually bolded dates, follow these steps:

1. Right-click on the **C1CalendarView** control and select **Properties** from the context menu.
2. In the **Properties** window, click the ellipsis button next to the **AnnuallyBoldedDates** property to open DateTime Collection Editor.

Note: To access DateTime Collection Editor for bolded or disabled dates, click the ellipsis button next to the **BoldedDates** or the **DisabledDates** property respectively.

Using Controls

The following sections allow you to explore various features offered by both CalendarView and DateEdit controls.

CalendarView Features
: Learn how to use the CalendarView control to navigate and select dates, months, or years through Mouse as well Keyboard.

DateEdit Features
: Learn how to use the DateEdit control to enter or display dates in several predefined or custom formats.

CalendarView Features

CalendarView provides features to help you navigate and select days, months, and years. In addition, the control supports features to let you set culture, display multiple months, and customize the appearance of calendar.

The following sections let you explore various features of CalendarView.

Navigation
: Learn how to navigate through dates, months, or years in CalendarView.

Selection
: Learn how to select single or multiple days in CalendarView.

Keyboard Support
: Learn how to use several key combinations for quick navigation and selection in CalendarView.

Multi-Month View
: Learn how to display multiple months in CalendarView.

Custom Dates
: Learn how to add or remove custom dates in CalendarView.

Culture Settings
: Learn how to change current culture in CalendarView.

Right to Left Support
: Learn how to implement right to left support in CalendarView.

Appearance and Styling
: Learn how to customize the appearance of CalendarView.

Navigation

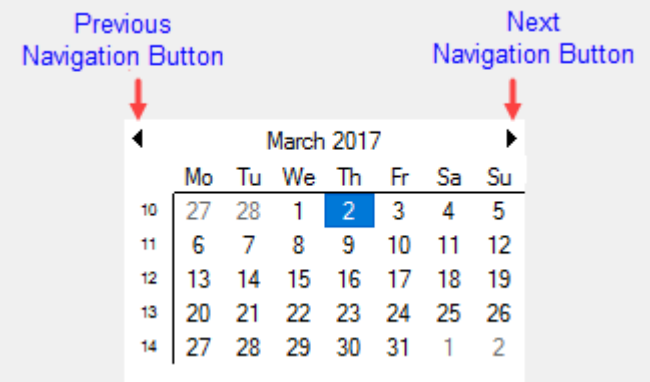
CalendarView contains an advanced navigation system for navigating through dates, months, or years. The control allows navigation in three ways, as follows:

- **Navigation buttons**
- **Popup month and year selectors**
- **Year field**

Navigation buttons

CalendarView consists of navigation buttons, Previous and Next, which allow you to go to the previous or the next month within the maximum and minimum date range. On navigating to a different month, calendar sets the first selectable day of the previous or next month as the current day.

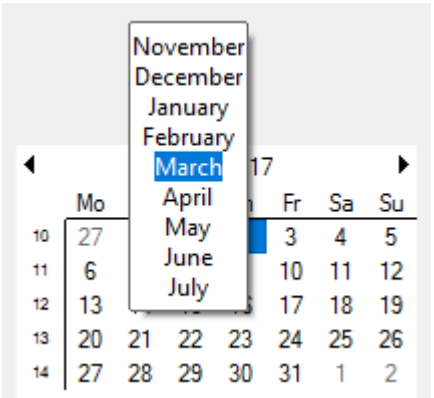
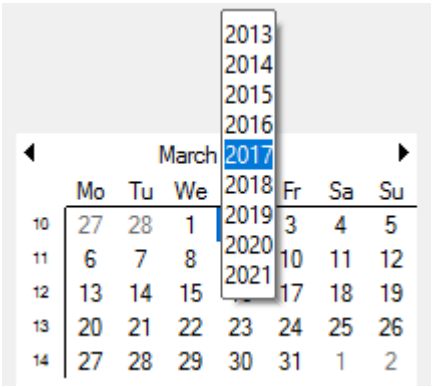
The following image displays the navigation buttons.



Popup month and year selectors

Clicking the month in calendar title opens the popup month selector. The selector displays nine consecutive months, four before and four after the current month. Similarly, clicking the year in calendar title opens up the popup year selector displaying nine consecutive years. The years include four years before and four years after the current year. The popup year selector opens only if you set the value of the PeriodSelectionType property to **List**.

The following images display the popup month and year selectors.

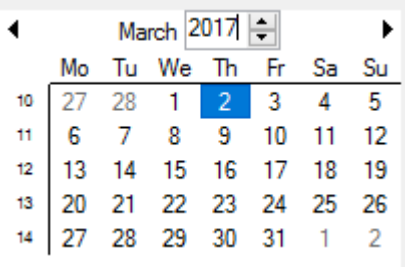
	
Popup Month Selector	**Popup Year Selector**

Back to Top

Year field

Clicking the year in calendar title activates the year field in the case of PeriodSelectionType set to **Field**. The year field enables you to input any year value and jump to that specific year right from the current year. However, the input year value should lie in the range of selectable dates.

The following image displays the year field.



Back to Top

Selection

CalendarView allows selecting single or multiple days depending upon the maximum number of days you can select. However, the control enables you to select only a single day by default. To specify the maximum number of selectable days, set the MaxSelectionCount property.

Multiple selection

When selecting multiple days, CalendarView supports contiguous or non-contiguous selection. Contiguous selection lets you select adjacent days, while keeping the SHIFT key pressed. On the other hand, non-contiguous selection lets you select separate days, while keeping the CTRL key pressed.

Selectable days

Selectable days are the days that you allow to be selected. They are not set as disabled and they lie between the maximum and the minimum allowable dates. By default, the maximum and the minimum allowable dates are 12/31/9999 and 01/01/0001 respectively. To specify the maximum and minimum allowable dates, set the MaxDate and the MinDate properties respectively.

Week number selector

CalendarView provides the week number selector that appears vertically on the left of calendar. The selector contains week numbers that represent each week in the calendar month. Because a calendar year contains 52 weeks in total, CalendarView has week numbers from 1 to 52.

