ComponentOne

DateTimeEditors for UWP

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)

Table of Contents

DateTime Editors for UWP

Get six fundamental date and time editors including both classic and Modern UI-inspired designs. The **C1DateSelector**, **C1TimeSelector** and **C1DateTimeSelector** controls provide a series of drop-downs to collect **DateTime** input. The C1DatePicker control provides a drop-down calendar in the most familiar fashion.
The C1DateTimePicker control combines C1DatePicker and C1TimeEditor into one colossal control.

Getting Started with UWP Edition

Help with UWP Edition

Getting Started

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

Key Features

DateTimeEditors for UWP allows you to create customized, rich applications. Make the most of **DateTimeEditors for UWP** by taking advantage of the following key features:

- **Multiple Display Versions**

  Choose one of the available edit modes: **DateTime (default)**, **Date**, or **Time**. Select from preset date formats, including **Short** and **Long**. Choose from preset time formats, including **ShortTime**, **LongTime**, and **TimeSpan**.

- **Supports Spin Buttons**

  The **C1DateTimePicker** and **C1TimeEditor** controls support spin (up/down) buttons for selecting date and time.

- **Supports Null Values**

  The C1DateTimePicker, C1DatePicker and C1TimeEditor controls allow you to enter null values, by default. This can be disabled by setting the AllowNull property to **False**.

C1DateTime Selector Help

Exchange date and time information using **DateTimeSelector for UWP**. It provides a simple and intuitive UI for selecting date and time, just date values, or just time values. The date and time can be selected by using the buttons or by using the keyboard arrows.

C1DateTime Selector Quick Start

The following quick start guide is intended to get you up and running with **DateTimeSelector for UWP**. In this quick start, you'll start in Visual Studio to create a new project, add a C1DateTimeSelector control to your application, and customize the C1DateTimeSelector control.

Step 1 of 3: Creating an Application with C1DateTime Selector Control

In this step, you'll begin in Visual Studio to create a UWP-style application using **C1DateTimeSelector for UWP**.

Complete the following steps:

1. In Visual Studio, Select **File | New | Project**.
2. Select Templates | Visual C# | Windows | Universal. From the templates list, select **Blank App (Universal Windows)**. Enter a **Name** and click **OK** to create your project.
3. Open MainPage.xaml if it isn't already open, place the cursor between the <Grid> and </Grid> tags, and click once.
4. Navigate to the Toolbox and double-click the C1DateTimeSelector icon to add the control to the grid. The XAML markup resembles the following:

Markup

```
<Grid Background="{ThemeResource ApplicationPageBackgroundThemeBrush}"
x:Name="LayoutRoot">
    <Xaml:C1DateTimeSelector HorizontalAlignment="Left"
VerticalAlignment="Top"/>
</Grid>
```

You have successfully created a UWP-style application containing a **C1DateTimeSelector** control. In the next step, you will modify the appearance of the control.

Step 2 of 3: Customizing C1DateTime Selector Control

In the previous step, you created a UWP-style application with a C1DateTimeSelector control. In this step, you will modify the appearance of the control.

Complete the following steps:

1. Locate the **Brush** Category in your application's Properties window and click to display the options. You can set custom colors for the following properties: **Background**, **BorderBrush**, and **Foreground**.

   You can also change the appearance of your C1DateTimeSelector control by inserting the following XAML markup into the <Xaml:C1DateTimeSelector/> tag:

Markup

```
Background="#CC66709C" Foreground="White"
```

2. Customize the control's Text properties to change the **FontFamily** to Plantagenet Cherokee and the **FontSize** to 20.

   You can also change the **FontFamily** and **FontSize** by inserting the following XAML markup into the <Xaml:C1DateTimeSelector/> tag:

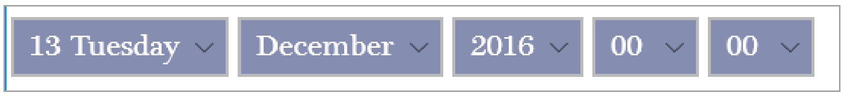
   Markup

   ```
   FontFamily="Plantagenet Cherokee" FontSize="20"
   ```

Now that you've customized the application, you can run the project and observe the run time behaviors of the control.

Step 3 of 3: Running the Project

In the last step, you customized the C1DateTimeSelector control. In this step, you will run the project and observe some of the run-time features of the control. Your C1DateTimeSelector control should resemble the following image:



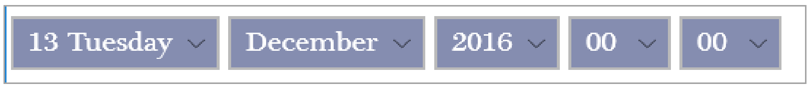
Congratulations – you have completed the **C1DateTimeSelector for UWP** quick start!

Working with C1DateTimeSelector

The following topics will provide you with an overview of the C1DateTimeSelector control's elements and features.

C1DateTimeSelector Elements

DateTimeEditors for UWP includes the control, a simple control which provides, by default, both a date selector and a time selector. When you add the C1DateTimeSelector control to a XAML window, it exists as a completely functional date and time selector. By default, the control's interface looks similar to the following image:



The C1DateTimeSelector control consists of the following elements:

- **Date Selector**

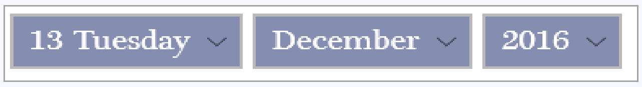
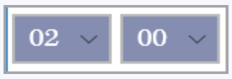
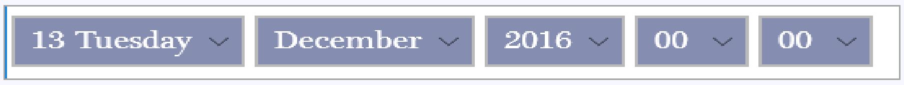
  The Date Selector is comprised of three fields: a month field, a date and day selector, and a year selector. You can set the month, date and day, and year using the drop-down arrows and choosing the month, day and date, and year from the drop-down lists.

- **Time Selector**

  The Time Selector is comprised of three fields: an hour field, a minute field, and an AM/PM field. You can set the hour, the minute, and the AM/PM setting by using the drop-down arrows and choosing the hour, the minute, and the AM/PM setting from the drop-down list.

C1DateTimeSelector Edit Modes

By default, the C1DateTimeSelector control will appear on the page with both the date selector and the time selector. You can change the selectors that are displayed by setting the C1DateTimeSelector.EditMode property to **Date**, **Time**, or **DateTime**. You can set the C1DateTimeSelector.EditMode property to **Date** to display only the date selector; you can set the C1DateTimeSelector.EditMode property to **Time** to only display the time selector; and you can set the C1DateTimeSelector.EditMode property to **DateTime** to display both the time selector and date selector. The table below illustrates each editor mode.

Edit Mode	Result
Date	 13 Tuesday December 2016
Time	 02 00
DateTime	 13 Tuesday December 2016 00 00

C1DateTimeSelector Task-Based Help

The task-based help assumes that you are familiar with programming in Visual Studio and know how to use the C1DateTimeSelector control in general. If you are unfamiliar with the C1DateTimeSelector control, please see the C1DateTimeSelector Quick Start first.

Each topic in this section provides a solution for specific tasks using the C1DateTimeSelector product.

Each task-based help topic also assumes that you have created a new project.

Creating a C1DateTimeSelector Control in Code

This topic will walk you through creating a **C1DateTimeSelector** control and applying some basic styling to the control via code.

Follow these steps:

1. Open MainPage.xaml if it isn't already open and locate the <Grid> </Grid> tag set. Edit the markup so that it resembles the following markup:

Markup

```
<Grid Background="{StaticResource ApplicationPageBackgroundThemeBrush}"
x:Name="LayoutRoot">
</Grid>
```

2. Right-click your page and select **View Code** from the list to switch to code view.
3. Add the following namespaces to your application

C#

```
using C1.Xaml;
```

Visual Basic

```
Imports C1.Xaml
```

3. Insert the following constructor directly below the **InitializeComponent()** method:

C#

```
C1DateTimeSelector c1DTS1 = new C1DateTimeSelector();
```

Visual Basic

```
Dim c1DTS1 As C1DateTimeSelector = New C1DateTimeSelector
```

4. Add the following code inside the constructor to control the width and height of the C1DateSelector control:

C#

```
c1DTS1.Height = 55;
c1DTS1.Width = 600;
```

Visual Basic

```
c1DTS1.Height = 55
c1DTS1.Width = 600
```

5. Insert the final line of code to display the C1DateSelector:

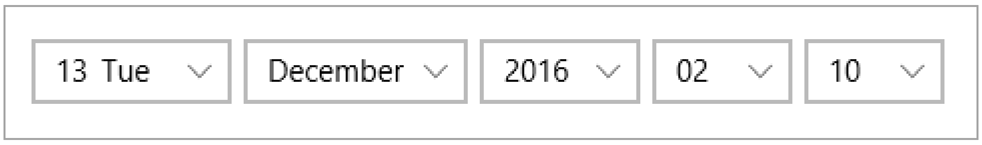
C#

```
LayoutRoot.Children.Add(c1DTS1);
```

Visual Basic

```
LayoutRoot.Children.Add(c1DTS1)
```

6. Run your application. The C1DateTimeSelector control should resemble the following image:



Selecting the Edit Mode

By default, the **C1DateTimeSelector** control shows both the date and time selectors, but you may also choose to show only the date selector or only the time selector. In this topic, you will learn how to change the editor mode in in XAML and in code.

