Course 2D_SL: 2D-Computer Graphics with Silverlight
Chapter C2: The Complete Code of the Draw Project

Install 1) **Visual Web Developer 2008 Express Edition with Service Pack 1 English**
and 2) **Silverlight Tools for Visual Studio 2008 SP1**.

Preliminaries

Guidance for **Visual Web Developer 2008 Express**:

1) Main Menu after start of VWD Express: `Tools` → `Options` →
check lower left checkbox: `Show all Settings` →
→ `Projects and Solutions` → `Projects location:` → `C:\temp`.
→ `Text Editor` (double click) → `All Languages` (double click) → `Tabs` →
`Indenting: None` → `Tab size: 2` → `Insert spaces`.
→ `Text Editor` (double click) → `C#` (double click) → `Formatting` →
uncheck all three check boxes → `OK`.

2) Main Menu after start of VWD Express: `File` → `New Project...` → `Project types: Visual C#` (double click) → `Silverlight` → `Templates: Silverlight Application`
`Name: SL_draw1` → `Location: C:\temp\SLProjects` → `Create directory for solution:` switch off → `OK`.
An `Add Silverlight Application`-Window appears.
Choose "`Automatically generate a test page to host Silverlight at build time`" → `OK`.

In the `Solution Explorer - SL_draw1` click the branch `Page.xaml`.
A split window will appear containing the default xaml code.
Delete this code completely and replace it by the code of **Page.xaml** which follows.

Page.xaml

```
<UserControl x:Class="SL_draw1.Page"
 xmlns="http://schemas.microsoft.com/client/2007"
 xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
 <StackPanel Orientation="Vertical">
  <TextBlock x:Name="text_area" HorizontalAlignment="Center"
                    VerticalAlignment="Top" FontFamily="Courier" FontSize="12"/>
  <Canvas x:Name="draw_area" Background="Cornsilk" Width="400" Height="400"
   MouseLeftButtonDown="OnMouseLeftButtonDown"
   MouseMove="OnMouseMove"
   MouseLeftButtonUp="OnMouseLeftButtonUp">
   <Polyline x:Name="pp" Stroke="#FF000000">
    <Polyline.RenderTransform>
     <TransformGroup>
      <RotateTransform x:Name="PolylineRotateTransform"
                       CenterX="0" CenterY="0" Angle="0"/>
      <ScaleTransform  x:Name="PolylineScaleTransform"
                       CenterX="0" CenterY="0" ScaleX="1" ScaleY="1"/>
     </TransformGroup>
    </Polyline.RenderTransform>
```

```
    <Polyline.Resources>
     <Storyboard x:Name="PolylineAnimationStoryboard">
      <DoubleAnimation
       Storyboard.TargetName="PolylineRotateTransform"
       Storyboard.TargetProperty="RenderTransform.Angle"
       From="0" To="360" Duration="0:0:2" AutoReverse="True"/>
      <DoubleAnimation
       Storyboard.TargetName="PolylineScaleTransform"
       Storyboard.TargetProperty="RenderTransform.ScaleX"
       From="1" To="0.1" Duration="0:0:2" AutoReverse="True"/>
      <DoubleAnimation
       Storyboard.TargetName="PolylineScaleTransform"
       Storyboard.TargetProperty="RenderTransform.ScaleY"
       From="1" To="0.1" Duration="0:0:2" AutoReverse="True"/>
     </Storyboard>
    </Polyline.Resources>
   </Polyline>
  </Canvas>
  <StackPanel x:Name="two_buttons" Orientation="Horizontal"
              HorizontalAlignment="Center" VerticalAlignment="Bottom">
   <Button x:Name="close_polyline_button" Click="OnClosePolylineClick"/>
   <Button x:Name="animation_button"      Click="OnAnimationClick"    />
  </StackPanel><!-- horizontal: close_polyline_button + animation_button-->
 </StackPanel> <!-- vertical: text_area + draw_area + 2 buttons-->
</UserControl>
```

Page.xaml.cs

```
using System;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Input;
using System.Windows.Shapes;

namespace SL_draw1
{ public partial class Page : UserControl
  { Point p0  = new Point();
    Point p1  = new Point();
    Point mid = new Point();
    double length;
    bool MouseButtonPressed = false;
    public Page()
    { InitializeComponent();
      draw_area.Width = draw_area.Height = two_buttons.Width = 400;
      close_polyline_button.Width = animation_button.Width = two_buttons.Width/2;
      text_area.Text="Press the left mouse button and draw something!";
    }
    private void OnMouseLeftButtonDown(object sender, MouseEventArgs e)
    { MouseButtonPressed = true;
      p0 = e.GetPosition( draw_area );
      pp.Points.Clear();
      pp.Points.Add( p0 );
      text_area.Text = String.Format( "({0:D}/{1:D})", (int)p0.X, (int)p0.Y );
    }
    private void OnMouseMove(object sender, MouseEventArgs e)
    { if ( !MouseButtonPressed ) return;
      p1 = e.GetPosition( draw_area );
      double dx = p1.X - p0.X;
      double dy = p1.Y - p0.Y;
      if ( dx*dx+dy*dy < 100 ) return;
      pp.Points.Add( p1 );
      p0 = p1;
      text_area.Text = String.Format( "(x={0:D}/y={1:D})", (int)p0.X, (int)p0.Y );
    }
```

```
    private void OnMouseLeftButtonUp(object sender, MouseEventArgs e)
    { MouseButtonPressed = false;
      mid.X = mid.Y = 0;
      for ( int i=0; i < pp.Points.Count; i++ )
      { mid.X += pp.Points[i].X;
        mid.Y += pp.Points[i].Y;
      }
      mid.X /= pp.Points.Count;
      mid.Y /= pp.Points.Count;
      length = 0;
      for ( int i=0; i < pp.Points.Count-1; i++ )
      { double dx = pp.Points[i+1].X - pp.Points[i].X;
        double dy = pp.Points[i+1].Y - pp.Points[i].Y;
        length += Math.Sqrt( dx*dx + dy*dy );
      }
      text_area.Text = "No. of Vertices = " + pp.Points.Count.ToString() +
                       "   Center of Gravity: " +
                       String.Format( "({0:F0}/{1:F0})", mid.X, mid.Y ) +
                       "   Line Length: " +
                       String.Format( "{0:F1}", length );
      close_polyline_button.Content = "Close the polygon";
      animation_button.Content = "Start the animation";
    }

    private void OnAnimationClick(object sender, EventArgs e)
    { PolylineRotateTransform.CenterX = PolylineScaleTransform.CenterX = mid.X;
      PolylineRotateTransform.CenterY = PolylineScaleTransform.CenterY = mid.Y;
      PolylineAnimationStoryboard.Stop();
      PolylineAnimationStoryboard.Begin();
    }

    private void OnClosePolylineClick(object sender, EventArgs e)
    { p0 = pp.Points[0];
      p1 = pp.Points[pp.Points.Count-1];
      if ( p1 == p0 ) return;
      double dx = p1.X - p0.X;
      double dy = p1.Y - p0.Y;
      length += Math.Sqrt( dx*dx + dy*dy );
      pp.Points.Add( p0 );
      text_area.Text = "No. of Vertices = " + pp.Points.Count.ToString() +
                       "   Center of Gravity: " +
                       String.Format( "({0:F0}/{1:F0})", mid.X, mid.Y ) +
                       "   Perimeter: " +
                       String.Format( "{0:F1}", length );
      close_polyline_button.Content = null;
    }
  } //end of class Page

}   //end of namespace
```