The week number selector allows selecting all days in a specific week in a single click. Note that all days in a week are selectable only if MaxSelectionCount is greater than or equal to 7.

Keyboard Support

CalendarView provides keyboard support for quick navigation and selection through dates and months. The following table lists the applicable key combinations and their corresponding actions:

Keyboard Command	Action
Up	Navigates or selects the day by one position in the upward direction.
Down	Navigates or selects the day by one position in the downward direction.
Left	Navigates or selects the day by one position in the left direction.
Right	Navigates or selects the day by one position in the right direction.
Home	Navigates to the first day of the current month.
End	Navigates to the last day of the current month.
Page Up	Navigates to the same day of the previous month.
Page Down	Navigates to the same day of the next month.
Shift + Up	Selects contiguous days in the decreasing order from the current day based on maximum selectable days.
Shift + Down	Selects contiguous days in the increasing order from the current day based on maximum selectable days.

Shift + Left	Selects days successively by one position in the left direction until maximum selectable days..
Shift + Right	Selects days successively by one position in the right direction until maximum selectable days.
Shift + Home	Selects contiguous days in the decreasing order from the current day based on maximum selectable days.
Shift + End	Selects contiguous days in the increasing order from the current day based on maximum selectable days.
Shift + Page Up	Selects contiguous days in the decreasing order from the current day based on maximum selectable days.
Shift + Page Down	Selects contiguous days in the increasing order from the current day based on maximum selectable days.
Ctrl + Up	Selects days upwards in the same column from the current day based on maximum selectable days.
Ctrl + Down	Selects days downwards in the same column from the current day based on maximum selectable days.
Ctrl + Left	Selects days successively by one position in the left direction until maximum selectable days..
Ctrl + Right	Selects days successively by one position in the right direction until maximum selectable days.
Ctrl + Home	Selects the first day of the current month keeping the current day selected.
Ctrl + End	Selects the last day of the current month keeping the current day selected.
Ctrl + Page Up	Selects the same day of the previous month keeping the current day in the current month selected.
Ctrl + Page Down	Selects the same day of the next month keeping the current day in the current month selected.

Multi-Month View

CalendarView supports multi-month view that enables you to create a full year view to see several months of calendar on the same screen at the same time. A full year view is easily visible, scanned, and understood. It also lets you make the navigation and selection seamless for users. In addition, viewing a full year at a glance can make planning long-term activities and projects easier. For instance, it is easier to view holidays in a specific year when it displays all months in the multi-month mode.

Multi-month presentation

CalendarView enables you to display multi-month view with fast rendering. You can easily configure the control to show more than one month in the calendar area.

To display multiple months in CalendarView, set the CalendarDimensions property to a value greater than 1 and change the control dimensions. The control should show as many months as possible into the available space, depending upon the value set in CalendarDimensions.

Horizontal and vertical views

CalendarView supports horizontal as well as vertical view. A horizontal calendar contains more columns than rows, while a vertical calendar contains more rows than columns.

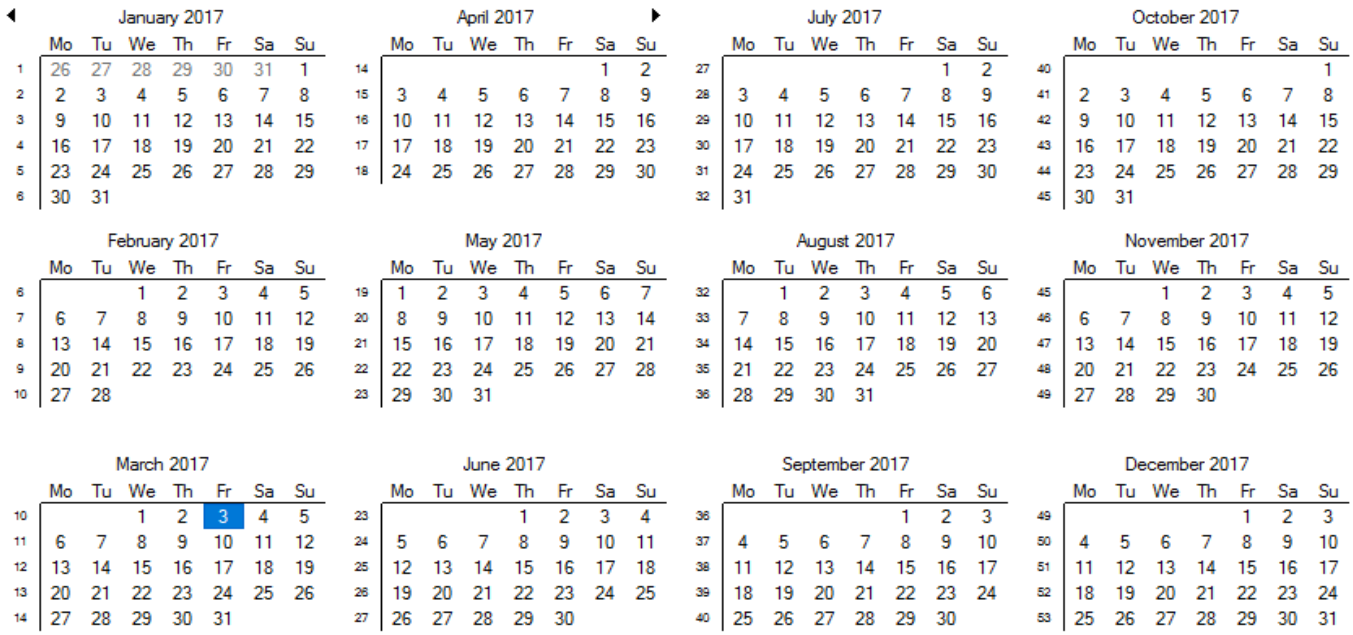
Display CalendarView horizontally or vertically by simply decreasing the height and increasing the width of the control or vice-versa. Set the width as per the number of columns the control needs to display. To specify the number of columns to display, set the MaxColumns property.