In XAML

To select the edit mode, edit the XAML markup so it resembles one of the following samples:

Markup
```
<Xaml:C1DateTimeSelector EditMode="DateTime"/>
```

Markup
```
<Xaml:C1DateTimeSelector EditMode="Date"/>
```

Markup
```
<Xaml:C1DateTimeSelector EditMode="Time"/>
```

In Code

To select the edit mode, add the following code:

C#
```
c1DTS1.EditMode = C1.Xaml.C1DateTimePickerEditMode.Date;
```

Visual Basic
```
c1DTS1.EditMode = C1.Xaml.C1DateTimePickerEditMode.Date
```

Specifying the Selected Date

You can specify the time and date of a C1DateTimeSelector control by setting the **SelectedDate** property in XAML or code.

In XAML

To specify the initially selected date, edit your XAML markup to resemble the following:

Markup
```
<Xaml:C1DateTimeSelector SelectedDate="1932, 10, 12/>
```

In Code

C#
```
c1DTS1.SelectedDate = new DateTime( 1932, 10, 12 );
```

Visual Basic
```
c1DTS1.SelectedDate = New DateTime(1932, 10, 12)
```

Setting the Minimum and Maximum Calendar Dates

You can change the dates that the user can choose by setting the **MinimumDate** and **MaximumDate** properties in code.

Add the following code to your application to set the minimum and maximum calendar dates:

C#

```
c1DTS1.MinDate = new DateTime( 1915, 06, 15);
c1DTS1.MaxDate = new DateTime( 2015, 06, 15);
```

Visual Basic

```
 c1DTS1.MinDate = New DateTime(1915, 6, 15)
 c1DTS1.MaxDate = New DateTime(2015, 6, 15)
```

C1DateSelector Help

Display and choose date information using **DateSelector for UWP**. The **C1DateSelector** control provides a simple and intuitive UI for selecting date values. Click the **C1DateSelector's** drop-down arrows and select the month, the date and day, and the year.

C1DateSelector Quick Start

The following quick start guide is intended to get you up and running with **DateSelector for UWP**. In this quick start, you'll start in Visual Studio to create a new project, add a **C1DateSelector** control to your application, and customize the **C1DateSelector** control.

Step 1 of 3: Creating an Application with a C1DateSelector Control

In this step, you'll begin in Visual Studio to create a UWP-style application using **C1DateSelector for UWP**.

Complete the following steps:

1. In Visual Studio, Select **File | New | Project**.
2. Select Templates | Visual C# | Windows | Universal. From the templates list, select **Blank App (Universal Windows)**. Enter a **Name** and click **OK** to create your project.
3. Open MainPage.xaml if it isn't already open, place the cursor between the <Grid> and </Grid> tags, and click once.
4. Navigate to the Toolbox and double-click the **C1DateSelector** icon to add the control to the grid. The XAML markup resembles the following:

Markup

```
<Grid x:Name="LayoutRoot">
<Xaml:C1DateSelector/>
</Grid>
```

You have successfully created a UWP-style application containing a C1DateSelector control. In the next step, you will modify the appearance of the control.

Step 2 of 3: Customizing the C1DateSelector Control

In the previous step, you created a UWP-style application with a **C1DateSelector** control. In this step, you will modify the appearance of the control.

Complete the following steps:

1. Locate the **Brush** Category in your application's Properties window and click to display the options. You can set custom colors for the following properties: **Background**, **BorderBrush**, and **Foreground**.

   You can also change the appearance of your **C1DateSelector** control by inserting the following XAML markup into the <Xaml:C1DateTimeSelector/> tag:

Markup

```
Background="#CC96BB78" BorderBrush="#CCC7C7C7" Foreground="White"
```

2. Customize the control's Text properties to change the **FontFamily** to Andalus and the **FontSize** to 20 px.

   You can also change the **FontFamily** and **FontSize** by inserting the following XAML markup into the <Xaml:C1DateTimeSelector/> tag:

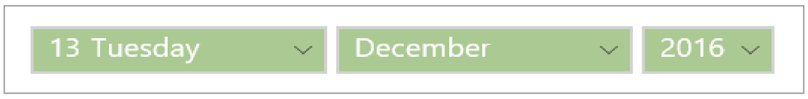
   Markup

   ```
   FontFamily="Andalus" FontSize="20"
   ```

Now that you've customized the application, you can run the project and observe the run time behaviors of the control.

Step 3 of 3: Running the Project

In the last step, you customized the **C1DateSelector** control. In this step, you will run the project and observe some of the run-time features of the control. Your **C1DateSelector** control should resemble the following image:



Congratulations!

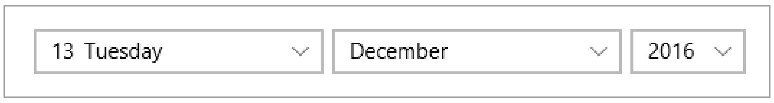
You have completed the **C1DateSelector for UWP** quick start!

Working with C1DateSelector

The following topics will provide you with an overview of the **C1DateSelector** control's elements and features.

C1DateSelector Elements

DateTimeEditors for UWP includes the **C1DateSelector** control, a simple control which provides a date selector that, when clicked at run time, allows you to choose a date from a drop-down list. When you add the **C1DateSelector** control to a XAML window, it exists as a completely functional date selector. By default, the control's interface looks similar to the following image:



The **C1DateSelector** control consists of the following elements:

- **Date and Day Selector**
  You can use the drop-down arrow to choose the date and day from the drop-down list.
- **Month Selector**
  You can use the drop-down arrow to choose a month from the drop-down list.
- **Year Selector**
  You can use the drop-down arrow to choose a year from the drop-down list.

C1DateSelector Task-Based Help

The task-based help assumes that you are familiar with programming in Visual Studio and know how to use the **C1DateSelector** control in general. If you are unfamiliar with the **C1DateSelector** control, please see the C1DateSelector Quick Start first.

Each topic in this section provides a solution for specific tasks using the **C1DateSelector** product.

Each task-based help topic also assumes that you have created a new project.

Creating a C1DateSelector Control in Code

This topic will walk you through creating a **C1DateSelector** control and applying some basic styling to the control via code.

Follow these steps:

1. Open MainPage.xaml if it isn't already open and locate the <Grid> </Grid> tag set. Edit the markup so that it resembles the following markup:

XAML

```
<Grid Background="{StaticResource ApplicationPageBackgroundThemeBrush}"
x:Name="LayoutRoot">
</Grid>
```

2. Right-click your page and select **View Code** from the list to switch to code view.
3. Add the following namespaces to your application:

C#

```
using C1.Xaml;
```

Visual Basic

```
Imports C1.Xaml
```

4. Insert the following constructor just below the **InitializeComponent()** method:

C#

```
C1DateSelector c1dateselect1 = new C1DateSelector();
```

Visual Basic

```
Dim c1dateselect1 As C1DateSelector = New C1DateSelector
```

5. Add the following code inside the constructor to control the width and height of the **C1DateSelector**:

C#

```
c1dateselect1.Height = 55;
c1dateselect1.Width = 350;
```

Visual Basic

```
c1dateselect1.Height = 55
```

```
c1dateselect1.Width = 350
```

6. Insert the final line of code to display the **C1DateSelector**:

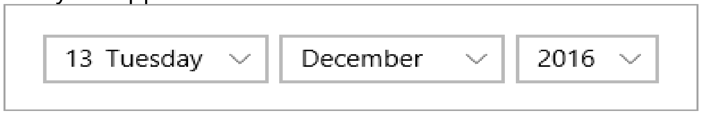
C#

```
LayoutRoot.Children.Add(c1dateselect1);
```

Visual Basic

```
LayoutRoot.Children.Add(c1dateselect1)
```

7. Run your application. The C1DateSelector control should resemble the following image:



Setting the C1DateSelector Selected Date

This topic will take you through setting the **SelectedDate** Property in XAML markup and in code.

In XAML

Add the following XAML markup to the <Xaml:C1DateSelector/> tag to set the **SelectedDate** property:

Markup

```
SelectedDate="1932, 01, 10"
```

In Code

Use the following code to set the SelectedDate property:

C#

```
c1dateselect1.SelectedDate = new DateTime(1932, 01, 10);
```

Visual Basic

```
c1dateselect1.SelectedDate = New DateTime(1932, 1, 10)
```

Setting the First Year Property

This topic will take you through setting the **FirstYear** property in XAML markup and in code.

In XAML

Add the following XAML markup to the <Xaml:C1DateSelector/> tag to set the **FirstYear** property:

Markup

```
FirstYear="500"
```

In Code

Use the following code to set the **FirstYear** property:

C#

```
c1dateselect1.FirstYear = 500;
```

Visual Basic

```
c1dateselect1.FirstYear = 500
```

Setting the Last Year Property

This topic will take you through setting the LastYear property in XAML markup and in code.

