Carousel for WPF

By GrapeCity, Inc.
# Table of Contents

ComponentOne Carousel for WPF Overview ........................................................................... 5
  Help with ComponentOne Studio for WPF ........................................................................ 5

Carousel for WPF Features ..................................................................................................... 7

Carousel for WPF Quick Start ................................................................................................. 7
  Step 1 of 3: Creating the WPF Application ........................................................................ 8
  Step 2 of 3: Adding Content to the Application ................................................................ 8
  Step 3 of 3: Running the Application ................................................................................ 10

Working with Carousel for WPF .......................................................................................... 11
  Using C1CarouselPanel ...................................................................................................... 11
  Customizing the Carousel .................................................................................................. 12

Run-Time Interaction ............................................................................................................ 13

Carousel for WPF Task Based Help ...................................................................................... 15
  Adding C1Carousel to the Application .............................................................................. 15
  Adding and Moving Content in C1Carousel ..................................................................... 16
    Moving Content Along an Elliptical Path ....................................................................... 16
    Moving Content Along a ZigZag Path ............................................................................ 18
    Moving Content Along a Diagonal Path ......................................................................... 21
    Moving Content Along a Parabolic Path ....................................................................... 24
    Moving Content Along a Spiral Path ............................................................................ 26
  Using the ScrollViewer ........................................................................................................ 29
  Limiting the Number of Visible Elements ....................................................................... 30
ComponentOne Carousel for WPF

Overview

C1CarouselPanel is a virtualizing panel control that arranges child elements along an arbitrary 3D path. You can use C1CarouselPanel with any items control to turn a flat list of items into a virtualizing carousel with true 3D perspectives and scrolling animation.

For a list of the latest features added to ComponentOne Studio for WPF, visit What's New in Studio for WPF.

Help with ComponentOne Studio for WPF

Getting Started

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

What's New

For a list of the latest features added to ComponentOne Studio for WPF, visit What's New in Studio for WPF.
Carousel for WPF Features

ComponentOne Carousel for WPF provides the C1CarouselPanel control that allows you to view content and controls – including standard or 3rd-party controls – in an interactive carousel view. Similar in concept to the C1CoverFlow or C1HyperPanel control, C1CarouselPanel is a unique way of displaying content and data. Items in the panel revolve on a path when manipulated at run time.

ComponentOne Carousel for WPF allows you to create customized, rich applications. Make the most of Carousel for WPF by taking advantage of the following key features:

- **View Data Interactively**
  The C1CarouselPanel control allows you to display content and data in a unique, highly interactive format.

- **Define an Arbitrary Path**
  C1CarouselPanel items can navigate along any arbitrary path. Define a simple ellipse path, or create a more advanced path such as a zig-zag or a figure '8'. C1CarouselPanel provides a simplified 3D path definition as a composition of a 2D path and a rotation angle in a 3D space.

- **True 3D Perspective**
  When distributing elements along a 3D path, the placements of child elements in 3D space are fully honored. This means that more distant elements are overlapped by nearer ones, and more distant elements will be diminished more than the nearer ones. This creates a true 3D scene with a perspective effect.

- **Perspective Effects**
  The perspective effect is controlled via a single property that defines how much the elements will be diminished. This depends on the distance of each element from the screen plane in 3D space.

- **Scroll Animation**
  C1CarouselPanel provides scrolling animation in order to create the effect of elements movement during scroll.

- **Many Control Options**
  C1CarouselPanel provides a number of control options such as alignment of the path, placement of elements and more.

- **Use with any Items Control**
  You can use C1CarouselPanel with any items control to turn a flat list of items into a virtualizing and interactive carousel.

Carousel for WPF Quick Start

The following quick start guide is intended to get you up and running with ComponentOne Carousel for WPF. In this quick start you'll create a simple project using a C1CarouselPanel control. You'll create a new WPF application, create a ListBox with media, add the C1CarouselPanel control in a template, and observe some of the run-time interaction possible with Carousel for WPF.

Note that in this example, you'll use images installed with the ComponentOne Studio for WPF samples. You can substitute other images, but you will have to update the steps below.
Step 1 of 3: Creating the WPF Application

In this step you'll create and set up WPF application. Note that in this example, you'll use images installed with the ComponentOne Studio for WPF samples. If you choose, you can substitute other images but you will have to update the steps below. To set up your project, complete the following steps:

1. From the Visual Studio File menu select New and choose Project.
2. In the New Project dialog box choose a language and WPF Application in the left-side menu, choose .NET Framework 4 in the Framework drop-down list, and enter a name for the project. In this example the application will be named "QuickStart". If you name the project something else, in later steps you may need to change references to "QuickStart" with the name of your project.
3. In the Solution Explorer, right-click the project name and choose Add Reference. In the Add Reference dialog box, locate and select the C1.WPF and C1.WPF.Carousel assemblies and click OK to add references to your project.
4. Open the XAML view of the MainWindow.xaml file; in this quick start you'll use XAML markup and code.
5. Add the XAML namespace to the Window tag with the following markup:

   xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"

   The namespaces will now appear similar to the following:

   <Window x:Class="MainWindow"
           xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
           xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
           xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
           Title="MainWindow" Height="350" Width="525">

   This is a unified namespace that will enable you to work with most ComponentOne WPF or Silverlight controls without adding multiple namespaces.