Orientation

CalendarView supports both horizontal and vertical orientation. The horizontal orientation is the default orientation.

The CalendarView control displays multiple months from left to right in horizontal orientation and from top to bottom in vertical orientation. To display multi-month view in vertical orientation, set the VerticalOrientationLayout property.

The following image displays yearly calendar 2017 using multi-month view in CalendarView.



The following code snippet shows how to implement multi-month view in CalendarView.

- **Visual Basic**

```
' specify the number of months to display
C1CalendarView1.CalendarDimensions = 12

' specify the maximum number of columns to display
C1CalendarView1.MaxColumns = 4

' set the vertical orientation
C1CalendarView1.VerticalOrientationLayout = True
```

- **C#**

```
// specify the number of months to display
c1CalendarView1.CalendarDimensions = 12;

// specify the maximum number of columns to display
c1CalendarView1.MaxColumns = 4;

// set the vertical orientation
```

```
c1CalendarView1.VerticalOrientationLayout = true;
```

Custom Dates

CalendarView supports custom dates, such as selected, bolded, annually bolded, and disabled dates. Using custom dates allows highlighting specific dates for important events or disable certain dates for providing custom functionality in calendar. The control allows adding or removing custom dates through DateTime Collection Editor as well as through code.

The following sections describe how to add custom dates in CalendarView:

Selected Dates
: Learn how to add selected dates through code in CalendarView.

Bolded Dates
: Learn how to add bolded dates through code in CalendarView.

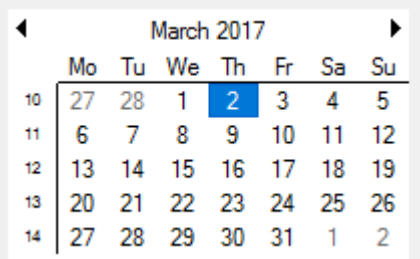
Disabled Dates
: Learn how to add disabled dates through code in CalendarView.

Selected Dates

In CalendarView, selected dates are highlighted with a different backcolor and forecolor.

To add selected dates in CalendarView, set the SelectedDates property of C1CalendarView.

The following image shows a selected date in CalendarView.



The following code snippet shows how to add a selected date using the SelectedDates property in CalendarView.

- **Visual Basic**

```
' add a selected date
C1CalendarView1.SelectedDates =
    New DateTime() {New DateTime(2017, 2, 23, 0, 0, 0, 0)}
```

- **C#**

```
// add a selected date
c1CalendarView1.SelectedDates = new DateTime[]
{
    new DateTime(2017, 2, 23, 0, 0, 0, 0)
};
```

Bolded Dates

Bolded dates allow you to highlight important events, appointments, or activities on specific dates in calendar.

At design-time, it is possible to add bolded dates through DateTime Collection Editor. For more information about

adding bolded dates through DateTime Collection Editor, see Collection editors. To add bolded dates through code in CalendarView, set the BoldedDates property.

The following image shows a bolded date in CalendarView.

The following code snippet shows how to add a bolded date using the BoldedDates property in CalendarView.

- **Visual Basic**

```
' add a bolded date
C1CalendarView1.BoldedDates =
    New DateTime() {New DateTime(2017, 3, 13, 0, 0, 0, 0)}
```

- **C#**

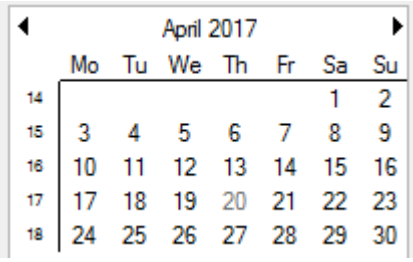
```
// add a bolded date
c1CalendarView1.BoldedDates = new DateTime[]
{
    new DateTime(2017, 3, 13, 0, 0, 0, 0)
};
```

Disabled Dates

Disabled dates are the ones that appear disabled and are not selectable at run-time.

Add disabled dates at design-time through DateTime Collection Editor. For more information about adding disabled dates through DateTime Collection Editor, see Collection editors. To add disabled dates through code in CalendarView, set the DisabledDates property.

The following image shows a disabled date in CalendarView.



The following code snippet shows how to add a disabled date using the DisabledDates property in CalendarView.

- **Visual Basic**

```
' add a disabled date
C1CalendarView1.DisabledDates =
    New DateTime() {New DateTime(2017, 3, 20, 0, 0, 0, 0)}
```

- **C#**

```
// add a disabled date
c1CalendarView1.DisabledDates = new DateTime[]
{
```

```
    new DateTime(2017, 3, 20, 0, 0, 0, 0)
};
```

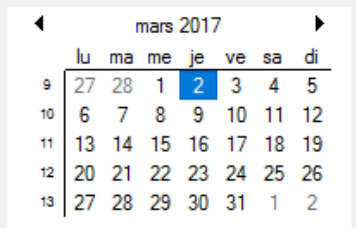
Culture Settings

CalendarView allows you to change regional settings to display calendar in specific locales or languages. The control enables formatting, parsing, and validating data depending upon cultural settings. It is possible to include string comparison, numeric, and date time formats and special characters like decimal point character in culture settings.

CalendarView supports full localization on week title and month title through the System.Globalization Namespace. It includes information pertaining to the culture such as language, country/region, and format patterns for dates, currency, and numbers etc. Specifying locale for the current culture affects the string displayed on the week title and the month title. By default, the current culture designates through the CurrentCulture property of **System.Threading.Thread.CurrentThread**.

To use a culture other than the current culture, set the desired culture by specifying the culture name. For more information about different culture names and identifiers, see CultureInfo class in .NET Framework documentation.

The following image shows CalendarView with the French culture applied.



The following code snippet shows how to set the current culture in CalendarView.