In XAML

Add the following XAML markup to the <Xaml:C1DateSelector/> tag to set the LastYear property:

Example Title

```
LastYear="1950"
```

In Code

Use the following code to set the LastYear property:

C#

```
c1dateselect1.LastYear = 1950;
```

Visual Basic

```
c1dateselect1.LastYear = 1950
```

C1TimeSelector Help

Exchange time information with an end-user using **TimeSelector for UWP**. It provides a simple interface for selecting time values. The time can be selected by using the dropdown arrows or by using the keyboard arrows.

C1TimeSelector Quick Start

The following quick start guide is intended to get you up and running with **TimeSelector for UWP**. In this quick start, you'll start in Visual Studio to create a new project, add a **C1TimeSelector** control to your application, and customize the **C1TimeSelector** control.

Step 1 of 3: Creating an Application with a C1TimeSelector Control

In this step, you'll begin in Visual Studio to create a UWP-style application using **C1TimeSelector for UWP**.

Complete the following steps:

1. In Visual Studio, Select **File | New | Project**.
2. Select Templates | Visual C# | Windows | Universal. From the templates list, select **Blank App (Universal Windows)**. Enter a **Name** and click **OK** to create your project.
3. Open MainPage.xaml if it isn't already open, place the cursor between the <Grid> and </Grid> tags, and click once.
4. Navigate to the Toolbox and double-click the C1TimeSelector icon to add the control to the grid. The XAML markup resembles the following:

   Markup
   ```
   <Grid x:Name="LayoutRoot">
   <Xaml:C1TimeSelector/>
   </Grid>
   ```

You have successfully created a UWP-style application containing a **C1TimeSelector** control. In the next step, you will modify the appearance of the control.

Step 2 of 3: Customizing the C1TimeSelector Control

In the previous step, you created a UWP-style application with a C1TimeSelector control. In this step, you will modify the appearance of the control.

Complete the following steps:

1. Locate the **Brush** Category in your application's Properties window and click to display the options. You can set custom colors for the following properties: **Background**, **BorderBrush**, and **Foreground**. You can also set the Linear Gradient for the **Background**.

   You can also change the appearance of your C1DateTimeSelector control by inserting the following XAML markup into the <Xaml:C1DateTimeSelector/> tag. Note that this XAML markup includes the <Xaml:C1TimeSelector.Background> tag:

Markup

```
<Xaml:C1TimeSelector HorizontalAlignment="Left" VerticalAlignment="Top"
Margin="22,55,0,0" Width="302" BorderBrush="#CC9CC391" Foreground="Beige">
  <Xaml:C1TimeSelector.Background>
    <LinearGradientBrush EndPoint="0.5,1" StartPoint="0.5,0">
      <GradientStop Color="#FF1C3E1F" />
      <GradientStop Color="#FF44A24C" Offset="0.869" />
    </LinearGradientBrush>
  </Xaml:C1TimeSelector.Background>
</Xaml:C1TimeSelector>
```

2. Customize the control's Text properties to change the **FontFamily** to Calibri, the **FontSize** to 18 px, and the **FontWeight** to Bold.

   You can also change the **FontFamily** and **FontSize** by inserting the following XAML markup into the <Xaml:C1DateTimeSelector/> tag:

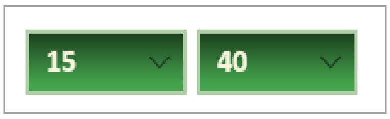
   Markup

   ```
   FontFamily="Calibri" FontSize="18" FontWeight="Bold"
   ```

Now that you've customized the application, you can run the project and observe the run time behaviors of the control.

Step 3 of 3: Running the Project

In the last step, you customized the **C1TimeSelector** control. In this step, you will run the project and observe some of the run-time features of the control. Your **C1TimeSelector** control should resemble the following image:



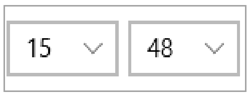
Congratulations – you have completed the **C1TimeSelector for UWP** quick start!

Working with C1TimeSelector

The following topics will provide you with an overview of the **C1TimeSelector** control's elements and features.

C1TimeSelector Elements

DateTimeEditors for UWP includes the **C1TimeSelector** control, a simple control which provides a time selector. When you add the **C1TimeSelector** control to a XAML window, it exists as a completely functional time selector. By default, the control's interface looks similar to the following image:



The **C1TimeSelector** control consists of the following elements:

- **Hour Selector**

  You can use the drop-down arrow to choose an hour from the drop-down list.

- **Minute Selector**

  You can use the drop-down arrow to choose a minute from the drop-down list.

C1TimeSelector Task- Based Help

The task-based help assumes that you are familiar with programming in Visual Studio and know how to use the **C1TimeSelector** control in general. If you are unfamiliar with the **C1TimeSelector** control, please see the **C1TimeSelector** Quick Start first.

Each topic in this section provides a solution for specific tasks using the **C1TimeSelector** product.

Each task-based help topic also assumes that you have created a new project.

Creating a C1TimeSelector Control in Code

This topic will walk you through creating a **C1TimeSelector** control and applying some basic styling to the control via code.

Follow these steps:

1. Open MainPage.xaml if it isn't already open and locate the <Grid> </Grid> tag set. Edit the markup so that it resembles the following markup:

   Markup
   ```
   <Grid Background="{StaticResource ApplicationPageBackgroundThemeBrush}"
   x:Name="LayoutRoot">
   </Grid>
   ```

2. Right-click your page and select **View Code** from the list to switch to code view.
3. Add the following namespaces to your application

   C#
   ```
   using C1.Xaml;
   ```

   Visual Basic
   ```
   Imports C1.Xaml
   ```

4. Insert the following constructor directly below the **InitializeComponent()** method:

   C#
   ```
   C1TimeSelector c1timeselect1 = new C1TimeSelector();
   ```

   Visual Basic
   ```
   Dim c1timeselect1 As C1TimeSelector = New C1TimeSelector
   ```

5. Add the following code below the constructor to control the width and height of the C1TimeSelector:

C#
```
c1timeselect1.Height = 55;
c1timeselect1.Width = 350;
```

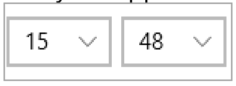
Visual Basic
```
c1timeselect1.Height = 55
c1timeselect1.Width = 350
```

6. Insert the final line of code to display the **C1TimeSelector**:

C#
```
LayoutRoot.Children.Add(c1timeselect1);
```

Visual Basic
```
LayoutRoot.Children.Add(c1timeselect1)
```

7. Run your application. The **C1TimeSelector** control should resemble the following image:



Setting the Selected Time

You can specify the time of a **C1TimeSelector** control by setting the **SelectedTime** property in XAML or code.

In XAML

To specify the initially selected time, edit your XAML markup to resemble the following:

Markup
```
<Xaml:C1DateTimeSelector SelectedTime="02:05:05"/>
```

In Code

C#
```
c1timeselect1.SelectedTime = new TimeSpan( 02, 05, 05);
```

Visual Basic
```
c1timeselect1.SelectedTime = New TimeSpan(02, 05, 05)
```

C1DateTime Picker Help

Exchange date and time information using **DateTimePicker for UWP**. It provides a simple and intuitive UI for selecting date and time, just date values, or just time values. The date and time can be selected by using the buttons, the keyboard arrows, or by typing in fields.

C1DateTime Picker Quick Start

The following quick start guide is intended to get you up and running with **DateTimePicker for UWP**. In this quick start, you'll start in Visual Studio to create a new project, add a C1DateTimePicker control to your application, and customize the **C1DateTimePicker** control.

Step 1 of 3: Creating an Application with C1DateTimePicker Control

In this step, you'll begin in Visual Studio to create a UWP-style application using **DateTimePicker for UWP**.

Complete the following steps:

1. In Visual Studio, Select **File | New | Project**.
2. Select Templates | Visual C# | Windows | Universal. From the templates list, select **Blank App (Universal Windows)**. Enter a **Name** and click **OK** to create your project.
3. Open MainPage.xaml if it isn't already open, place the cursor between the <Grid> and </Grid> tags, and click once.
4. Navigate to the Toolbox and double-click the **C1DateTimePicker** icon to add the control to the grid. The XAML markup resembles the following:

Markup

```
<Grid x:Name="LayoutRoot">
<DateTimeEditors:C1DateTimePicker/>
</Grid>
```

You have successfully created a UWP-style application containing a C1DateTimePicker control. In the next step, you will modify the appearance of the control.

Step 2 of 3: Customizing the Control

In the previous step, you created a UWP-style application with a C1DateTimePicker control. In this step, you will modify the appearance of the control.

