M03 WPF Common Controls

Author: Yi Chen
Date: 2021-02-02

Subject

Table of Contents

List of Tables

List of Figures

1. WPF Common Controls

Using Windows Presentation Foundation (WPF) .NET create an app which animates a series of image sequences.

2. Project Requirements

- *Use WPF .NET (.NET Core 3.1.0) to make an application which will display a series of images in sequence.*
- *Use the supplied image sequence below or find your own online (make sure you have the rights to use).*
- *Create the following button controls:*
- *Start Animation: This button will begin the animation, rotating through the images at specified interval.*
- *Stop Animation: This will stop the sequence from occurring.*
- *Previous Frame: Displays the previous frame.*
- *Next Frame: Displays the next frame.*
- *Create a slider control to allow the user to select the interval speed.*
- *Create a control (your choice) to allow the user to select forwards/backwards animation.*
- *Create a control (your choice) to allow the user to select the rotation angle of the images.*
- *Ensure your application has appropriate labels and title.*

2.1. Derived Requirements

1. *A prepared PDF document with your commented code.*

    - *Document should have relevant meta information and be well formatted.*

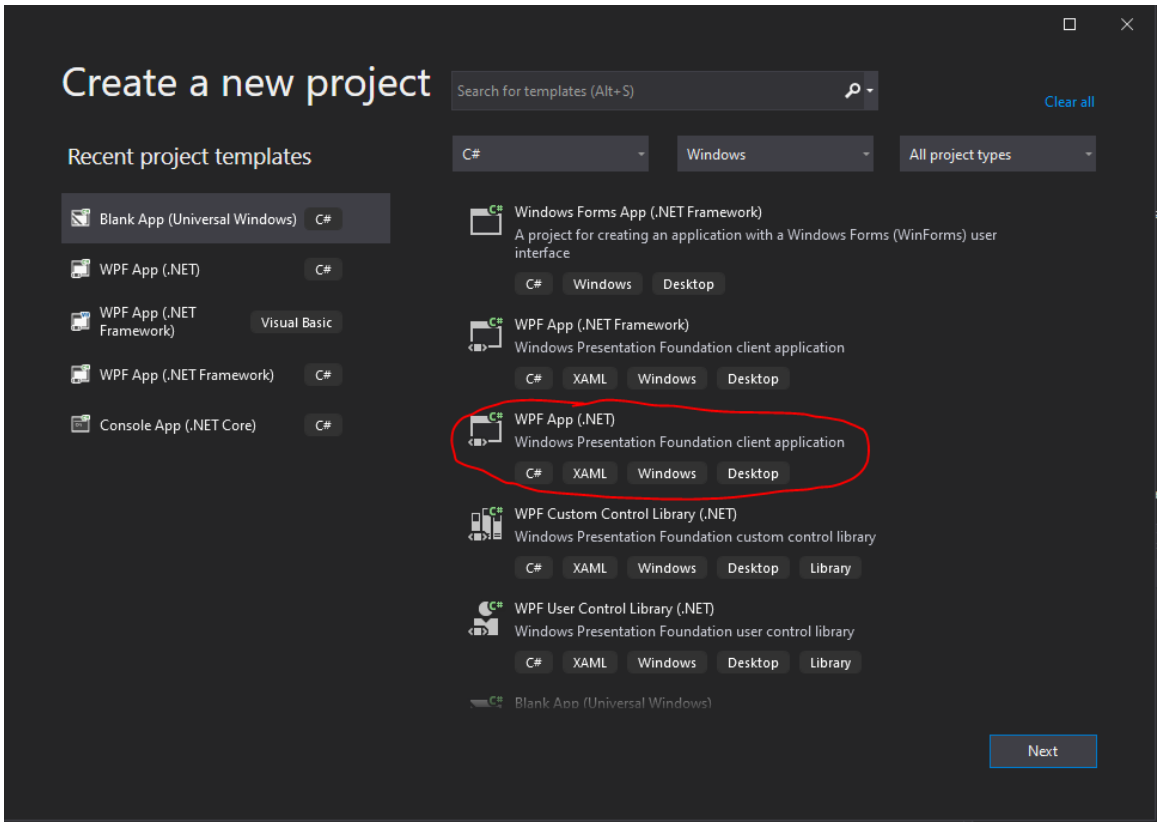
2. *Application executable in zipped format (do not submit a .exe file to Brightspace).*

3. Design Plans

Use WPF .NET (.NET Core 3.1.0) to make an application which will display a series of images in sequence. Common controls to implement code.