- **Visual Basic**

```
' Set desired culture, for example here the French (France) locale.
System.Threading.Thread.CurrentThread.CurrentCulture = New System.Globalization.CultureInfo("fr-FR")

' This call is required by the Windows Form Designer.
InitializeComponent()
```

- **C#**

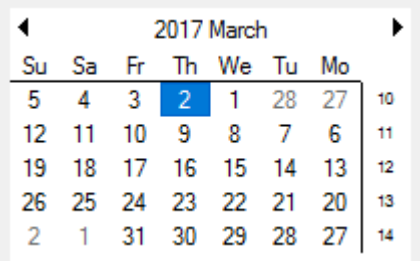
```
// Set desired culture, for example here the French (France) locale.
System.Threading.Thread.CurrentThread.CurrentCulture = new System.Globalization.CultureInfo("fr-FR");

// This call is required by the Windows Form Designer.
InitializeComponent();
```

Right to Left Support

CalendarView supports Right-to-Left functionality for languages that follow Right-to-Left scripts.

The control enables you to display calendar in the Right-to-Left direction by using the RightToLeftLayout and the **RightToLeft** properties. When both RightToLeftLayout and RightToLeft properties are **true** and **Yes** respectively, the calendar appears in the Right-to-Left direction.

The following image displays CalendarView in the Right-to-Left direction.



The following code snippet shows how to set the RightToLeftLayout and the RightToLeft properties in CalendarView.

- **Visual Basic**

```
' enable the Right to Left Layout
C1CalendarView1.RightToLeftLayout = True

' set the RightToLeft property
C1CalendarView1.RightToLeft = RightToLeft.Yes
```

- **C#**

```
// enable the Right to Left Layout
c1CalendarView1.RightToLeftLayout = true;

// set the RightToLeft property
c1CalendarView1.RightToLeft = RightToLeft.Yes;
```

Appearance and Styling

CalendarView provides several styling features that allow customization of calendar as well as its elements including month view, navigation, and title elements.

The list of various styling features is as follows:

- **Alignment:** Apply horizontal or vertical alignment to both calendar and its elements, such as title, navigation, and month view elements.
  - **Horizontal alignment**: Align calendar or its elements horizontally by using the HorizontalAlignment property.
  - **Vertical alignment**: Align calendar or its elements vertically by using the VerticalAlignment property.

- **Background images:** Add, align, and scale background images in calendar as well as navigation buttons and titles including day, month, and weekend titles.
  - **Background image**: Set background image by using the BackImage property.
  - **Background image alignment:** Align background image by using the BackImageAlignment property.
  - **Background image scaling**: Scale background image by using the BackImageScaling property.

- **Common styles:** Apply common styles, such as border, border color, font settings and background or foreground color to calendar and elements in it.
  - **Border**: Set the width of the visual element by setting the Border property.
  - **Border color**: Change the color of the border by setting the BorderColor property.
  - **Font settings**: Apply font settings to all types of dates, such as bolded, ordinary, selected, today, disabled, trailing, and weekend and to all titles including day, month, week, and weekend.
  - **Background color**: Change the background color of calendar or its elements by setting the BackColor property.
  - **Foreground color**: Apply foreground color to different dates, titles, and navigation buttons in calendar by setting the ForeColor property.

- **Navigation button images:** Set images for both Previous and Next navigation buttons.
  - **Previous arrow button image**: Set the Previous arrow button image by setting the ImageArrowPrevious property.
  - **Next arrow button image**: Set the Next arrow button image by setting the ImageArrowNext property.

The following image displays styling features applied to CalendarView and its elements.

The following code snippet shows how to implement styling features in CalendarView using relevant properties.

- **Visual Basic**

```
' set horizontal and vertical alignment for titles and calendar
C1CalendarView1.Theme.Titles.Week.HorizontalAlignment = C1.Framework.Alignment.Far
C1CalendarView1.Theme.Common.HorizontalAlignment = C1.Framework.Alignment.Near
C1CalendarView1.Theme.Titles.Week.VerticalAlignment = C1.Framework.Alignment.Far
C1CalendarView1.Theme.Common.VerticalAlignment = C1.Framework.Alignment.Near

' set borders for different dates in calendar
C1CalendarView1.Theme.Day.Bolded.Border = New C1.Framework.Thickness(2, 2, 2, 2)
C1CalendarView1.Theme.Day.Ordinary.Border = New C1.Framework.Thickness(1, 1, 1, 1)
C1CalendarView1.Theme.Day.Weekend.Border = New C1.Framework.Thickness(1, 1, 1, 1)

' set border color for dates and titles in calendar and for calendar as well
C1CalendarView1.Theme.Day.Bolded.BorderColor = Color.Crimson
C1CalendarView1.Theme.Day.Ordinary.BorderColor = Color.PeachPuff
C1CalendarView1.Theme.Day.Today.BorderColor = SystemColors.ControlLightLight
C1CalendarView1.Theme.Day.Weekend.BorderColor = Color.LightPink
C1CalendarView1.Theme.Titles.Day.BorderColor = Color.Crimson
C1CalendarView1.Theme.Titles.Week.BorderColor = Color.Crimson
C1CalendarView1.Theme.Titles.Weekend.BorderColor = Color.DarkCyan
C1CalendarView1.Theme.Common.BorderColor = Color.PeachPuff

' apply font settings to dates and titles in calendar
C1CalendarView1.Theme.Day.Bolded.Font =
    New Font("Microsoft Sans Serif", 8.25F,
             (FontStyle.Bold Or FontStyle.Underline))
C1CalendarView1.Theme.Day.Weekend.Font =
    New Font("Microsoft Sans Serif", 8.25F,
             FontStyle.Bold, GraphicsUnit.Point, 0)
C1CalendarView1.Theme.Titles.Day.Font =
    New Font("Microsoft Sans Serif", 8.25F,
             FontStyle.Bold, GraphicsUnit.Point, 0)
C1CalendarView1.Theme.Titles.Month.Font =
    New Font("Microsoft Sans Serif", 9.0F,
             (FontStyle.Bold Or FontStyle.Underline), GraphicsUnit.Point, 0)
C1CalendarView1.Theme.Titles.Week.Font =
    New Font("Microsoft Sans Serif", 6.0F,
             FontStyle.Bold)

' set foreground color for dates and titles in calendar
C1CalendarView1.Theme.Day.Bolded.ForeColor = Color.Crimson
C1CalendarView1.Theme.Day.Disabled.ForeColor = SystemColors.ControlText
C1CalendarView1.Theme.Day.Trail.ForeColor = Color.Transparent
C1CalendarView1.Theme.Day.Weekend.ForeColor = Color.Crimson
C1CalendarView1.Theme.Titles.Month.ForeColor = Color.Crimson

' set background color for dates and titles in calendar and for calendar as well
C1CalendarView1.Theme.Day.Bolded.BackColor = Color.Pink
C1CalendarView1.Theme.Day.Disabled.BackColor = Color.LightGray
C1CalendarView1.Theme.Day.Ordinary.BackColor = Color.White
```

```
C1CalendarView1.Theme.Day.Trail.BackColor = Color.White
C1CalendarView1.Theme.Titles.Day.BackColor = Color.PeachPuff
C1CalendarView1.Theme.Titles.Week.BackColor = Color.PeachPuff
C1CalendarView1.Theme.Day.Weekend.BackColor = Color.Pink
C1CalendarView1.Theme.Common.BackColor = Color.Linen
```