Complete the following steps:

1. Add `Height="60"` to the <DateTimeEditors:C1DateTimePicker/> tag to determine height of the control. The XAML markup appears as follows:

C#

```
<DateTimeEditors:C1DateTimePicker Height="60"/>
```

2. Add `Width="450"` to the <DateTimeEditors:C1DateTimePicker/> tag to determine the width of the control.

The XAML markup appears as follows:

Markup
```
<DateTimeEditors:C1DateTimePicker Height="60" Width="450"/>
```

3. Add `TimeFormat="ShortTime"` to the <DateTimeEditors:C1DateTimePicker/> tag to change the format of the time to a short format consisting of only hours and minute spaces. The XAML markup appears as follows:

Markup
```
<DateTimeEditors:C1DateTimePicker Height="60" Width="450"
TimeFormat="ShortTime"/>
```

4. Add `DateFormat="Long"` to the <DateTimeEditors:C1DateTimePicker/> tag to change the format of the date to a longer format that includes the weekday. The XAML markup appears as follows:

Markup
```
<DateTimeEditors:C1DateTimePicker Height="60" Width="450" TimeFormat="ShortTime"
DateFormat="Long"/>
```

5. Add FirstDayOfWeek="Wednesday" to the <DateTimeEditors:C1DateTimePicker/> tag; this will change the first day of the drop-down calendar's week to Wednesday. The XAML markup appears as follows:

Markup
```
<DateTimeEditors:C1DateTimePicker Height="60" Width="450" TimeFormat="ShortTime"
DateFormat="Long" FirstDayOfWeek="Wednesday"/>
```

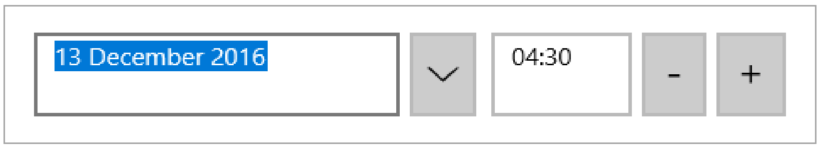
In this step, you customized the appearance of the **C1DateTimePicker** control. In the next step, you will run the project and experience the functionality of the control.

Step 3 of 3: Running the Project

In the last step, you customized the C1DateTimePicker control. In this step, you will run the project and observe some of the run-time features of the control.

Complete the following steps:

1. From the **Debug** menu, select **Start Debugging**. Your application will appear similar to the following:



2. Using your cursor, highlight an area in the date picker.

3. Click the time picker drop-down arrow to reveal the calendar and observe that the calendar's weeks start with Wednesday.

Congratulations!

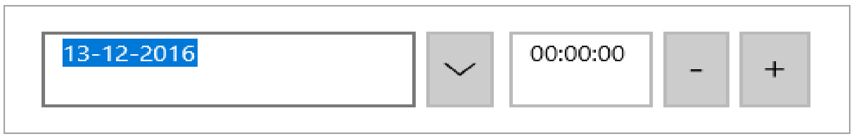
You have completed the **DateTimePicker for UWP** quick start. In this quick start, you created a UWP-style application containing a C1DateTimePicker control, modified the control's appearance, and experienced the control's run-time appearance features.

Working with C1DateTimePicker

The following topics will provide you with an overview of the C1DateTimePicker control’s elements and features.

C1DateTime Picker Elements

DateTimeEditors for UWP includes the C1DateTimePicker control, a simple control which provides, by default, both a date picker and a time picker. When you add the **C1DateTimePicker** control to a XAML window, it exists as a completely functional date and time picker. By default, the control's interface looks similar to the following image:



The **C1DateTimePicker** control consists of the following elements:

- **Date Picker**

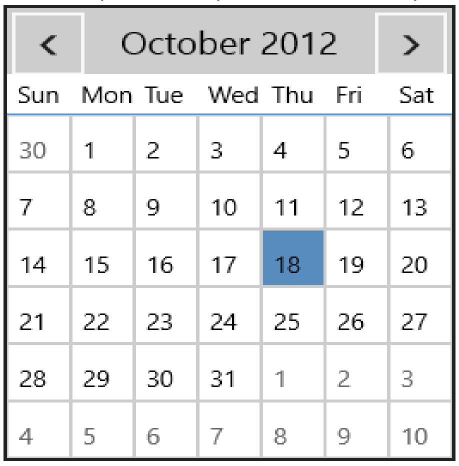
  The date picker element is comprised of a date field and the calendar drop-down button. You can set the date by entering numeric values or by selecting a date from the calendar.

- **Time Picker**

  The time picker element is comprised of the time field, the increase time button, and the decrease time button. You can set the time by entering numeric values or by clicking the buttons.

- **Date Picker Drop-Down Button**

  The date picker drop-down button opens a calendar from which you can select a date for the date picker.



- **Increase Time Button**

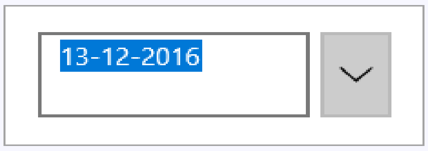
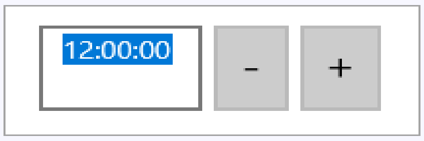
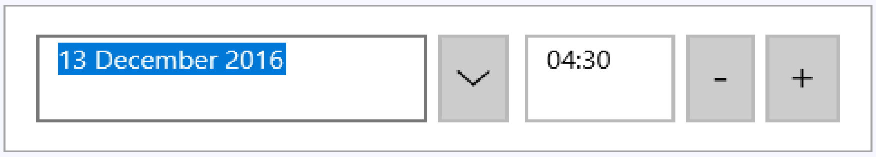
  The increase time button allows you to increase the time displayed in the time picker. Clicking the increase button will increase the time by one minute.

- **Decrease Time Button**

  The decrease time button allows you to decrease the time displayed in the time picker. Clicking the decrease button will decrease the time by one minute.

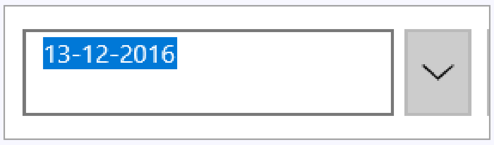
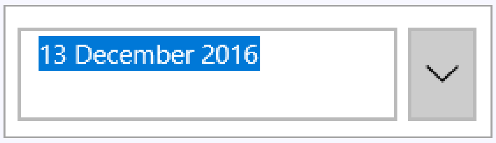
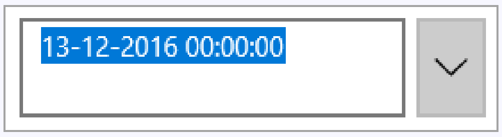
C1DateTimePicker Edit Modes

By default, the C1DateTimePicker control will appear on the page with both the date picker and the time picker. You can change the pickers that are displayed by setting the C1DateTimePicker.EditMode property to **Date, Time**, or **DateTime**. You can set the **C1DateTimePicker.EditMode** property to **Date** to display only the date picker; you can set the **C1DateTimePicker.EditMode** property to **Time** to only display the time picker; and you can set the **C1DateTimePicker.EditMode** property to **DateTime** to display both the time picker and date picker. The table below illustrates each editor mode.

Editor Mode	Result
Date	 13-12-2016
Time	 12:00:00
DateTime	 13 December 2016 04:30

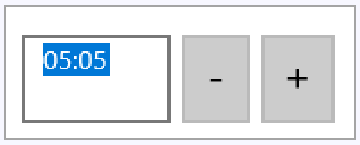
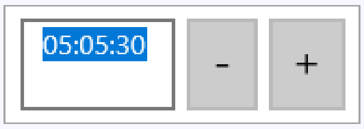
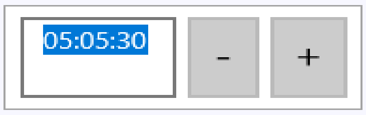
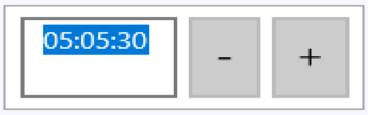
C1DateTimePicker Date Format

You can use the C1DateTimePicker.DateFormat property to set the format that the date picker displays. You can set **C1DateTimePicker.DateFormat** property to **Short**, **Long**, or **Custom**. The table below illustrates the two date formats.

Date Format	Result
Short (Default)	 13-12-2016
Long	 13 December 2016
Custom	 13-12-2016 00:00:00

C1DateTimePicker Time Format

You can use the C1DateTimePicker.TimeFormat property to set the format that the date picker displays. You can set C1DateTimePicker.TimeFormat property to ShortTime, LongTime, TimeSpan, or Custom. The table below illustrates the two date formats.

Time Format	Result
ShortTime	 05:05 - +
LongTime (default)	 05:05:30 - +
TimeSpan (default)	 05:05:30 - +
Custom (default)	 05:05:30 - +

C1DateTimePicker Task-Based Help

The task-based help assumes that you are familiar with programming in Visual Studio and know how to use the C1DateTimePicker control in general. If you are unfamiliar with the **C1DateTimePicker** control, please see the C1DateTimePicker Quick Start first.

Each topic in this section provides a solution for specific tasks using the **C1DateTimePicker** product.

Each task-based help topic also assumes that you have created a new project.

Allowing Null Values

By default, the C1DateTimePicker control doesn't allow users to enter null values, but you can force the control to accept a null value by setting the C1DateTimePicker.AllowNull property to **True**. In this topic, you will learn how to set the **C1DateTimePicker.AllowNull** property to **True** in the designer, in XAML, and in code.