6. In the Solution Explorer window, right-click the QuickStart item and select Add | New Folder to add a folder. Name the folder "Resources".
7. In the Solution Explorer window, right-click the Resources folder and select Add | New Folder to add a folder. Name the folder "covers".
8. Right-click the covers folder and select Add Existing Item. This will open the Add Existing Item dialog box.
9. In the Add Existing Item dialog box, locate the covers folder in the CarouselSamples sample folder, select all the files, and click Add. By default this sample should be located in the Documents or MyDocuments folder in ComponentOne Samples\Studio for WPF\C1WPFCControlExplorer\CarouselSamples\Resources\covers. Several images will be added to the application.

You've successfully created a new WPF application and added image resources to the application. In the next step you'll add a ListBox and create a C1CarouselPanel template to apply to the ListBox control.

Step 2 of 3: Adding Content to the Application

In the previous step you created a new WPF application and added image resources to the application. In this step you'll add a ListBox and create a C1CarouselPanel template to apply to the ListBox control. Complete the following steps:

1. In the XAML view add a Resources tag just under the Window tag and above the Grid tag with the following markup:
<Window.Resources>
</Window.Resources>

You will add templates within this tag.

2. Add an ItemsPanelTemplate within the Resources tag to define the C1CarouselPanel:

   <ItemsPanelTemplate x:Key="carouselPanelTemplate">
      <c1:C1CarouselPanel Padding="0, 10, 50, 50" VerticalPathAlignment="Center" HorizontalItemAnchorOnPath="Center" VerticalItemAnchorOnPath="Center"/>
   </ItemsPanelTemplate>

3. Add a DataTemplate within the Resources tag:

   <DataTemplate x:Key="carouselItemTemplate">
      <Image Source="{Binding}" Stretch="None" />
   </DataTemplate>

4. Add a Style within the Resources tag to define the C1CarouselPanel's path:

   <Style x:Key="circlePanelStyle" TargetType="ListBox">
      <Setter Property="c1:C1CarouselPanel.HorizontalPathAlignment" Value="Left"/>
      <Setter Property="c1:C1CarouselPanel.VerticalPathAlignment" Value="Top"/>
      <Setter Property="c1:C1CarouselPanel.PerspectiveAngle" Value="55"/>
      <Setter Property="c1:C1CarouselPanel.PerspectiveFactor" Value="0.4"/>
   </Style>

5. Create a ListBox control by adding the following ListBox tag within the Grid tags:

   <ListBox Background="Transparent" Name="carouselListBox" Grid.Row="1" ItemsPanel="{StaticResource carouselPanelTemplate}" ItemTemplate="{StaticResource carouselItemTemplate}" Style="{StaticResource circlePanelStyle}"
   </ListBox>

   Note that this ListBox uses the templates you just added. The ItemsPanel property assigns the C1CarouselPanel to the ListBox via the ItemsPanelTemplate.


7. Add the following import statements at the top of the page of the Code Editor:

   - Visual Basic
     
     Imports System.Windows.Media.Imaging
     Imports C1.WPF
     Imports C1.WPF.Carousel
   
   - C#
     
     using C1.WPF;
using C1.WPF.Carousel;

8. Add code to the main class so it appears like the following:

- **Visual Basic**
  
  ```vbnet
  Public Sub New()
  InitializeComponent()
  InitData()
  End Sub
  ```

- **C#**
  
  ```csharp
  public MainPage()
  {
  InitializeComponent();
  InitData();
  }
  ```

9. Add the following code below the main class:

- **Visual Basic**
  
  ```vbnet
  Private Sub InitData()
  For i As Integer = 101 To 125
  carouselListBox.Items.Add(New BitmapImage(New Uri("Resources/covers/cover" & i & ".jpg", UriKind.Relative)))
  Next
  End Sub
  ```

- **C#**
  
  ```csharp
  private void InitData()
  {
  for (int i = 101; i <= 125; ++i)
  {
  carouselListBox.Items.Add(new BitmapImage(new Uri("Resources/covers/cover" + i + ".jpg", UriKind.Relative)));
  }
  }
  ```

This loads the images you added into the **ListBox** control. Note that if you added different images to your project, you may have to adapt the above code.

In this step you added a **ListBox** and create a **C1CarouselPanel** template to apply to the **ListBox** control. Now all that's left is to run the application.

### Step 3 of 3: Running the Application

Now that you've created a WPF application and added content to it, the only thing left to do is run your application. To run your application and complete the following steps:

1. From the **Debug** menu, select **Start Debugging** to view how your application will appear at run time.

   The application will appear similar to the following:
2. Click on the left-most image. Notice that the carousel rotates in a circular path to the right. This path was specified in the Style resource.

3. Click on the right-most image. Notice that the carousel rotates in a circular path to the left.

4. Click on the horizontal scroll bar in your browser. Notice that that clicking on the scroll bar causes the carousel to rotate.

Congratulations! You've completed the Carousel for WPF quick start and created a simple Silverlight application, added and customized a Carousel for WPF control, and viewed some of the run-time capabilities of the control.

Working with Carousel for WPF

**ComponentOne Carousel for WPF** includes the C1CarouselPanel control, a simple viewer that allows you to add a carousel-like animation to your application. C1CarouselPanel can be used to lay out any control with child elements that can be arranged in an arbitrary panel; for example, you can use C1CarouselPanel as ItemsPanel of ListBox or as MonthsPanel of Scheduler for WPF's C1Calendar. Use the carousel panel for a unique, interactive data presentation.