- **C#**

```
// set horizontal and vertical alignment for titles and calendar
c1CalendarView1.Theme.Titles.Week.HorizontalAlignment = C1.Framework.Alignment.Far;
c1CalendarView1.Theme.Common.HorizontalAlignment = C1.Framework.Alignment.Near;
c1CalendarView1.Theme.Titles.Week.VerticalAlignment = C1.Framework.Alignment.Far;
c1CalendarView1.Theme.Common.VerticalAlignment = C1.Framework.Alignment.Near;

// set borders for different dates in calendar
c1CalendarView1.Theme.Day.Bolded.Border = new C1.Framework.Thickness(2, 2, 2, 2);
c1CalendarView1.Theme.Day.Ordinary.Border = new C1.Framework.Thickness(1, 1, 1, 1);
c1CalendarView1.Theme.Day.Weekend.Border = new C1.Framework.Thickness(1, 1, 1, 1);

// set border color for dates and titles in calendar and for calendar as well
c1CalendarView1.Theme.Day.Bolded.BorderColor = Color.Crimson;
c1CalendarView1.Theme.Day.Ordinary.BorderColor = Color.PeachPuff;
c1CalendarView1.Theme.Day.Today.BorderColor = SystemColors.ControlLightLight;
c1CalendarView1.Theme.Day.Weekend.BorderColor = Color.LightPink;
c1CalendarView1.Theme.Titles.Day.BorderColor = Color.Crimson;
c1CalendarView1.Theme.Titles.Week.BorderColor = Color.Crimson;
c1CalendarView1.Theme.Titles.Weekend.BorderColor = Color.DarkCyan;
c1CalendarView1.Theme.Common.BorderColor = Color.PeachPuff;

// apply font settings to dates and titles in calendar
c1CalendarView1.Theme.Day.Bolded.Font =
    new Font ("Microsoft Sans Serif", 8.25F,
    (FontStyle.Bold | FontStyle.Underline));
c1CalendarView1.Theme.Day.Weekend.Font =
    new Font("Microsoft Sans Serif", 8.25F,
    FontStyle.Bold, GraphicsUnit.Point, 0);
c1CalendarView1.Theme.Titles.Day.Font =
    new Font("Microsoft Sans Serif", 8.25F,
    FontStyle.Bold, GraphicsUnit.Point, 0);
c1CalendarView1.Theme.Titles.Month.Font =
    new Font("Microsoft Sans Serif", 9F,
    (FontStyle.Bold | FontStyle.Underline), GraphicsUnit.Point, 0);
c1CalendarView1.Theme.Titles.Week.Font =
    new Font("Microsoft Sans Serif", 6F, FontStyle.Bold);

// set foreground color for dates and titles in calendar
c1CalendarView1.Theme.Day.Bolded.ForeColor = Color.Crimson;
c1CalendarView1.Theme.Day.Disabled.ForeColor = SystemColors.ControlText;
c1CalendarView1.Theme.Day.Trail.ForeColor = Color.Transparent;
c1CalendarView1.Theme.Day.Weekend.ForeColor = Color.Crimson;
c1CalendarView1.Theme.Titles.Month.ForeColor = Color.Crimson;

// set background color for dates and titles in calendar and for calendar as well
c1CalendarView1.Theme.Day.Bolded.BackColor = Color.Pink;
c1CalendarView1.Theme.Day.Disabled.BackColor = Color.LightGray;
c1CalendarView1.Theme.Day.Ordinary.BackColor = Color.White;
c1CalendarView1.Theme.Day.Trail.BackColor = Color.White;
c1CalendarView1.Theme.Titles.Day.BackColor = Color.PeachPuff;
c1CalendarView1.Theme.Titles.Week.BackColor = Color.PeachPuff;
c1CalendarView1.Theme.Day.Weekend.BackColor = Color.Pink;
c1CalendarView1.Theme.Common.BackColor = Color.Linen;
```

DateEdit Features

DateEdit offers several features to help you work with selected or entered dates.

Date Formats
: Learn how to display dates in predefined or custom formats in DateEdit.

Null Value and Watermark Support
: Learn how to handle null values and display watermark in DateEdit.

Keyboard Support
: Learn how to use several key combinations in DateEdit.

Masking
: Learn how to restrict user input through masking in DateEdit.

Data Validation
: Learn how to implement pre-validation and post-validation in DateEdit.

Value Formatting and Parsing
: Learn how to format and parse values in DateEdit.

Internationalization
: Learn how to change current culture settings and implement right-to-left support in DateEdit.

Appearance and Styling
: Learn how to customize the appearance of DateEdit.