In the Designer

Complete the following steps:

1. Click the **C1DateTimePicker** control once to select it.
2. In the **Properties** window, select the C1DateTimePicker.AllowNull check box.

In XAML

To allow null values, place AllowNull="True" to the <DateTimeEditors:C1DateTimePicker>tag so that the markup resembles the following:

Markup

```
<DateTimeEditors:C1DateTimePicker AllowNull="True"/>
```

In Code

Complete the following steps:

1. Open the **MainPage.xaml.cs** page.
2. Place the following code beneath the **C1DateTimePicker_Loaded** event:

   Visual Basic
   ```
   C1DateTimePicker1.AllowNull = True
   ```

   C#
   ```
   c1DateTimePicker1.AllowNull = true;
   ```

3. Run the project.

Selecting the Edit Mode

By default, the C1DateTimePicker control shows both the date and time pickers, but you may also choose to show only the date picker or only the time picker. In this topic, you will learn how to change the editor mode in the designer, in XAML, and in code.

In the Designer

To change the edit mode, complete the following steps:

1. Click the **C1DateTimePicker** control once to select it.
2. In the **Properties** window, click the C1DateTimePicker.EditMode drop-down arrow and select a mode from the list. For this example, select **Date**.

In XAML

To change the edit mode, place EditMode="Date" to the <DateTimeEditors:C1DateTimePicker>tag so that the markup resembles the following:

Markup
```
<DateTimeEditors:C1DateTimePicker EditMode="Date">
```

In Code

To change the edit mode, complete the following steps:

1. Open the **MainPage.xaml.cs** page.
2. Import the following namespace:

   Visual Basic
   ```
   Imports C1.XAML.DateTimeEditors
   ```

   C#

```
using C1.XAML.DateTimeEditors;
```

3. Place the following code beneath the **C1DateTimePicker_Loaded** event:

Visual Basic

```
C1DateTimePicker1.EditMode = C1DateTimePickerEditMode.Date
```

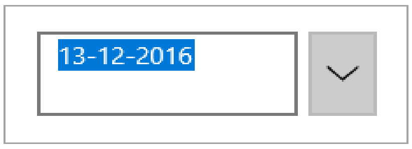
C#

```
c1DateTimePicker1.EditMode = C1DateTimePickerEditMode.Date;
```

4. Run the project.

This Topic Illustrates the Following:

In this topic, you set the **C1DateTimePicker.EditMode** to **Date**, which removes the time picker form the **C1DateTimePicker** control. The result of this topic will resemble the following image:


13-12-2016

Selecting the Time Format

By default, the C1DateTimePicker control's time picker displays the time in a long format that includes seconds, but it can also display time in a shorter format. In this topic, you will learn how to change the time format in the designer, in XAML, and in code.

In the Designer

To change the time format, complete the following steps:

1. Click the **C1DateTimePicker** control once to select it.
2. In the **Properties** window, click the C1DateTimePicker.TimeFormat drop-down arrow and select a mode from the list. For this example, select **ShortTime**.

In XAML

To change the time format, place TimeFormat="ShortTime" to the <DateTimeEditors:C1DateTimePicker>tag so that the markup resembles the following:

Markup

```
<DateTimeEditors:C1DateTimePicker TimeFormat="ShortTime">
```

In Code

To change the time format, complete the following steps:

1. Open the **MainPage.xaml.cs** page.
2. Import the following namespace:

Visual Basic

```
Imports C1.XAML.DateTimeEditors
```

C#

```
using C1.WAML.DateTimeEditors;
```

3. Place the following code beneath the **C1DateTimePicker_Loaded** event:

Visual Basic

```
C1DateTimePicker1.TimeFormat = C1TimeEditorFormat.ShortTime
```

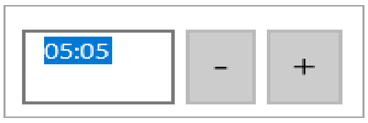
C#

```
c1DateTimePicker1.TimeFormat = C1TimeEditorFormat.ShortTime;
```

4. Run the project.

This Topic Illustrates the Following:

In this topic, you set the C1DateTimePicker.TimeFormat to ShortTime, which provides a shortened time display. The final result will resemble the following image:



Selecting the Date Format

By default, the C1DateTimePicker control's date picker displays the date in a short format, but it can also display time in a long format. In this topic, you will learn how to change the date format in the designer, in XAML, and in code.

In the Designer

To change the date format, complete the following steps:

1. Click the **C1DateTimePicker** control once to select it.
2. In the **Properties** window, click the **C1DateTimePicker.DateFormat** drop-down arrow and select a mode from the list. For this example, select **Long**.

In XAML

To change the date format, place DateFormat="Long" to the <DateTimeEditors:C1DateTimePicker>tag so that the markup resembles the following:

Markup

```
<DateTimeEditors:C1DateTimePicker DateFormat="Long">
```

In Code

To change the date format, complete the following steps:

1. Open the **MainPage.xaml.cs** page.
2. Place the following code beneath the **C1DateTimePicker_Loaded** event:

Visual Basic

```
C1DateTimePicker1.DateFormat = Microsoft.Windows.Controls.DatePickerFormat.Long
```

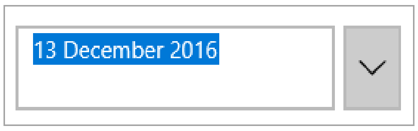
C#

```
c1DateTimePicker1.DateFormat = Microsoft.Windows.Controls.DatePickerFormat.Long;
```

3. Run the project.

This Topic Illustrates the Following:

In this topic, you set the C1DateTimePicker.DateFormat to Long, which will display the date in a long format. The final result will resemble the following image:



Specifying the Date and Time

You can specify the time and date of a C1DateTimePicker control by setting the C1DateTimePicker.DateTime property using code.

> Try to avoid setting the **C1DateTimePicker.DateTime** property in XAML. Parsing these values from strings is culture specific. If you set a value with your current culture and a user is using different culture, the user can get XamlParseException when loading your site. The best practice is to set these values from code or via data binding.

In Code

Complete the following steps:

1. Open the **MainPage.xaml.cs** page.
2. Place the following code beneath the **C1DateTimePicker_Loaded** event:

Visual Basic

```
C1DateTimePicker1.DateTime = New DateTime(2010, 1, 17, 11, 04, 0)
```

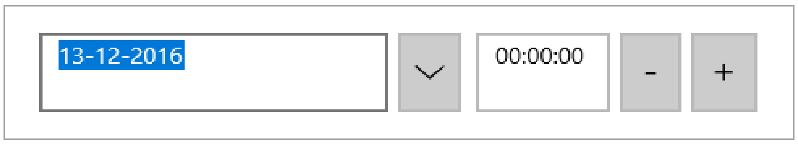
C#

```
c1DateTimePicker1.DateTime = new DateTime(2010, 1, 17, 11, 04, 0);
```

3. Run the project and observe that the **C1DateTimePicker** control shows the date and time as 01/17/2010 11:04:00 AM.

This Topic Illustrates the Following:

The result of this topic resembles the following image:



C1DatePicker Help

Display and choose date information using **DatePicker for UWP**. The C1DatePicker control provides a simple and intuitive UI for selecting date values. Click the **C1DatePicker's** drop-down arrow and select a date in the calendar.

C1DatePicker Quick Start

The following quick start guide is intended to get you up and running with the C1DatePicker control. In this quick start, you'll start in Visual Studio to create a new project, add a **C1DatePicker** control to your application, and customize the **C1DatePicker** control.

Step 1 of 3: Creating an Application with C1DatePicker Control

In this step, you'll create a UWP-style application and add a C1DatePicker control to the window.

Complete the following steps:

1. In Visual Studio, Select **File | New | Project**.
2. Select Templates | Visual C# | Windows | Universal. From the templates list, select **Blank App (Universal Windows)**. Enter a **Name** and click **OK** to create your project.
3. Open MainPage.xaml if it isn't already open, place the cursor between the <Grid> and </Grid> tags, and click once.
4. Navigate to the Toolbox and double-click the **C1DatePicker** icon to add the control to the grid. The XAML markup resembles the following:

Markup

```
<Grid x:Name="LayoutRoot">
<DateTimeEditors:C1DatePicker></DateTimeEditors:C1DateTimePicker>
</Grid>
```

You have completed the first step of the **C1DatePicker for UWP** quick start. In this step, you created a project and added a **C1DatePicker** control to it. In the next step, you'll customize the control.

Step 2 of 3: Customizing the C1DatePicker

In this step, you will customize the C1DatePicker control.

Select the **C1DatePicker** control and then, in Visual Studio **Properties** window, set the following properties:

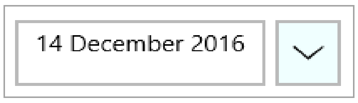
- Set the C1DatePicker.SelectedDateFormat property to Long. This will change the date format of the control to include the full week day name and month name.
- Click the drop-down arrow next to the C1DatePicker.ButtonBackground property and select a color from the drop-down color picker.

Now that you've customized the application, you can run the project and observe the run time behaviors of the control.

Step 3 of 3: Running the Application

In the previous two steps, you created a UWP-style application with a **C1DatePicker** control and customized the control. In the last step of this quick start, you will run the project and interact with the control.