3.1. Build Project

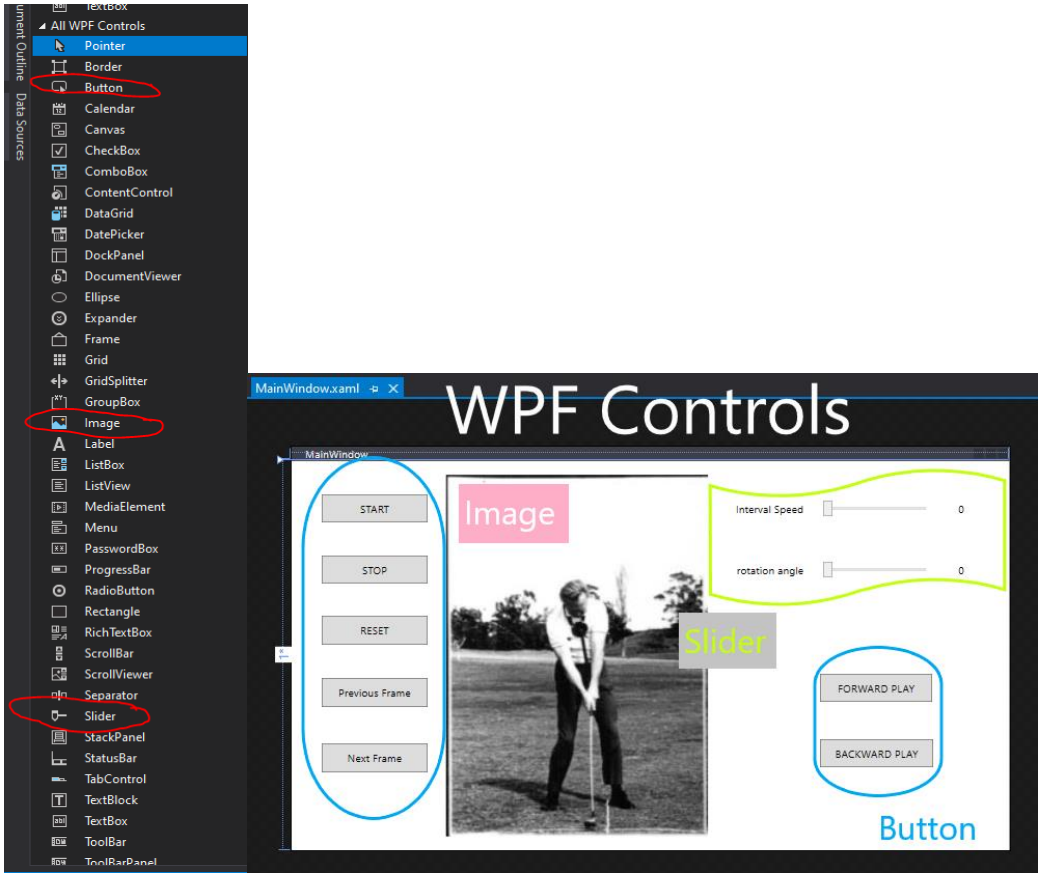
Build a new project use WPD App



3.1.1. Design Components

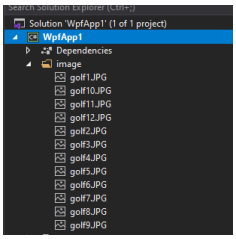
Use WPF controls to make all components.

Subject



3.1.2. WPF Controls - Image

WPF Controls Image common – source – load image files where collected at image file of the project.



Main XAML
<Image x:Name="imageHolder" Grid.RowSpan="2" Margin="172,3,336,3" Source="/golf1.JPG" RenderTransformOrigin="0.521,0.507"/>
MainWindows.xmal.cs

i++;
 	
if (i > 12)
{
i = 1;
}

```
imageHolder.Source = new BitmapImage(new Uri("/golf" + i + ".JPG", UriKind.Relative));
```

3.1.3. Button

Represents a Windows button control, which reacts to the Click event.

Table 1: Button Click Event

START

```
private void start_btn_Click(object sender, RoutedEventArgs e)
{
    aTimer = new System.Windows.Threading.DispatcherTimer();
    aTimer.Tick += new System.EventHandler(OnTimeEvent);
    aTimer.Interval = new TimeSpan(0, 0, 0, 0, (int)interval_slider.Value);
    aTimer.Start();
}

/* picture move frame to frame by use counter */
private void OnTimeEvent(object source, System.EventArgs e)
{
    counter += 1;
    i += 5;
    imageHolder.Source = new BitmapImage(new Uri("/golf" + counter + ".JPG", UriKind.Relative));

    if (counter > 12)
    {
        counter = 1;
    }

    imageHolder.Source = new BitmapImage(new Uri("/golf" + counter + ".JPG", UriKind.Relative));
    imageHolder.RenderTransform = new RotateTransform(i);

}
```

