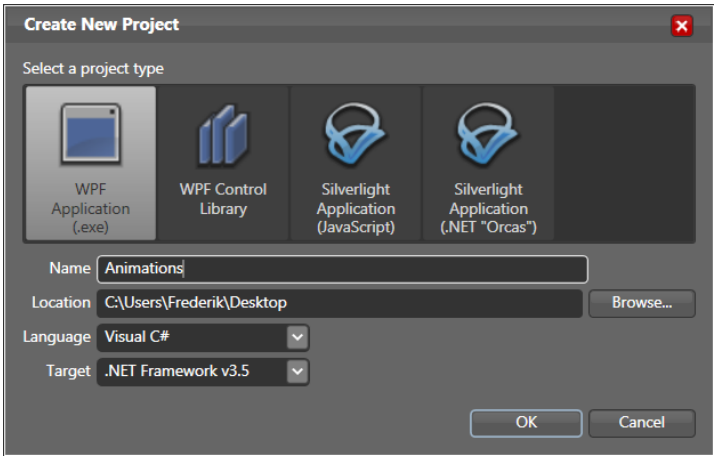
Control animations from code behind

Step 1: create a new project

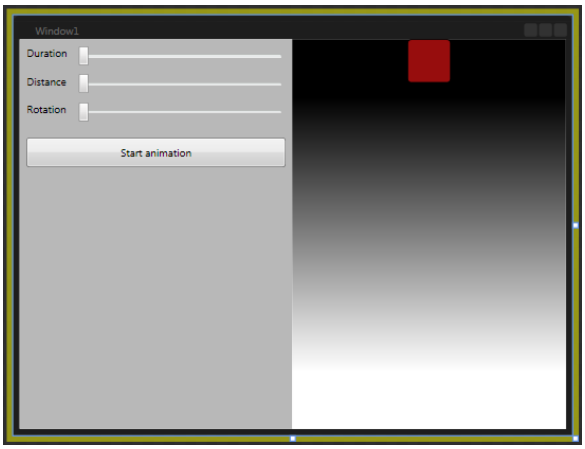
I use the beta 2 of Visual Studio 2008 (codename Orcas) and Expression Blend 2.0 September preview for this tutorial. You can download the beta2 form http://msdn2.microsoft.com/en-us/vstudio/aa700831.aspx. The preview can be downloaded from http://www.microsoft.com/expression/products/features.aspx?key=blend2preview.

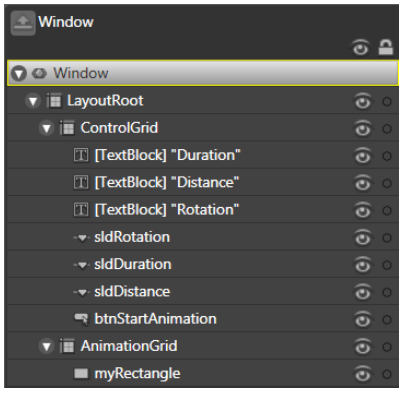
- Start Expression Blend and choose File → New → Project.
- Choose Visual C# as language and a WPF Application as template. Also fill in a name for the project and a location.



Step 2: place the controls on the stage

- Split the stage in two parts and place a grid in the left part and the right part.
- Give the grids a backgroundcolor
- Put a rectangle in the right grid and give it a backgroundcolor
- In the left part, place 3 textblocks or labels, 3 sliders and a button
- The textblocks and sliders will be used to adjust the distance, duration and rotation of the animation



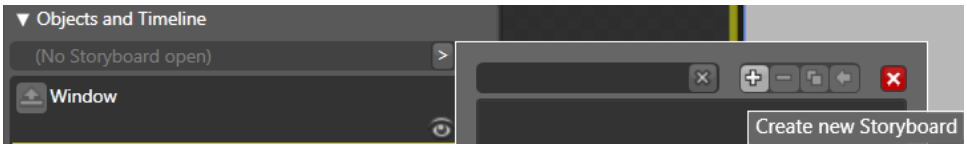


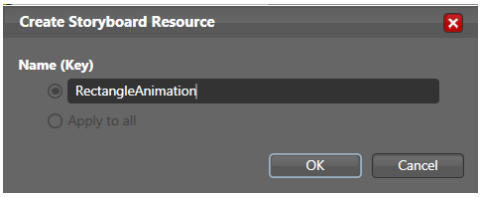
- Set the property AutoToolTipPlacement to TopLeft for the 3 sliders. I also set the maximum and minimum values for the sliders:
- sldRotation (number of degrees that the rectangle can rotate):
    - Minimum: 30
    - Maximum: 360
- sldDuration (number of seconds needed to complete the animation):
    - Minimum: 1
    - Maximum: 10
- sldDistance (number of pixels that the rectangle can be moved):
    - Minimum: 100
    - Maximum: 600