Press **F5** to run the project. Notice the drop-down arrow is the color you specified in the C1DatePicker.ButtonBackground property. The C1DatePicker.SelectedDate should appear in the display box and it should show the full week day and month names as in the following image:



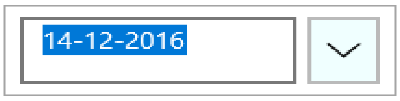
Congratulations – you have completed the quick start!

Working with C1DatePicker

The following topics will provide you with an overview of the C1DatePicker control's elements and features.

C1DatePicker Elements

DateTimeEditors for UWP includes the C1DatePicker control, a simple control which provides a date picker that, when clicked at run time, allows you to choose a date from a drop-down calendar. When you add the **C1DatePicker** control to a XAML window, it exists as a completely functional date picker. By default, the control's interface looks similar to the following image:



The **C1DatePicker** control consists of the following elements:

- **Display Box**

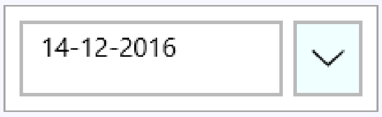
  The display box presents the selected date. This can be set using the C1DatePicker.SelectedDate property. Users can also input numeric date into the display box at run time. When you enter a numeric value, it will automatically be converted to a date. The control can use the **C1DatePicker.SelectedDate** property display dates in three edit modes: **Long**, **Short** (default), and **Custom**.

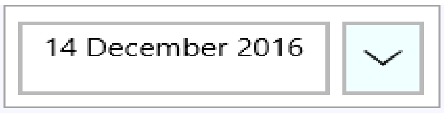
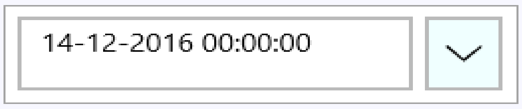
- **Drop-Down Arrow**

  Clicking the drop-down arrow of the **C1DatePicker** control at run time allows you to select a date from a drop-down calendar that appears.

Date Formats

You can use the C1DatePicker.SelectedDateFormat property to set the format that the date picker displays. You can set **C1DatePicker.SelectedDateFormat** property to Short, Long, or Custom. The table below illustrates each date format.

Date Format	Result	Description
Short (default)		The control displays a short date format with numeric values.

Long	 14 December 2016	The control displays a long date format.
Custom	 14-12-2016 00:00:00	The control displays the custom format defined in the C1DatePicker.CustomFormat property. **NOTE:** Use full formats, not abbreviated formats.

C1DatePicker Task-Based Help

The task-based help assumes that you are familiar with programming in Visual Studio and know how to use the C1DatePicker control in general. If you are unfamiliar with the **C1DatePicker** control, please see the C1DatePicker Quick Start first.

Each topic in this section provides a solution for specific tasks using the **C1DatePicker** control.

Each task-based help topic also assumes that you have created a new project.

Allowing Null Values

By default, the C1DatePicker control doesn't allow users to enter null values, but you can force the control to accept a null value by setting the C1DatePicker.AllowNull property to **True**. In this topic, you will learn how to set the C1DatePicker.AllowNull property to **True** in the designer, in XAML, and in code.

In the Designer

Complete the following steps:

1. Click the C1DatePicker control once to select it.
2. In the Visual Studio **Properties** window, select the C1DatePicker.AllowNull check box.

In XAML

To allow null values, place AllowNull="True" within the <DateTimeEditors:C1DatePicker>tags so that the markup resembles the following:

Markup
```
<DateTimeEditors:C1DatePicker AllowNull="True"/>
```

In Code

Complete the following steps:

1. Open the **MainWindow.xaml.cs** page.
2. Place the following code beneath the **C1DatePicker_Loaded** event:

Visual Basic
```
C1DatePicker1.AllowNull = True
```

C#

```
c1DatePicker1.AllowNull = true;
```

3. Run the project.

Selecting the Date Format

By default, the C1DatePicker control displays the date in a short format, but it can also display the date in a long or custom format. In this topic, you will learn how to change the date format in the designer, in XAML, and in code.

In the Designer

To change the date format, complete the following steps:

1. Click the **C1DatePicker** control once to select it.
2. In the **Properties** window, click the **C1DatePicker.SelectedDateFormat** drop-down arrow and select an option from the list. For this example, select **Long**.

In XAML

To change the date format, place SelectedDateFormat="Long" within the <DateTimeEditors:C1DatePicker>tags so that the markup resembles the following:

Markup
```
<DateTimeEditors:C1DatePicker SelectedDateFormat="Long">
```

In Code

To change the date format, complete the following steps:

1. Open the **MainWindow.xaml.cs** page.
2. Place the following code beneath the **C1DatePicker_Loaded** event:

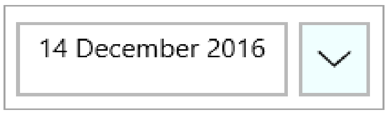
Visual Basic
```
C1DatePicker1.SelectedDateFormat =
C1.Xaml.DateTimeEditors.C1DatePickerFormat.Long
```

C#
```
c1DateTimePicker1.SelectedDateFormat =
C1.Xaml.DateTimeEditors.C1DatePickerFormat.Long;
```

3. Run the project.

This Topic Illustrates the Following:

In this topic, you set the C1DatePicker.SelectedDateFormat to **Long**, which will display the date in a long format. The final result will resemble the following image:



Setting the First Day of the Week

By default, the C1DatePicker control uses Sunday as the first day of the week in the drop-down calendar, but you can change the starting day if necessary. In this topic, you will learn how to change the first day of the week in the designer, in XAML, and in code.

In the Designer

To change the first day of the week, complete the following steps:

1. Click the **C1DatePicker** control once to select it.
2. In the **Properties** window, click the **C1DatePicker.FirstDayOfWeek** drop-down arrow and select a day from the list. For this example, select **Monday**.

In XAML

To change the first day of the week, place FirstDayOfWeek="Monday" within the <DateTimeEditors:C1DatePicker>tags so that the markup resembles the following:

Markup

```
<DateTimeEditors:C1DatePicker FirstDayOfWeek="Monday">
```

In Code

To change the date format, complete the following steps:

1. Open the **MainWindow.xaml.cs** page.
2. Place the following code beneath the **C1DatePicker_Loaded** event:

   Visual Basic

   ```
   C1DatePicker1.FirstDayOfWeek= DayOfWeek.Monday
   ```

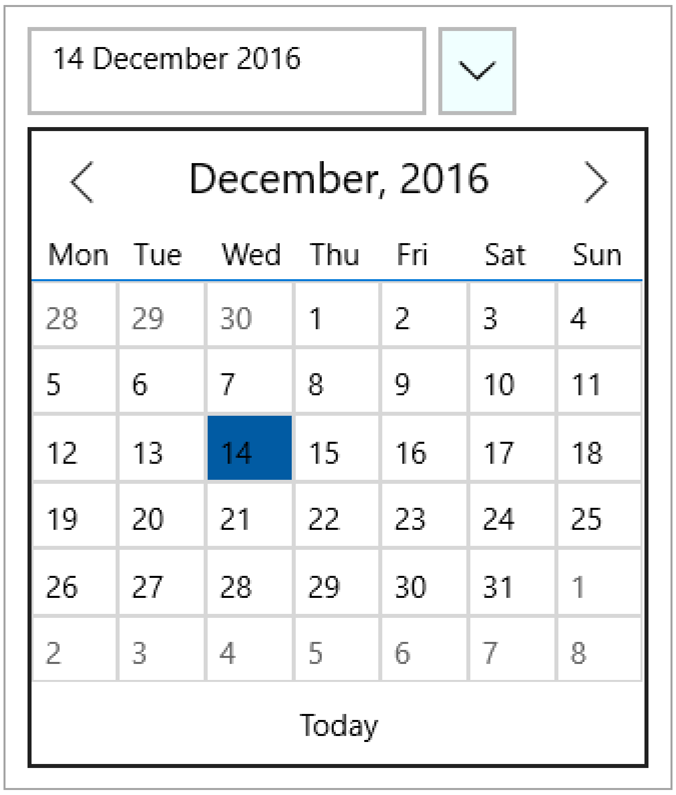
   C#

   ```
   c1DatePicker1.FirstDayOfWeek= DayOfWeek.Monday;
   ```

3. Run the project.

This Topic Illustrates the Following:

In this topic, you set the C1DatePicker.FirstDayOfWeek property to **Monday** so that Monday is the first day of the week in the drop-down calendar. The final result will resemble the following image:



Setting the Calendar Start and End Date

You can change the dates that appear in the drop-down calendar by setting the C1DatePicker.DisplayDateStart and C1DatePicker.DisplayDateEnd properties. In this topic, you will learn how to change the start and end dates in code.