Previous Frame

Next Frame

```
/*Next page*/
private void nt_btn_Click(object sender, RoutedEventArgs e)
{
    i++;

    if (i > 12)
    {
        i = 1;
    }

    imageHolder.Source = new BitmapImage(new Uri("/golf" + i + ".JPG", UriKind.Relative));
}

/*Back page*/
private void pf_btn_Click(object sender, RoutedEventArgs e)
{
    i--;

    if (i < 1)
    {
        i = 12;
    }

    imageHolder.Source = new BitmapImage(new Uri("/golf" + i + ".JPG", UriKind.Relative));

}
```

Subject

STOP

RESET

```
/*Stop buttom, when click stop current page */
private void stop_btn_Click(object sender, RoutedEventArgs e)
{
    aTimer.Stop();
}

/*reset the frames to inital ang 0 and page 1*/
private void reset_btn_Click(object sender, RoutedEventArgs e)
{
    imageHolder.Source = new BitmapImage(new Uri("/golf1.JPG", UriKind.Relative));
    imageHolder.RenderTransform = new RotateTransform(0);
}
```

FORWARD PLAY

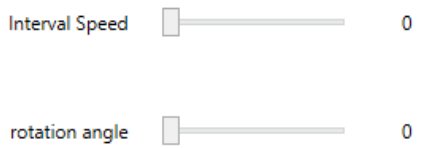
BACKWARD PLAY

```
private void Forward_btn_Click(object sender, RoutedEventArgs e)
{
    aTimer = new System.Windows.Threading.DispatcherTimer();
    aTimer.Tick += new System.EventHandler(ForwordTimeEvent);
    aTimer.Interval = new TimeSpan(0, 0, 0, 0, (int)interval_slider.Value);
    aTimer.Start();
}

/*picture move frame to frame by use counter*/
private void ForwordTimeEvent(object source, System.EventArgs e)
{
    counter += 1;

    if (counter > 12)
    {
        counter = 1;
    }

    imageHolder.Source = new BitmapImage(new Uri("/golf" + counter + ".JPG", UriKind.Relative));

}

private void backward_Click(object sender, RoutedEventArgs e)
{
    aTimer = new System.Windows.Threading.DispatcherTimer();
    aTimer.Tick += new System.EventHandler(BackwordTimeEvent);
    aTimer.Interval = new TimeSpan(0, 0, 0, 0, (int)interval_slider.Value);
    aTimer.Start();
}

private void BackwordTimeEvent(object source, System.EventArgs e)
{

    counter--;


    if (counter < 1)
    {
        counter = 12;
    }

    imageHolder.Source = new BitmapImage(new Uri("/golf" + counter + ".JPG", UriKind.Relative));
}
```

3.2. Implement Slider

Interval_slider.value is be used to Start button for user input interval
Rotation button can be used to user change angle of the image.

Figure 1: Interval and rotation Slider




```
/*interval slider change the animation interval*/
private void interval_slider_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)
{

    interval_count.Content = (int)interval_slider.Value;

}

private void rotation_slider_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)
{

    counter ++;
    if (counter > 12)
    {
        counter = 1;
    }

    counter --;
    if (counter < 1)
    {
        counter = 12;
    }


    imageHolder.Source = new BitmapImage(new Uri("/golf" + counter + ".JPG", UriKind.Relative));
    imageHolder.RenderTransform = new RotateTransform((int)rotation_slider.Value );
    angle.Content = (int)rotation_slider.Value;

}
```

```
/*interval slider change the animation interval*/
private void interval_slider_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)
{

    interval_count.Content = (int)interval_slider.Value;

}

/*Ratation Slider*/
private void rotation_slider_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)
{

    counter ++;
    if (counter > 12)
    {
        counter = 1;
    }

    counter --;
    if (counter < 1)
    {
        counter = 12;
    }


    imageHolder.Source = new BitmapImage(new Uri("/golf" + counter + ".JPG", UriKind.Relative));
    imageHolder.RenderTransform = new RotateTransform((int)rotation_slider.Value );
    angle.Content = (int)rotation_slider.Value;

}
```

3.3. Implement DispatcherTimer.Interval Property

The following code creates a DispatcherTimer. A new DispatcherTimer object named aTimer is created. The event handler dispatcherTimer_Tick is added to the Tick event. The Interval is set to "User input by slider" second using a TimeSpan object.

Table 2: DispatcherTimer.Interval Table

DispatcherTimer.Interval

`aTimer = new System.Windows.Threading.DispatcherTimer();`
`aTimer.Tick += new System.EventHandler(OnTimeEvent);`
`aTimer.Interval = new TimeSpan(0, 0, 0, 0, (int)interval_slider.Value);`
`aTimer.Start();`