Step 3: Create the animation

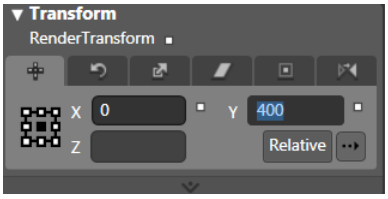
In this step, we will create the animation in Expression Blend. In the next step, we will manipulate the animation in Visual Studio.

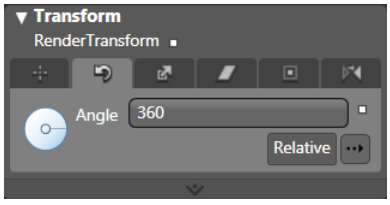
- In the Objects and Timeline panel, first click the > button and then the + button to create a new storyboard. Name the storyboard “RectangleAnimation” and click ok.





- Everything you change will be recorded (if you see the red border) in the animation.
- Select the rectangle in the right part of the stage.
- Move the playhead (yellow triangle) to five seconds
- Go to the section Transform in the property panel.
- Rotate the rectangle 360 degrees and set the Y translation to 400
- Now you should see the rectangle at the bottom of the stage.





- Close the storyboard and go to the triggers panel
- Remove the Window.Loaded trigger, because we will start the animation from code behind and we won’t start it when the window is loaded.
- Build your solution and Blend can be closed.

Step 4: Control the animation

If you look at the xaml code for the animation, you will see the following:

```
<Storyboard x:Key="RectangleAnimation">
        <DoubleAnimationUsingKeyFrames BeginTime="00:00:00"
Storyboard.TargetName="myRectangle"
Storyboard.TargetProperty="(UIElement.RenderTransform).(TransformGroup.Children)[3].(Translate
Transform.Y)">
                <SplineDoubleKeyFrame KeyTime="00:00:05" Value="400"/>
        </DoubleAnimationUsingKeyFrames>
        <DoubleAnimationUsingKeyFrames BeginTime="00:00:00"
Storyboard.TargetName="myRectangle"
Storyboard.TargetProperty="(UIElement.RenderTransform).(TransformGroup.Children)[2].(RotateTra
nsform.Angle)">
                <SplineDoubleKeyFrame KeyTime="00:00:05" Value="360"/>
        </DoubleAnimationUsingKeyFrames>
</Storyboard>
```

We will address the storyboard from code behind and step trough it to adjust some properties.

- Open your project in Visual Studio
- Double click the button to add some code for the click event.
- Insert the following code:

```
private void btnStartAnimation_Click(object sender, RoutedEventArgs e)
{
    // address the storyboard by his name
    Storyboard sb = (Storyboard)this.FindResource("RectangleAnimation");

    // search for the two DoubleAnimationUsingKeyFrames elements in the storyboard
    DoubleAnimationUsingKeyFrames translate = (DoubleAnimationUsingKeyFrames)sb.Children[0];
    DoubleAnimationUsingKeyFrames rotate = (DoubleAnimationUsingKeyFrames)sb.Children[1];

    // search for the two SplineDoubleKeyFrame elements in the storyboard
    SplineDoubleKeyFrame translateFrame = (SplineDoubleKeyFrame)translate.KeyFrames[0];
    SplineDoubleKeyFrame rotateFrame = (SplineDoubleKeyFrame)rotate.KeyFrames[0];

    //Duration: adjust the property KeyTime to the value of the slider sldDuration
    translateFrame.KeyTime = new TimeSpan(0, 0, (int)sldDuration.Value);
    rotateFrame.KeyTime = new TimeSpan(0, 0, (int)sldDuration.Value);

    //Rotation: adjust the property Value of the rotateFrame to the value of the slider
    sldRotation
    rotateFrame.Value = sldRotation.Value;

    //Distance: adjust the property Value of the translateFrame to the value of the slider
    sldDistance
    translateFrame.Value = sldDistance.Value;

    //Finally start the animation
    sb.Begin(this);
}
```

- Test your project and you could control your animation with the sliders.