In Code

To change the dates that appear in the calendar, complete the following steps:

1. Open the MainWindow.xaml.cs page.
2. Place the following code beneath the **C1DatePicker_Loaded** event:

Visual Basic

```
Dim dateStringStart As String = "5-2-2016"
Dim dateStringEnd As String = "25-2-2016"
c1DatePicker1.DisplayDateStart = DateTime.Parse(dateStringStart)
c1DatePicker1.DisplayDateEnd = DateTime.Parse(dateStringEnd)
```

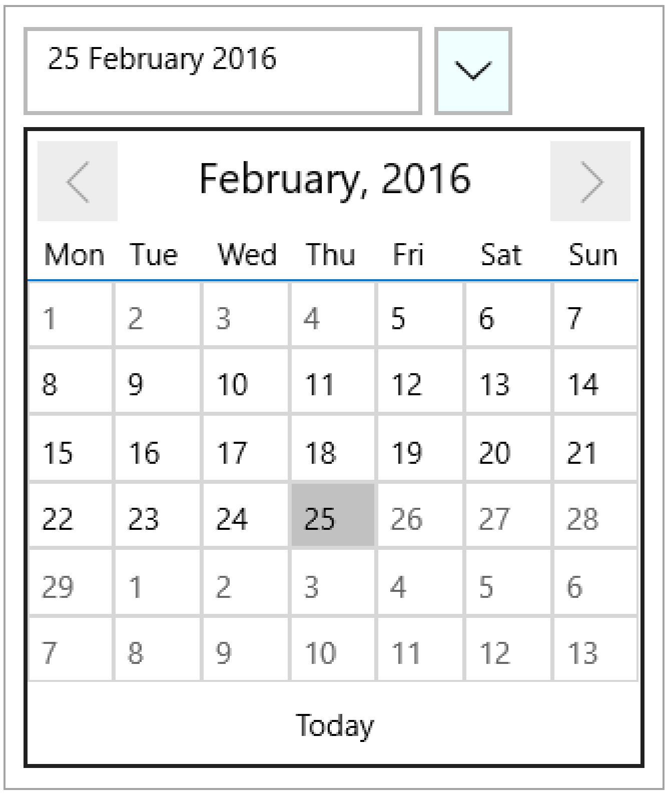
C#

```
string dateStringStart = "5-2-2016";
string dateStringEnd = "25-2-2016";
c1DatePicker1.DisplayDateStart = DateTime.Parse(dateStringStart);
c1DatePicker1.DisplayDateEnd = DateTime.Parse(dateStringEnd);
```

3. Run the project.

This Topic Illustrates the Following:

In this topic, you set the **C1DatePicker.DisplayDateStart** and **C1DatePicker.DisplayDateEnd** properties to determine the dates that appear in the drop-down calendar. The final result will resemble the following image:



C1TimeEditor Help

Exchange time information with an end-user using **TimePicker for UWP**. It provides a simple interface for selecting time values. The time can be selected by using the increment and decrement buttons, the keyboard arrows, or by typing in fields.

C1TimeEditor Quick Start

The following quick start guide is intended to get you up and running with **TimeEditor for UWP**. In this quick start, you'll start in Visual Studio to create a new project, add a C1TimeEditor control to your application, and customize the **C1TimeEditor** control.

Step 1 of 3: Creating an Application with a C1TimeEditor Control

In this step, you'll begin in Visual Studio to create a UWP-style application using **TimeEditor for UWP**.

Complete the following steps:

1. In Visual Studio, Select **File | New | Project**.
2. Select Templates | Visual C# | Windows | Universal. From the templates list, select **Blank App (Universal Windows)**. Enter a **Name** and click **OK** to create your project.
3. Open MainPage.xaml if it isn't already open, place the cursor between the <Grid> and </Grid> tags, and click once.
4. Navigate to the Toolbox and double-click the **C1TimeEditor** icon to add the control to the grid. The XAML markup resembles the following:

Markup

```
<Grid x:Name="LayoutRoot">
<DateTimeEditors:C1TimeEditor></DateTimeEditors:C1TimeEditor>
</Grid>
```

You have completed the first step of the **TimeEditor for UWP** quick start. In this step, you created a project and added a C1TimeEditor control to it. In the next step, you'll customize the control.

Step 2 of 3: Customizing the Control

In this step, you will customize the C1TimeEditor control:

1. Select the C1TimeEditor control and then, in **Properties** panel, set the following properties:
   - Set the C1TimeEditor.Format property to **ShortTime**. This will change the time format of the control so that it shows only hours and minutes.
   - Set the C1TimeEditor.Increment property to "01:00:00". This will cause the value of the control to change by one hour each time a user clicks the spin button.
   - Set the C1TimeEditor.Interval property to "1000". This will cause the control to hesitate for one second before changing the value of the control.
2. Set the current time to **5:00 p.m.** by completing the following steps:

1. In the XAML editor, add `x:Name="C1TimeEditor1"` to the `<DateTimeEditors:C1TimeEditor>` tag so that the control will have a unique identifier for you to call in code.
2. Open the **MainPage.xaml.cs** page.
3. Place the following code beneath the **C1TimeEditor_Loaded** event:

Visual Basic

```
C1TimeEditor1.Value = New TimeSpan(17, 0, 0)
```

C#

```
C1TimeEditor1.Value = new TimeSpan(17,0,0);
```

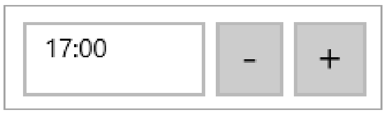
Now that you've customized the application, you can run the project and observe the run time behaviors of the control.

Step 3 of 3: Running the Application

In the previous two steps, you created a UWP-style application with a C1TimeEditor control and customized the control. In the last step of this quick start, you will run the project and interact with the control.

Complete the following steps:

1. On the toolbar, select **Project | Run Project** to run the project. Observe that it loads with a time value of 17:00 and that the control only shows hours and minutes:



2. Click the decrease time button and observe that time value decreases by one hour to 16:00.

3. Click and hold the increase time button so that the control will spin through values. Observe that the control waits one second between value changes.

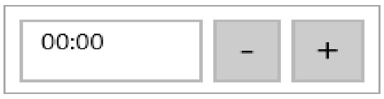
Congratulations – you have completed the **TimeEditor for UWP** quick start!

Working with C1TimeEditor

The following topics will provide you with an overview of the C1TimeEditor control's elements and features.

C1TimeEditor Elements

DateTimeEditors for UWP includes the C1TimeEditor control, a simple control which provides a time picker that can show short time, long time, and time spans. When you add the **C1TimeEditor** control to a XAML window, it exists as a completely functional time picker. By default, the control's interface looks similar to the following image:



The **C1TimeEditor** control consists of the following elements:

- **Display Box**

The display box presents the selected time. This can be set using the C1TimeEditor.Value property. Users can also input numeric date into the display box. When you enter a numeric value, it will automatically be converted into time. The control can display time in three edit modes: **LongTime** (default), **ShortTime**, and **TimeSpan**.

- **Increase Time Button**

  The increase time button allows you to increase the time displayed in the time picker. Clicking the increase button will increase the time by one minute unless you have specified another interval.

- **Decrease Time Button**

  The decrease time button allows you to decrease the time displayed in the time picker. Clicking the decrease button will decrease the time by one minute unless you have specified another interval.

Spin Interval

There are two ways that users can increase or decrease values using the spin button: they can either repeatedly click one of the buttons to increase or decrease the time at their own pace, or they can hold down the decrease time button or increase time button while time increases or decreases at the speed of program-specified intervals. You can specify the interval by setting the C1TimeEditor.Interval property.

By default, the **C1TimeEditor.Interval** property is set to 33 milliseconds, which allows users to scroll through time values at faster rates. You can slow that scrolling time down by specifying a higher number, such as 500 milliseconds (one-half of a second), or speed it up by specifying a lower number, such as 10 milliseconds (one-hundredth of a second).

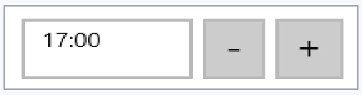
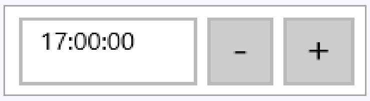
Note: You cannot set the **C1TimeEditor.Interval** to "0".

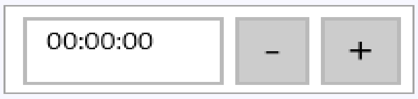
Value Increment

Each time a user clicks the increase time or decrease time spin buttons, the value of the control increases or decreases by a program-specified increment. By default, this increment is 00:01:00, or one minute. You can increase or decrease this increment by setting the C1TimeEditor.Increment property. The **C1TimeEditor.Increment** property will take any value between 00:00:00 (which will disable the spin buttons) and 23:59:59.

Time Formats

You can use the C1TimeEditor.Format property to set the format that the date picker displays. You can set **C1TimeEditor.Format** property to **ShortTime**, **LongTime**, or **TimeSpan**. The table below illustrates each date formats.

Time Format	Result	Description
ShortTime	 17:00 - +	The control displays a short time format that excludes seconds.
LongTime (default)	 17:00:00 - +	The control displays a long time format that includes seconds.

TimeSpan		The control displays a time span.
Custom		The control displays custom time format.

C1TimeEditor Task-Based Help

The task-based help assumes that you are familiar with programming in Visual Studio and know how to use the C1TimeEditor control in general. If you are unfamiliar with the **C1TimeEditor** control, please see the C1TimeEditor Quick Start first.

Each topic in this section provides a solution for specific tasks using the **C1TimeEditor** control.