Using C1CarouselPanel

To use the C1CarouselPanel to create a carousel-like interactive effect, you can set it as an ItemsControl control's panel and assign your visual elements collection to the ItemsControl.Items property. So, for example, in the XAML below an ItemsPanelTemplate template is defined in the Window's resources and includes the C1CarouselPanel panel. An ItemsControl containing arbitrary elements later points to the ItemsPanelTemplate template:

```
<Window x:Class="MainWindow"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
Title="MainWindow" Height="350" Width="525">
<Window.Resources>
```
Customizing the Carousel

You can customize the **C1CarouselPanel** in the example in the Using C1CarouselPanel (page 11) topic by setting properties in either the `<c1:C1CarouselPanel/>` tag in the **ItemsPanelTemplate** or in the `<ItemsControl>` tag. Setting properties in the `<ItemsControl>` tag is possible because all of the properties introduced in carousel are attached dependency properties, providing the ability to change carousel's property at run time.

**In the C1CarouselPanel Tag**

So, for example, to limit the number of visible elements in the C1CarouselPanel, you can set the PageSize property in the `<c1:C1CarouselPanel>` tag:

```xml
<Window.Resources>
  <!-- An ItemsPanelTemplate template defining the ClCarouselPanel. -->
  <ItemsPanelTemplate x:Key="carouselPanel">
    <c1:C1CarouselPanel PageSize="3" />
  </ItemsPanelTemplate>
</Window.Resources>
```

**In the ItemsControl Tag**

To limit the number of visible elements in the C1CarouselPanel, you can also set the PageSize property in the in the `<ItemsControl>` tag:

```xml
<ItemsControl ItemsPanel="{StaticResource carouselPanel}"
             ItemsPanel="{StaticResource carouselPanel}"><Image Width="51" Height="51" Source="image1.png"/>
<Image Width="51" Height="51" Source="image2.png"/>
<Image Width="51" Height="51" Source="image3.png"/>
<Button Height="23" Name="Button1" Width="75">Button</Button>
</ItemsControl>
```
Run-Time Interaction

Users can interact with items in the toolbar and content area of the C1CarouselPanel control at run time. Users can move and drag content in the content area or use the scrollbar to manipulate the items displayed in the content area.

Moving Content

To move content, click on any item in the C1CarouselPanel. The content in the C1CarouselPanel will shift to the left or right according to the path you have specified and the clicked item will become the front-most item. For example, clicking the right-most item in the following image:

![Image of items in a C1CarouselPanel]

Will move that item to the center of the C1CarouselPanel:
Scrolling Content

You can also move the items in the C1CarouselPanel control by clicking in the horizontal scrollbar in your browser:

Click and drag on the thumb button to move content, or click on the scrollbar next to the thumb button to move the content in the direction you choose.
Carousel for WPF Task Based Help

The following task-based help topics assume that you are familiar with Visual Studio and Expression Blend and know how to use the C1Carousel control in general. If you are unfamiliar with the ComponentOne Carousel for WPF product, please see the Carousel for WPF Quick Start (page 7) first.

Each topic in this section provides a solution for specific tasks using the ComponentOne Carousel for WPF product. Most task-based help topics also assume that you have created a new WPF project and added a C1Carousel control to the project – for information about creating the control, see Adding C1Carousel to the Application (page 15).

Adding C1Carousel to the Application

Follow these steps to add a C1Carousel control to your project:

1. From the Visual Studio File menu select New and choose Project.
2. In the New Project dialog box choose a language in the left-side menu, choose .NET Framework 4 in the Framework drop-down list, and enter a name for the project.
3. In the Solution Explorer, right-click the project name and choose Add Reference. In the Add Reference dialog box, locate and select the following assemblies and click OK to add references to your project:
   - C1.WPF.dll
   - C1.WPF.Carousel.dll
4. Add the XAML namespace to the Window tag with the following markup:
   `xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"`

   The namespaces will now appear similar to the following:

   ```xml
   <Window x:Class="QuickStart.MainWindow"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
   xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
   mc:Ignorable="d"
   d:DesignHeight="300"
   d:DesignWidth="400">
   ```

5. In the XAML view add a set of `<Window.Resources>` tag just under the `<Window>` tag and above the `<Grid>` tag with the following markup:

   ```xml
   <Window.Resources>
   </Window.Resources>
   ```

   You will add templates within this tag.

6. Add an `<ItemsPanelTemplate>` within the `<Window.Resources>` tags to define the C1CarouselPanel:

   ```xml
   <ItemsPanelTemplate x:Key="carouselPanelTemplate">
     <c1:C1CarouselPanel Padding="0, 10, 50, 50"
                         VerticalPathAlignment="Center"
                         HorizontalItemAnchorOnPath="Center"
                         VerticalItemAnchorOnPath="Center"/>
   </ItemsPanelTemplate>
   ```
You have now added a C1CarouselPanel to the page. If you run your application now, there will be an empty WPF application displayed.

Adding and Moving Content in C1Carousel

The following Help topics will walk you through adding content to the Carousel control through adding items to the Resources folder and then adding code to call items from the Resources folder. They will also walk you through creating a path along which your content can move. There are several different paths along which you can move content in the C1CarouselPanel control.