Date Formats

The DateEdit control supports a number of date formats classified as follows:

- **Predefined formats**: DateEdit supports short date, long date, general date, short time, long time, scientific, percent, and several other predefined formats. You can display date and time in any of these formats by setting the FormatType property from FormatTypeEnum.
- **Custom formats**: DateEdit allows custom formatting of dates. You can specify custom formats and display dates in those formats. To enable custom formatting, set the FormatType property to **CustomFormat** from FormatTypeEnum. Further, to display dates in custom formats, set custom format specifiers in the CustomFormat property.

For information on custom format specifiers, see Custom Date and Time Format Strings. The DateEdit control supports majority of these format strings.

The following image shows the DateEdit control displaying date in a custom format.

The following code snippet shows how to display a date in a custom format in DateEdit.

- **Visual Basic**

```
' specify the format type as custom format
C1DateEdit1.FormatType = C1.Win.C1Input.FormatTypeEnum.CustomFormat

' set the custom format specifier
C1DateEdit1.CustomFormat = "MMM-dd-yyyy"
```

- **C#**

```
// specify the format type as custom format
c1DateEdit1.FormatType = C1.Win.C1Input.FormatTypeEnum.CustomFormat;

// set the custom format specifier
c1DateEdit1.CustomFormat = "MMM-dd-yyyy";
```

Null Value and Watermark Support

Null values can be difficult to handle without appropriate rules or settings. However, DateEdit provides flexible rules for handling null values and controlling relevant aspects in a variety of cases.

In ReadOnly mode

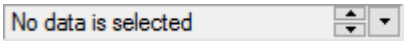
The DateEdit control allows displaying null values through the NullText property of DisplayFormat when the control is in ReadOnly mode. It is possible to override the NullText property in DisplayFormat to display watermark in the DateEdit control.

In edit mode

This is default mode of DateEditor. In edit mode (when ReadOnly is false), DateEdit allows displaying null values by setting the NullText property of EditFormat.

In both modes, it is possible to display empty strings by setting the EmptyAsNull property to **true**. When editing date time values in DateEdit with DateTimeInput set to **true,** the control appears empty representing the null value. As you start editing values in control, the control turns to the last non-null value assigned to it or today's date.

The following image shows DateEdit displaying null text (watermark) in the ReadOnly mode.



The following code snippet shows how to show null text in DateEdit in ReadOnly mode.

- **Visual Basic**

```
' set calendar pop-up as ReadOnly
C1DateEdit1.[ReadOnly] = True

' display null text in DateEdit
C1DateEdit1.DisplayFormat.NullText = "No data is selected"
```

- **C#**

```
//set calendar pop-up as ReadOnly
c1DateEdit1.ReadOnly = true;

// display null text in DateEdit
c1DateEdit1.DisplayFormat.NullText = "No data is selected";
```

Keyboard Support

The following table lists the applicable key combinations and their corresponding actions in the DateEdit control.

Keyboard Command	Action
Up	Increases the selected value by one.
Down	Decreases the selected value by one.

Left	Shifts the selection by one position in the left direction.
Right	Shifts the selection by one position in the right direction.
Home	Shifts the selection to the first value in the combo box.
End	Shifts the selection to the last value in the combo box.
Shift + Up/Left	Selects contiguous values from the current selection by one position in the left direction .
Shift + Down/Right	Selects contiguous values from the current selection by one position in the right direction.
Shift + Home	Selects all values from the current selection to the first value in the combo box.
Shift + End	Selects all values from the current selection to the last value in the combo box.
Ctrl + Up/Left	Causes the cursor to skip values of the same group in the left direction.
Ctrl + Down/Right	Causes the cursor to skip values of the same group in the right direction.
Ctrl + Home	Shifts the cursor to the the first day from the current selection.
Ctrl + End	Shifts the cursor to the the last day from the current selection.

Masking

Masking is helpful in restricting user input in editors. In the DateEdit control, masking input allows entering input in a specific format. For instance, the DateEdit control should only accept numeric values or date and time values in a 24-hour format. To restrict users to input values in such formats, masking can be used in the DateEdit control.

The following image shows the DateEdit control with masking applied in the edit mode.

DateEdit allows masking input through the EditMask property. To enable masked input, set the EditMask property to a mask string comprising placeholders and literals. The control allows controlling other aspects of masked input through the MaskInfo property. It is also possible to define your own placeholders consisting of user-defined mask characters by using the CustomPlaceholders property. Note that masking does not work if the DateEdit control is in ReadOnly mode.

The following table lists all placeholders and literals supported by DateEdit.

Placeholder	Description
#	Digit placeholder permits a numeric character or a plus or minus sign in this position (entry optional).
.	Decimal placeholder. The actual character used is the one specified as the decimal placeholder in your international settings. This character is treated as a literal for masking purposes.
,	Thousands separator. The actual character used is the one specified as the thousands separator in your international settings. This character is treated as a literal for masking purposes.

:	Time separator. The actual character used is the one specified as the time separator in your international settings. This character is treated as a literal for masking purposes.
/	Date separator. The actual character used is the one specified as the date separator in your international settings. This character is treated as a literal for masking purposes.
\	Treat the next character in the mask string as a literal. This allows you to include the #, &, A, … characters in the mask. This character is treated as a literal for masking purposes.
&	Character placeholder (entry required). Any character is permitted.
>	Convert all the characters that follow to uppercase.
<	Convert all the characters that follow to lowercase.
~	Turns off the previous < or >.
!	Causes the optional characters that follow in the edit mask to display from right to left, rather than from left to right. So, blanks appear on the left.
^	Turns off the previous ! character. After ^, blanks appear on the right.
A	Alphanumeric character placeholder (entry required). For example: a – z, A – Z, or 0 – 9.
a	Alphanumeric character placeholder (entry optional).
0	Digit placeholder (entry required). For example: 0 – 9.
9	Digit placeholder (entry optional).
C	Character or space placeholder (entry optional). Any character is permitted.
L	Letter placeholder (entry required). For example: a – z or A – Z.
?	Letter placeholder (entry optional).
\n	New line literal. It is applicable when **Multiline** property is set to **True**.
"	All characters in a string enclosed in double quotes are considered as literals.
Literal	All other symbols are displayed as literals; that is, as themselves.