Each task-based help topic also assumes that you have created a new project.

Allowing Null Values

By default, the C1TimeEditor control doesn't allow users to enter null values, but you can force the control to accept a null value by setting the C1TimeEditor.AllowNull property to **True**. In this topic, you will learn how to set the C1TimeEditor.AllowNull property to **True** in the designer, in XAML, and in code.

In the Designer

Complete the following steps:

1. Click the C1TimeEditor control once to select it.
2. In the **Properties** window, select the C1TimeEditor.AllowNull check box.

In XAML

To allow null values, place AllowNull="True" to the <DateTimeEditors:C1TimeEditor>tag so that the markup resembles the following:

Markup
```
<DateTimeEditors:C1TimeEditor AllowNull="True"/>
```

In Code

Complete the following steps:

1. Open the **MainPage.xaml.cs** page.
2. Place the following code beneath the **C1TimeEditor_Loaded** event:

   Visual Basic
   ```
   C1TimeEditor1.AllowNull = True
   ```

   C#
   ```
   C1TimeEditor1.AllowNull = true;
   ```

3. Run the project.

Removing the Spin Buttons

You can remove the **C1TimeEditor** control's spin buttons by setting the C1TimeEditor.ShowButtons property to **False**. In this topic, you will learn how to set the C1TimeEditor.ShowButtons property to **False** in the designer, in XAML, and in code.

In the Designer

Complete the following steps:

1. Click the C1TimeEditor control once to select it.
2. In the **Properties** window, clear the C1TimeEditor.ShowButtons check box.

In XAML

To remove the spin buttons, place ShowButtons="False" to the <DateTimeEditors:C1TimeEditor>tag so that the markup resembles the following:

Markup
```
<DateTimeEditors:C1TimeEditor ShowButtons="False"/>
```

In Code

Complete the following steps:

1. Open the **MainPage.xaml.cs** page.

2. Place the following code beneath the **C1TimeEditor_Loaded** event:

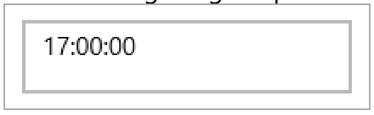
   Visual Basic
   ```
   C1TimeEditor1.ShowButtons = False
   ```

   C#
   ```
   C1TimeEditor1.ShowButtons = false;
   ```

3. Run the project.

This Topic Illustrates the Following:

The following image depicts a C1TimeEditor control with its spin buttons removed:



Selecting the Time Format

By default, the **C1TimeEditor** control displays the time in a long format that includes seconds, but it can also display time in a shorter format or into a time span format. In this topic, you will learn how to change the time format in the designer, in XAML, and in code.

In the Designer

To change the time format, complete the following steps:

1. Click the C1TimeEditor control once to select it.
2. In the **Properties** window, click the Format drop-down arrow and select a mode from the list. For this example, select **ShortTime**.

In XAML

To change the time format, place Format="ShortTime" to the <DateTimeEditors:C1TimeEditor>tag so that the markup resembles the following:

Markup
```
<DateTimeEditors:C1TimeEditor Format="ShortTime">
```

In Code

To change the time format, complete the following steps:

1. Open the **MainPage.xaml.cs** page.
2. Import the following namespace:

   Visual Basic
   ```
   Imports C1.WPF.DateTimeEditors
   ```

   C#
   ```
   using C1.WPF.DateTimeEditors;
   ```

3. Place the following code beneath the **C1TimeEditor_Loaded** event:

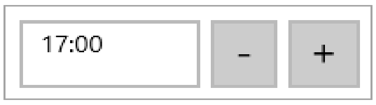
   Visual Basic
   ```
   C1TimeEditor1.Format = C1TimeEditorFormat.ShortTime
   ```

   C#
   ```
   C1TimeEditor1.Format = C1TimeEditorFormat.ShortTime;
   ```

4. Run the project.

This Topic Illustrates the Following:

In this topic, you set the Format to **ShortTime**, which provides a shortened time display. The final result will resemble the following image:



Setting the Spin Interval

By default, the C1TimeEditor.Interval property is set to 33 milliseconds, which allows users to scroll through the time values at faster rates. In this topic, you will specify a longer interval between value changes by setting the **C1TimeEditor.Interval** property to 1000 milliseconds. For more information on spin intervals, visit the Spin Interval topic.

In the Designer

Complete the following steps:

1. Click the **C1TimeEditor** control once to select it.
2. In the **Properties** window, locate the **C1TimeEditor.Interval** property and enter "1000" into its text box.
3. Run the project and then click and hold the increase time button +. Observe that the value only increases once a second.

In XAML

Complete the following steps:

1. Add Interval="1000" to the <my:C1TimeEditor>tag so that the markup resembles the following:

   Markup
   ```
   <my:C1TimeEditor Interval="1000"/>
   ```

2. Run the project and then click and hold the increase time button +. Observe that the value only increases once a second.

In Code

Complete the following steps:

1. Open the **MainPage.xaml.cs** page.

2. Place the following code beneath the **InitializeComponent()** method:

   Visual Basic
   ```
   C1TimeEditor1.Interval = 1000
   ```

   C#
   ```
   C1TimeEditor1.Interval = 1000;
   ```

3. Run the project and then click and hold the increase time button +. Observe that the value only increases once a second.

Setting the Value Increment

By default, the time on a **C1TimeEditor** control is set to move in one minute increments. You can change this by setting the C1TimeEditor.Increment property to whatever time increment you specify. In this topic, you will set the time increment on the **C1TimeEditor** control to one hour and thirty minutes. For more information about time increments, visit the **Value Increment** topic.

In the Designer

Complete the following steps:

1. Click the **C1TimeEditor** control once to select it.
2. In the **Properties** window, locate the **C1TimeEditor.Increment** property and enter "01:30:00" into its text box.
3. Run the project and click the increase time button. Observe that time jumps ahead by one hour and thirty minutes.

In XAML

Complete the following steps:

1. Add Increment="01:30:00" to the <DateTimeEditors:C1TimeEditor>tag so that the markup resembles the following:

   Markup
   ```
   <DateTimeEditors:C1TimeEditor Increment="01:30:00"/>
   ```

2. Run the project and click the increase time button. Observe that time jumps ahead by one hour and thirty minutes.

In Code

Complete the following steps:

1. Open the **MainPage.xaml.cs** page.
2. Place the following code beneath the **InitializeComponent()** method:

   Visual Basic
   ```
   C1TimeEditor1.Increment= New TimeSpan(01,30,00)
   ```

   C#
   ```
   C1TimeEditor1.Increment= new TimeSpan(01,30,00);
   ```

3. Run the project and click the increase time button . Observe that time jumps ahead by one hour and thirty minutes.

Specifying the Current Time

You can specify the current time of a **C1TimeEditor** control by setting the C1TimeEditor.Value property in code.

> Try to avoid setting the **C1TimeEditor.Value** property in XAML as a string value. Parsing these values from strings is culture specific. If you set a value with your current culture and a user is using different culture, the user can get XamlParseException when loading your site. The best practice is to set these values from code or via data binding.

In the Designer

Complete the following steps:

1. Click the **C1TimeEditor** control once to select it.
2. In the **Properties** window, set the **C1TimeEditor.Value** property to " 07:00:00". The control reflects a time of 7:00:00 a.m.

In XAML

To specify the current time, place Value="07:00:00" to the <DateTimeEditors:C1TimeEditor>tag so that the markup resembles the following:

Markup
```
<DateTimeEditors:C1TimeEditor Value="07:00:00"/>
```

The control reflects a time of 7:00:00 a.m.

In Code

Complete the following steps:

1. Open the **MainPage.xaml.cs** page.
2. Place the following code beneath the **C1TimeEditor_Loaded** event:

Visual Basic

```
C1TimeEditor1.Value = New TimeSpan(7, 0, 0)
```

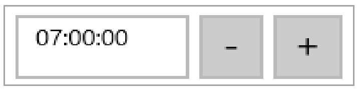
C#

```
C1TimeEditor1.Value = new TimeSpan(7,0,0);
```

3. Run the project and observe that control reflects a time of 7:00:00 a.m.

This Topic Illustrates the Following:

By following the steps in this topic, you have changed the time on the **C1TimeEditor** control to 7:00:00 a.m. The result will resemble the following:



Working with Time Spans

You can modify a C1TimeEditor control so that it will display a time span. In this tutorial, you will create a **C1TimeEditor** control that represents a time span between 5:00 and 10:00. You will also write code for the project that sets the starting value to 7:00 a.m.

Complete the following steps:

1. Click the **C1TimeEditor** control once to select it.
2. In the **Properties** window, complete the following steps:
    1. Set the Format property to **TimeSpan**.
    2. Set the C1TimeEditor.Maximum property to a value, such as "10:00:00".
    3. Set the C1TimeEditor.Minimum property to a value, such as "05:00:00".
    4. Set the C1TimeEditor.Value property to a value, such as "07:00:00".
3. Run the project and observe that the control loads with a time of 07:00:00 a.m.
4. Click the increase time button until you can go no further. It will stop at 10:00:00.
5. Click the decrease time button until you can go no further. It will stop at 05:00:00.