Moving Content Along an Elliptical Path

Follow these steps to move content along an Elliptical path:

1. From the Visual Studio File menu select New and choose Project.
2. In the New Project dialog box choose a language in the left-side menu, choose .NET Framework 4 in the Framework drop-down list, and enter a name for the project.
3. In the Solution Explorer, right-click the project name and choose Add Reference. In the Add Reference dialog box, locate and select the following assemblies and click OK to add references to your project:
   - C1.WPF.dll
   - C1.WPF.Carousel.dll
4. Right-click on the yourprojectname project and select Add | Folder from the list. Name the folder Resources.
5. Right-click on the Resources folder and select Add | Folder from the list. Name the new sub-folder covers.
6. Right-click on the covers folder and select Add | Existing Item from the list. For this Help, locate the covers that are included with the C1Carousel_Demo sample application. Select all the files by pressing Shift and then clicking the first and last covers. Click Add to add the files to your folder.
7. Add the XAML namespace to the Window tag with the following markup:
   xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml".
   The namespaces will now appear similar to the following:
   
   <Window x:Class="QuickStart.MainWindow"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
   xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
   mc:Ignorable="d" d:DesignHeight="300" d:DesignWidth="400">
8. In the XAML view add a <Resources> tag just under the <Window> tag and above the <Grid> tag with the following markup:
   
   <Window.Resources>
   </Window.Resources>
   You will add templates within this tag.
9. Add an ItemsPanelTemplate within the Resources tag to define the C1CarouselPanel:
   
   <ItemsPanelTemplate x:Key="carouselPanelTemplate"
10. Insert a `<DataTemplate>` to define the image source:

```xml
<DataTemplate x:Key="carouselItemTemplate">
    <Image Source="{Binding}" Stretch="None" />
</DataTemplate>
```

11. Add the following `<Style>` under the `<DataTemplate>` to set the PathGeometry for the C1Carousel control:

```xml
<Style x:Key="circlePanelStyle" TargetType="ListBox">
    <Setter Property="c1:C1CarouselPanel.HorizontalPathAlignment" Value="Left"/>
    <Setter Property="c1:C1CarouselPanel.VerticalPathAlignment" Value="Top"/>
    <Setter Property="c1:C1CarouselPanel.PerspectiveAngle" Value="55"/>
    <Setter Property="c1:C1CarouselPanel.PerspectiveFactor" Value="0.4"/>
</Style>
```

12. Add the following `ListBox` control within the `<Grid>`:

```xml
<ListBox Background="Transparent" Name="carouselListBox" Grid.Row="1" ItemsPanel="{StaticResource carouselPanelTemplate}" ItemTemplate="{StaticResource carouselItemTemplate}" Style="{StaticResource circlePanelStyle}" />
```

13. Right-click the `Page1.xaml` page and select View Code from the list.

14. Add the following namespaces to the top of the page:

   - C1.WPF.dll
   - C1.WPF.Carousel.dll

15. Insert the following method directly below the `InitializeComponent()` method:

   - Visual Basic
     ```vbnet
     InitData()
     ```
   - C#
     ```csharp
     InitData();
     ```
16. Call the following method to populate the C1CarouselPanel with data:

- Visual Basic
  
  ```vbs
  Private Sub InitData()
    For i As Integer = 101 To 140
      carouselListBox.Items.Add(New BitmapImage(New Uri("Resources/covers/cover" & i & ".jpg", UriKind.Relative)))
    Next
  End Sub
  ```

- C#
  
  ```cs
  private void InitData()
  {
    for (int i = 101; i <= 140; ++i)
    {
      carouselListBox.Items.Add(new BitmapImage(new Uri("Resources/covers/cover" + i + ".jpg", UriKind.Relative)));
    }
  }
  ```

17. Press F5 to run your application. The carousel should resemble the following image:

![Carousel Image]

**Moving Content Along a ZigZag Path**

Follow these steps:

1. From the Visual Studio File menu select New and choose Project.
2. In the New Project dialog box choose a language in the left-side menu, choose .NET Framework 4 in the Framework drop-down list, and enter a name for the project.
3. In the Solution Explorer, right-click the project name and choose **Add Reference**. In the **Add Reference** dialog box, locate and select the following assemblies and click **OK** to add references to your project:
   - C1.WPF.dll
   - C1.WPF.Carousel.dll

4. Right-click on the *yourprojectname*.Web project and select **Add | Folder** from the list. Name the folder **Resources**.

5. Right-click on the **Resources** folder and select **Add | Folder** from the list. Name the new sub-folder **covers**.

6. Right-click on the **covers** folder and select **Add | Existing Item** from the list. For this Help, locate the covers that are included with the **C1Carousel_Demo** sample application. Select all the files by pressing Shift and then clicking the first and last covers. Click **Add** to add the files to your folder.

7. Add the XAML namespace to the Window tag with the following markup:
   ```xml
   xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
   ```
   The namespaces will now appear similar to the following:
   ```xml
   <Window x:Class="QuickStart.MainWindow"
     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
     xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
     xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
     mc:Ignorable="d"
     d:DesignHeight="300"
     d:DesignWidth="400">
   ```

8. In the XAML view add a `<Resources>` tag just under the `<Window>` tag and above the `<Grid>` tag with the following markup:
   ```xml
   <Window.Resources>
   <ResourceDictionary>
   </Window.Resources>
   ```
   You will add templates within this tag.