The following code snippet shows how to set the EditMask property to apply masking in DateEdit.

- **Visual Basic**

```
' set the EditMask propery to apply masking
C1DateEdit1.EditMask = "00-LLL-9900"
```

- **C#**

```
// set the EditMask propery to apply masking
c1DateEdit1.EditMask = "00-LLL-9900";
```

Data Validation

The DateEdit control supports data validation of two types, pre-validation and PostValidation. pre-validation is data validation of the raw input string, while PostValidation is that of the typed value entered by the user.

The following sections describe pre-validation and post-validation in DateEdit.

Pre-validation
: Learn how to implement pre-validation in DateEdit.

Post-validation
: Learn how to implement post-validation in DateEdit.

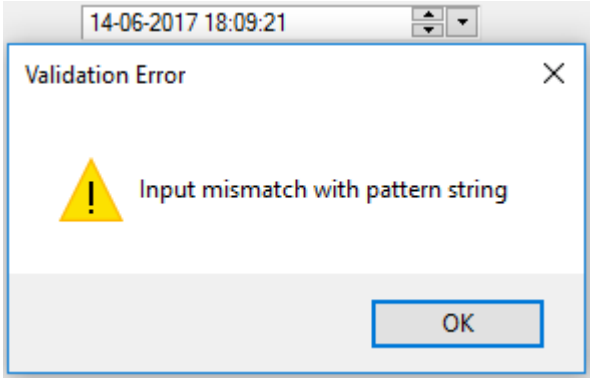
Pre-validation

Pre-validation is raw string validation, which means that rules are applied before parsing.

DateEdit allows controlling pre-validation of input string through the PreValidation property. The control provides various pre-validation options, as follows:

- **Case sensitive comparisons**: Allows you to make string comparisons case sensitive. To enable case sensitive comparisons, set the CaseSensitive property to **true**.
- **Validation pattern**: Allows you to specify string containing the validation pattern or rules. To specify the validation pattern, set the PatternString property. Specify multiple rules or sub-strings by using an item separator in PatternString. DateEdit supports (| by default) as the item separator for multiple strings. However, to modify the item separator, set the ItemSeparator property.
- **Validation method**: Allows you to specify the method of validation. To specify validation method, set the Validation property to any of the following values from PreValidationTypeEnum:
  - **ExactList**: PatternString contains a list of possible values separated by ItemSeparator.
  - **PreValidatingEvent**: PreValidating event is used in validation.
  - **Wildcards**: PatternString contains a list of wildcard patterns separated by the ItemSeparator. The following characters are reserved in a pattern: ? (any single character), # (any single digit), * (zero or more characters), \ (escape). You can also define your own custom pattern characters using the pre-validation property.
  - **RegexPattern**: PatternString contains a regular expression.
- **Error message**: Allows you to set the message that appears when input string does not match validation pattern. To set error message, set the ErrorMessage property.

The following image shows the DateEdit control displaying validation error message.



The following code snippet shows how to set various properties to apply pre-validation in DateEdit.

- **Visual Basic**

```
' Apply PreValidation in DateEdit
C1DateEdit1.PreValidation.CaseSensitive = True
C1DateEdit1.PreValidation.ErrorMessage = "Input mismatch with pattern string"
C1DateEdit1.PreValidation.ItemSeparator = "|"
C1DateEdit1.PreValidation.PatternString = "00|11"
C1DateEdit1.PreValidation.Validation = C1.Win.C1Input.PreValidationTypeEnum.ExactList
```

- **C#**

```
// apply PreValidation in DateEdit
c1DateEdit1.PreValidation.CaseSensitive = true;
c1DateEdit1.PreValidation.ErrorMessage = "Input mismatch with pattern string";
c1DateEdit1.PreValidation.ItemSeparator = "|";
c1DateEdit1.PreValidation.PatternString = "00|11";
c1DateEdit1.PreValidation.Validation = C1.Win.C1Input.PreValidationTypeEnum.ExactList;
```

Post-validation

Post-validation is validation rules applied after parsing. It allows validating the value entered by the user.

In DateEdit, post-validation is controlled through the PostValidation property. The control provides various options for post-validation , as follows:

- **Predefined values**: Allows you to check whether input values match values in predefined values list or not. To specify predefined values, set the Values property.
- **Excluded values**: Allows you to specify values not permitted as input values. To set these values, set the ValuesExcluded property.
- **Intervals**: Allows you to check if input values belong to a specific interval. DateEdit supports even multiple intervals that you can specify by using the Intervals property.
- **Validation method**: Allows you to specify the method of validation. Set the Validation property to **ValuesAndIntervals** or **PostValidatingEvent** for declarative and programmatic validation, respectively.

The following code snippet shows how to set various properties to apply post-validation in DateEdit.

- **Visual Basic**

```
' apply PostValidation in DateEdit
C1DateEdit1.PostValidation.Validation = C1.Win.C1Input.PostValidationTypeEnum.ValuesAndIntervals
C1DateEdit1.PostValidation.Values = New ValueType() {1, 2, 4}
'New () {1, 2, 4}
C1DateEdit1.PostValidation.ErrorMessage = "Wrong"
C1DateEdit1.PostValidation.CaseSensitive = False
```

- **C#**

```
// apply PostValidation in DateEdit
c1DateEdit1.PostValidation.Validation = C1.Win.C1Input.PostValidationTypeEnum.ValuesAndIntervals;
c1DateEdit1.PostValidation.Values = new[] { 1, 2, 4 };
c1DateEdit1.PostValidation.ErrorMessage = "Wrong";
c1DateEdit1.PostValidation.CaseSensitive = false;
```

Value Formatting and Parsing

Formatting

The DateEdit control allows controlling value formatting through the FormatType property. Its enumerated values define how data will be formatted in the control. Some of the options correspond to .NET standard format specifiers for numeric and date-time types, for example, StandardNumber and LongDate. For information about supported date formats in DateEdit, see Date formats.

Besides predefined formats, there is custom format that corresponds to the case of a custom format specifier determined by the CustomFormat property. In addition, there is a special format type, UseEvent, which delegates the formatting to the Formatting event. Furthermore, the ability to represent NULL values (System.DBNull) is controlled by the NullText and the EmptyAsNull property. For information about handling null values in DateEdit, see Null value and watermark support.

Sometimes, it is important to trim leading or trailing spaces when showing formatted values. To trim leading and

trailing spaces, set the TrimStart and the TrimEnd property respectively.

DateEdit supports two formats, one for display (when the control is read-only or is not in the edit mode), and another for edit mode. The formatting modes are accessible through the DisplayFormat and the EditFormat property.

Parsing

Converting the string entered by the user to the data type is called parsing. It is the opposite of formatting. The DateEdit control allows controlling parsing through the ParseInfo property. The ParseInfo property provides access to the ParseInfo Class that contains sub-properties that control different aspects of parsing.

Though default parsing suffices in a majority of cases, you can still change many aspects of how the control parses, by expanding the ParseInfo property, changing the (Inherit) flags, and setting desired properties. By default, the same format property value is used for parsing as for formatting.

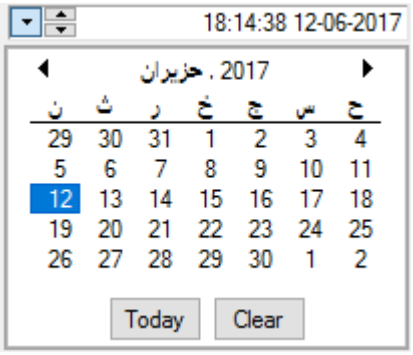
Internationalization

DateEdit supports internationalization in the following ways:

- **Culture settings**: DateEdit enables formatting, parsing, and validating data on the basis of a specific culture. The control allows specifying culture(s) through Culture property of C1TexBox class.
- **Right-to-left support**: DateEdit provides right-to-left support for languages that follow right-to-left scripts. To enable right-to-left functionality in the DateEdit control, set the **RightToLeft** property from RightToLeft Control in Systems.Windows.Forms.

For information about setting culture settings and right-to-left functionality in CalendarView, see Culture Settings and Right to Left Support in CalendarView Features.

The following image shows DateEdit with Arabic culture in right-to-left direction.



The following code snippet shows how to set the current culture and enable right-to-left functionality in the DateEdit control.

- **Visual Basic**

```
' set culture
C1DateEdit1.Culture = 2048

' enable right to left functionality
C1DateEdit1.RightToLeft = RightToLeft.Yes
```

- **C#**

```
// set the culture
c1DateEdit1.Culture= 2049;


// enable right to left functionality
```

```
c1DateEdit1.RightToLeft = RightToLeft.Yes;
```

Appearance and Styling

It is possible to customize the appearance of the DateEdit control to better fit the applications that use the control. DateEdit offers the following styling features that allow changing bacground and foreground colors, adding borders, changing text alignment, and setting theme.

- **Common styles:** Apply common styles including border color, border style, font settings, background and foreground colors etc. to the DateEdit control.
  - **Border color**: Change the color of the border by setting the BorderColor property.
  - **Border style**: Set the border style of the control by setting the BorderStyle property.
  - **Font settings**: Apply font settings to the text in the DateEdit control by setting the **Font** property from Font Control in Systems.Windows.Forms.
  - **Background color**: Change the background color of DateEdit by setting the BackColor property.
  - **Foreground color**: Apply foreground color to DateEdit by setting the ForeColor property.
- **Text alignment:** Set the alignment of the text in the DateEdit control by setting the **TextAlign** property from TextBox in Systems.Windows.Forms.
- **Themes**: Apply different themes to enhance the overall look of DateEdit by setting the VisualStyle property.

For information about customizing the appearance of calendar pop-up or CalendarView, see Appearance and Styling in CalendarView Features.

The following image displays styling features applied to the DateEdit control.

The following code snippet shows how to implement styling features in CalendarView using relevant properties.

- **Visual Basic**

```
' set the border color
C1DateEdit1.BorderColor = Color.DarkBlue

' set the border style
C1DateEdit1.BorderStyle = BorderStyle.FixedSingle

' apply font settings
C1DateEdit1.Font = New Font("Microsoft Sans Serif", 9.0F, FontStyle.Italic)

' set foreground and background colors
C1DateEdit1.ForeColor = Color.Brown
C1DateEdit1.BackColor = Color.LightCyan

' set text alignment
C1DateEdit1.TextAlign = HorizontalAlignment.Left
```

- **C#**

```
// set the border color
c1DateEdit1.BorderColor = Color.DarkBlue;

// set the border style
c1DateEdit1.BorderStyle = BorderStyle.FixedSingle;

// apply font settings
c1DateEdit1.Font = new Font("Microsoft Sans Serif", 9F, FontStyle.Italic);

// set foreground and background colors
c1DateEdit1.ForeColor = Color.Brown;
```

```
c1DateEdit1.BackColor = Color.LightCyan;

// set text alignment
c1DateEdit1.TextAlign = HorizontalAlignment.Left;
```

CalendarView Samples

C1Studio installer provides to that help you understand the product and its implementation. CalendarView sample is available in the installed folder - **Documents\ComponentOne Samples\WinForms\CalendarView**.

Sample	Description
Layout	Includes a sample that demonstrates how to change layout and apply various styles in CalendarView.

DateEdit Samples

C1Studio installer provides to that help you understand the product and its implementation. DateEdit sample is available in the installed folder - **Documents\ComponentOne Samples\WinForms\DateEdit**.

Sample	Description
Layout	Includes a sample that demonstrates how to change layout and apply various styles in DateEdit.