9. Add an `<ItemsPanelTemplate>` within the `<Window.Resources>` tags to define the C1CarouselPanel:
   ```xml
   <ItemsPanelTemplate x:Key="carouselPanelTemplate">
     <c1:C1CarouselPanel Padding="0, 10, 50, 50" VerticalPathAlignment="Center" HorizontalItemAnchorOnPath="Center" VerticalItemAnchorOnPath="Center"/>
   </ItemsPanelTemplate>
   ```

10. Insert a `<DataTemplate>` to define the image source:
    ```xml
    <DataTemplate x:Key="carouselItemTemplate">
      <Image Source="{Binding}" Stretch="None"/>
    </DataTemplate>
    ```

11. Add the following `<Style>` under the `<DataTemplate>` to set the PathGeometry for the C1CarouselPanel control:
    ```xml
    <Style x:Key="zigzagPanelStyle" TargetType="ListBox">
      <Setter Property="c1:C1CarouselPanel.PathGeometry">
        Value="M 259.228,50.3333C 361.691,64.8311 464.153,79.3289 525.213,93.8267C 586.274,108.324 605.932,122.822 607.421,137.32C"
      </Setter>
    </Style>
    ```
12. Add the following ListBox control within the <Grid>:

```xml
<ListBox Background="Transparent" Name="carouselListBox"
    Grid.Row="1"
    ItemsPanel="{StaticResource carouselPanelTemplate}"
    ItemTemplate="{StaticResource carouselItemTemplate}"
    Style="{StaticResource zigzagPanelStyle}"/>
```

13. Right-click the Page1.xaml page and select View Code from the list.

14. Add the following namespaces to the top of the page:

- C1.WPF.dll
- C1.WPF.Carousel.dll

15. Insert the following method directly below the InitializeComponent() method:

- Visual Basic
  ```vbnet
  Private Sub InitData()
  For i As Integer = 101 To 140
      carouselListBox.Items.Add(New BitmapImage(New Uri("Resources/covers/cover" & i & ".jpg", UriKind.Relative)))
  Next
  End Sub
  ```

- C#
  ```cs
  private void InitData()
  ```

16. Call the following method to populate the C1CarouselPanel with data:

- Visual Basic
  ```vbnet
  Private Sub InitData()
      For i As Integer = 101 To 140
          carouselListBox.Items.Add(New BitmapImage(New Uri("Resources/covers/cover" & i & ".jpg", UriKind.Relative)))
      Next
  End Sub
  ```

- C#
  ```cs
  private void InitData()
  ```
{ for (int i = 101; i <= 140; ++i) 
{ carouselListBox.Items.Add(new BitmapImage(new Uri("Resources/cover" + i + ".png", UriKind.Relative))); 
} 

17. Press F5 to run your application. The carousel should resemble the following image:

![Image](image-url)

### Moving Content Along a Diagonal Path

Follow these steps to move items along a diagonal path:

1. From the Visual Studio File menu select New and choose Project.
2. In the New Project dialog box choose a language in the left-side menu, choose .NET Framework 4 in the Framework drop-down list, and enter a name for the project.
3. In the Solution Explorer, right-click the project name and choose Add Reference. In the Add Reference dialog box, locate and select the following assemblies and click OK to add references to your project:
   - C1.WPF.dll
   - C1.WPF.Carousel.dll
4. Right-click on the **yourprojectname**.Web project and select **Add | Folder** from the list. Name the folder **Resources**.

5. Right-click on the **Resources** folder and select **Add | Folder** from the list. Name the new sub-folder **covers**.

6. Right-click on the **covers** folder and select **Add | Existing Item** from the list. For this Help, locate the covers that are included with the **C1Carousel_Demo** sample application. Select all the files by pressing Shift and then clicking the first and last covers. Click **Add** to add the files to your folder.

7. Add the XAML namespace to the Window tag with the following markup:

   xmlns:cl="http://schemas.componentone.com/winfx/2006/xaml".

   The namespaces will now appear similar to the following:

   ```xml
   <Window x:Class="QuickStart.MainWindow"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
   xmlns:cl="http://schemas.componentone.com/winfx/2006/xaml"
   mc:Ignorable="d" d:DesignHeight="300" d:DesignWidth="400">
   
   8. In the XAML view add a `<Window.Resources>` tag just under the `<Window>` tag and above the `<Grid>` tag with the following markup:

   ```xml
   <Window.Resources>
   
   </Window.Resources>
   
   You will add templates within this tag.

9. Add an `<ItemsPanelTemplate>` within the `<Window.Resources>` tags to define the C1CarouselPanel:

   ```xml
   <ItemsPanelTemplate x:Key="carouselPanelTemplate">
   
   <c1:C1CarouselPanel Padding="0, 10, 50, 50" VerticalPathAlignment="Center" HorizontalItemAnchorOnPath="Center" VerticalItemAnchorOnPath="Center"/>
   
   </ItemsPanelTemplate>
   
   10. Insert a `<DataTemplate>` to define the image source:

   ```xml
   <DataTemplate x:Key="carouselItemTemplate">
   
   <Image Source="{Binding}" Stretch="None" />
   
   </DataTemplate>
   
   11. Add the following `<Style>` under the `<DataTemplate>` to set the PathGeometry for the C1CarouselPanel control:

   ```xml
   <Style x:Key="diagonalPanelStyle" TargetType="ListBox">
   
   <Setter Property="c1:C1CarouselPanel.PathGeometry" Value="M 216, 60 L 600, 390"/>
   
   <Setter Property="c1:C1CarouselPanel.HorizontalPathAlignment" Value="Left"/>
   
   <Setter Property="c1:C1CarouselPanel.VerticalPathAlignment" Value="Top"/>
   
   <Setter Property="c1:C1CarouselPanel.PerspectiveAngle" Value="65"/>
   ```
<Setter Property="c1:C1CarouselPanel.PerspectiveFactor" Value="0.12"/>
</Style>

12. Add the following <ListBox> control within the <Grid>:

<ListBox Background="Transparent" Name="carouselListBox"
Grid.Row="1"
ItemsPanel="{StaticResource carouselPanelTemplate}"
ItemTemplate="{StaticResource carouselItemTemplate}"
Style="{StaticResource diagonalPanelStyle}"/>

13. Right-click the Page1.xaml page and select View Code from the list.

14. Add the following namespaces to the top of the page:

- C1.WPF.dll
- C1.WPF.Carousel.dll

15. Insert the following method directly below the InitializeComponent() method:

- Visual Basic
  InitData()

- C#
  InitData();

16. Call the following method to populate the C1CarouselPanel with data:

- Visual Basic
  Private Sub InitData()
      For i As Integer = 101 To 140
          carouselListBox.Items.Add(New BitmapImage(New Uri("Resources/covers/cover" & i & ".jpg", UriKind.Relative)))
      Next
  End Sub

- C#
  private void InitData()
  {
      for (int i = 101; i <= 140; ++i)
      {
          carouselListBox.Items.Add(new BitmapImage(new Uri("Resources/covers/cover" + i + ".jpg", UriKind.Relative)));
      }
  }

Press F5 to run your application. The carousel should resemble the following image:
Moving Content Along a Parabolic Path

Follow these steps:

1. From the Visual Studio File menu select New and choose Project.

2. In the New Project dialog box choose a language in the left-side menu, choose .NET Framework 4 in the Framework drop-down list, and enter a name for the project.

3. In the Solution Explorer, right-click the project name and choose Add Reference. In the Add Reference dialog box, locate and select the following assemblies and click OK to add references to your project:
   - C1.WPF.dll
   - C1.WPF.Carousel.dll

4. Right-click on the yourprojectname .Web project and select Add | Folder from the list. Name the folder Resources.

5. Right-click on the Resources folder and select Add | Folder from the list. Name the new sub-folder covers.

6. Right-click on the covers folder and select Add | Existing Item from the list. For this Help, locate the covers that are included with the C1Carousel_Demo sample application. Select all the files by pressing Shift and then clicking the first and last covers. Click Add to add the files to your folder.

7. Add the XAML namespace to the Window tag with the following markup:
   
   xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"

   The namespaces will now appear similar to the following:

   <Window x:Class="QuickStart.MainWindow"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
   xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
   mc:Ignorable="d" d:DesignHeight="300" d:DesignWidth="400">
8. In the XAML view add a `<Window.Resources>` tag just under the `<Window>` tag and above the `<Grid>` tag with the following markup:

```xml
<Window.Resources>
</Window.Resources>
```

You will add templates within this tag.

9. Add an `ItemsPanelTemplate` within the `Resources` tag to define the C1CarouselPanel:

```xml
<ItemsPanelTemplate x:Key="carouselPanelTemplate">
  <c1:C1CarouselPanel Padding="0, 10, 50, 50" VerticalPathAlignment="Center" HorizontalItemAnchorOnPath="Center" VerticalItemAnchorOnPath="Center"/>
</ItemsPanelTemplate>
```

10. Insert a `<DataTemplate>` to define the image source:

```xml
<DataTemplate x:Key="carouselItemTemplate">
  <Image Source="{Binding}" Stretch="None"/>
</DataTemplate>
```

11. Add the following `<Style>` under the `<DataTemplate>` to set the PathGeometry for the C1Carousel control:

```xml
<Style x:Key="parabolaPanelStyle" TargetType="ListBox">
  <Setter Property="c1:C1CarouselPanel.HorizontalPathAlignment" Value="Left"/>
  <Setter Property="c1:C1CarouselPanel.VerticalPathAlignment" Value="Top"/>
  <Setter Property="c1:C1CarouselPanel.PerspectiveAngle" Value="90"/>
  <Setter Property="c1:C1CarouselPanel.PerspectiveFactor" Value="-0.317"/>
</Style>
```

12. Add the following `ListBox` control within the `<Grid>`:

```xml
<ListBox Background="Transparent" Name="carouselListBox" Grid.Row="1">
  ItemsPanel="{StaticResource carouselPanelTemplate}"
  ItemTemplate="{StaticResource carouselItemTemplate}"
  Style="{StaticResource parabolaPanelStyle}"/>
```

13. Right-click the `Page1.xaml` page and select `View Code` from the list.

14. Add the following namespaces to the top of the page:

- `C1.WPF.dll`
15. Insert the following method directly below the `InitializeComponent()` method:

- **Visual Basic**
  
  ```vbnet
  InitData()
  ```

- **C#**
  
  ```csharp
  InitData();
  ```

16. Call the following method to populate the C1CarouselPanel with data:

- **Visual Basic**
  
  ```vbnet
  Private Sub InitData()
  For i As Integer = 101 To 140
      carouselListBox.Items.Add(New BitmapImage(New Uri("Resources/covers/cover" & i & ".jpg", UriKind.Relative)))
  Next
  End Sub
  ```

- **C#**
  
  ```csharp
  private void InitData()
  { }
  for (int i = 101; i <= 140; ++i)
  { }
      carouselListBox.Items.Add(new BitmapImage(new Uri("Resources/covers/cover" + i + ".jpg", UriKind.Relative)));
  }
  ```

Press F5 to run your application. The carousel should resemble the following image:

![Carousel Images](image-url)

**Moving Content Along a Spiral Path**

Follow these steps:

1. From the Visual Studio **File** menu select **New** and choose **Project**.
2. In the **New Project** dialog box choose a language in the left-side menu, choose **.NET Framework 4** in the **Framework** drop-down list, and enter a name for the project.

3. In the Solution Explorer, right-click the project name and choose **Add Reference**. In the **Add Reference** dialog box, locate and select the following assemblies and click **OK** to add references to your project:
   - C1.WPF.dll
   - C1.WPF.Carousel.dll

4. Right-click on the **yourprojectname.Web** project and select **Add | Folder** from the list. Name the folder **Resources**.

5. Right-click on the **Resources** folder and select **Add | Folder** from the list. Name the new sub-folder **covers**.

6. Right-click on the **covers** folder and select **Add | Existing Item** from the list. For this Help, locate the covers that are included with the **C1Carousel_Demo** sample application. Select all the files by pressing **Shift** and then clicking the first and last covers. Click **Add** to add the files to your folder.

7. Add the XAML namespace to the **Window** tag with the following markup:
   
   ```xml
   xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
   ```

   The namespaces will now appear similar to the following:

   ```xml
   <Window x:Class="QuickStart.MainWindow"
           xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
           xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
           xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
           xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
           mc:Ignorable="d"
           d:DesignHeight="300"
           d:DesignWidth="400">
   ```

8. In the XAML view add a **<Window.Resources>** tag just under the **<Window>** tag and above the **<Grid>** tag with the following markup:

   ```xml
   <Window.Resources>
   <Grid>
   <Window.Resources>
   ```

   You will add templates within this tag.

9. Add an **ItemsPanelTemplate** within the **Resources** tag to define the **C1CarouselPanel**:

   ```xml
   <ItemsPanelTemplate x:Key="carouselPanelTemplate">
       <c1:C1CarouselPanel Padding="0, 10, 50, 50"
                           VerticalPathAlignment="Center"
                           HorizontalItemAnchorOnPath="Center"
                           VerticalItemAnchorOnPath="Center"/>
   </ItemsPanelTemplate>
   ```

10. Insert a **<DataTemplate>** to define the image source:

    ```xml
    <DataTemplate x:Key="carouselItemTemplate">
        <Image Source="{Binding}" Stretch="None" />
    </DataTemplate>
    ```

11. Add the following **<Style>** under the **<DataTemplate>** to set the PathGeometry for the **C1CarouselPanel** control:

    ```xml
    <Style x:Key="spiralPanelStyle" TargetType="ListBox">
        <Setter Property="c1:C1CarouselPanel.PathGeometry" Value="F1 M 426.39877,214.56664"/>
    </Style>
    ```
12. Add the following ListBox control within the <Grid>:

```xml
<ListBox Background="Transparent" Name="carouselListBox"
  Grid.Row="1"
  ItemsPanel="{StaticResource carouselPanelTemplate}"
  ItemTemplate="{StaticResource carouselItemTemplate}"
  Style="{StaticResource sprialPanelStyle}"/>
```
13. Right-click the **Page1.xaml** page and select **View Code** from the list.

14. Add the following namespaces to the top of the page:
   - `C1.WPF.dll`
   - `C1.WPF.Carousel.dll`

15. Insert the following method directly below the `InitializeComponent()` method:
   - Visual Basic
     ```vbnet
     InitData()
     ```
   - C#
     ```csharp
     InitData();
     ```

16. Call the following method to populate the C1CarouselPanel with data:
   - Visual Basic
     ```vbnet
     Private Sub InitData()
         For i As Integer = 101 To 140
             carouselListBox.Items.Add(New BitmapImage(New Uri("Resources/covers/cover" & i & ".jpg", UriKind.Relative)))
         Next
     End Sub
     ```
   - C#
     ```csharp
     private void InitData()
     {
         for (int i = 101; i <= 140; ++i)
         {
             carouselListBox.Items.Add(new BitmapImage(new Uri("Resources/covers/cover" + i + ".jpg", UriKind.Relative)));
         }
     }
     ```

Press F5 to run your application. The items in the C1Carousel control should move through a spiral.

---

**Using the ScrollViewer**

You can allow users to scroll through C1CarouselPanel content either by clicking or by using the **ScrollViewer**. This topic will walk you through setting the **ScrollViewer** to visible and setting some of its properties.

This Help topic assumes that you have completed one of the previous Adding and Moving Content topics and are using that project for this Help.

Follow these steps to set the **ScrollViewer**:

1. Select the C1CarouselPanel control in your application to open the Properties window.
2. Locate the CanHorizontallyScroll property in the list and make sure it is checked.
3. Locate the ScrollOwner property in the list and use the drop-down list to select **ScrollViewer**. This will open the ScrollOwner sub-properties.

4. Locate the **HorizontalScrollBar** property and use the drop-down list to set it to **Visible**. Scroll down to the **VerticalScrollBar** property and set that to **Disabled**.

5. Press F5 to run your application. It should appear as in the following image:

![Image of scrollable content](image_url)

You will be able to scroll through the content using the scrollbar at the bottom of the page.

**Limiting the Number of Visible Elements**

You can customize the number of visible elements in the C1CarouselPanel by setting properties in the `<c1:C1CarouselPanel>` tag in the **ItemsPanelTemplate**.

**In the C1CarouselPanel Tag**

To limit the number of visible elements in the C1CarouselPanel follow these steps:
1. From the Visual Studio File menu select New and choose Project.
2. In the New Project dialog box choose a language in the left-side menu, choose .NET Framework 4 in the Framework drop-down list, and enter a name for the project.
3. In the Solution Explorer, right-click the project name and choose Add Reference. In the Add Reference dialog box, locate and select the following assemblies and click OK to add references to your project:
   - C1.WPF.dll
   - C1.WPF.Carousel.dll
4. Right-click on the yourprojectname .Web project and select Add | Folder from the list. Name the folder Resources.
5. Right-click on the Resources folder and select Add | Folder from the list. Name the new sub-folder covers.
6. Right-click on the covers folder and select Add | Existing Item from the list. For this Help, locate the covers that are included with the C1Carousel_Demo sample application. Select all the files by pressing Shift and then clicking the first and last covers. Click Add to add the files to your folder.
7. Add the XAML namespace to the Window tag with the following markup:
   ```xml
   xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
   ``
   The namespaces will now appear similar to the following:
   ```xml
   <Window x:Class="QuickStart.MainWindow"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:c1="http://schemas.componentone.com/winfx/2006/xaml"
   mc:Ignorable="d"
   d:DesignHeight="300" d:DesignWidth="400">
   </Window>
   ```
8. In the XAML view add a <Resources> tag just under the <Window> tag and above the <Grid> tag with the following markup:
   ```xml
   <Window.Resources>
   </Window.Resources>
   ```
   You will add templates within this tag.
9. Add an ItemsPanelTemplate within the Resources tag to define the C1CarouselPanel:
   ```xml
   <ItemsPanelTemplate x:Key="carouselPanelTemplate">
   <c1:C1CarouselPanel Padding="0, 10, 50, 50" VerticalPathAlignment="Center" HorizontalItemAnchorOnPath="Center" VerticalItemAnchorOnPath="Center"/>
   </ItemsPanelTemplate>
   ```
10. Locate the <c1:C1CarouselPanel> tag and add the following XAML markup to limit the number of visible elements:
    ```xml
    PageSize="3"
    ```
    The total <c1:C1CarouselPanel> markup should resemble the following:
    ```xml
    <c1:C1CarouselPanel Padding="0, 10, 50, 50" VerticalPathAlignment="Center" HorizontalItemAnchorOnPath="Center" VerticalItemAnchorOnPath="Center" PageSize="3"/>
    ```
11. Insert a <DataTemplate> to define the image source:
12. Add the following <Style> under the <DataTemplate> to set the PathGeometry for the C1CarouselPanel control:

```xml
<Style x:Key="diagonalPanelStyle" TargetType="ListBox">
    <Setter Property="c1:C1CarouselPanel.PathGeometry" Value="M 216, 60 L 600, 390"/>
    <Setter Property="c1:C1CarouselPanel.HorizontalPathAlignment" Value="Left"/>
    <Setter Property="c1:C1CarouselPanel.VerticalPathAlignment" Value="Top"/>
    <Setter Property="c1:C1CarouselPanel.PerspectiveAngle" Value="65"/>
    <Setter Property="c1:C1CarouselPanel.PerspectiveFactor" Value="0.12"/>
</Style>
```

13. Add the following ListBox control within the <Grid>:

```xml
<ListBox Background="Transparent" Name="carouselListBox" Grid.Row="1">
    <ItemsPanel>{StaticResource carouselPanelTemplate}</ItemsPanel>
    <ItemTemplate>{StaticResource carouselItemTemplate}</ItemTemplate>
    <Style>{StaticResource diagonalPanelStyle}</Style>
</ListBox>
```

14. Right-click the Page1.xaml page and select View Code from the list.

15. Add the following namespaces to the top of the page:

- C1.WPF.dll
- C1.WPF.Carousel.dll

16. Insert the following method directly below the InitializeComponent() method:

- Visual Basic
  ```vbnet
  InitData()
  ```
- C#
  ```csharp
  InitData();
  ```

17. Call the following method to populate the C1CarouselPanel with data:

- Visual Basic
  ```vbnet
  Private Sub InitData()
      For i As Integer = 101 To 140
          carouselListBox.Items.Add(New BitmapImage(New Uri("Resources/covers/cover" & i & ".jpg", UriKind.Relative)))
      Next
  ```
End Sub

- C#
  private void InitData()
  {
    for (int i = 101; i <= 140; ++i)
    {
      carouselListBox.Items.Add(new BitmapImage(new Uri("Resources/covers/cover" + i + ".jpg", UriKind.Relative)));
    }
  }

Press F5 to run your application. The carousel should resemble the following image: