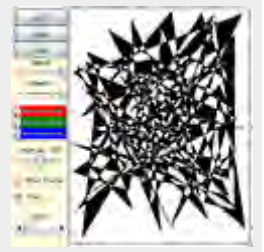
Course 2D_SL: 2D-Computer Graphics with Silverlight
Chapter C1: The Complete Code of the Intro Project



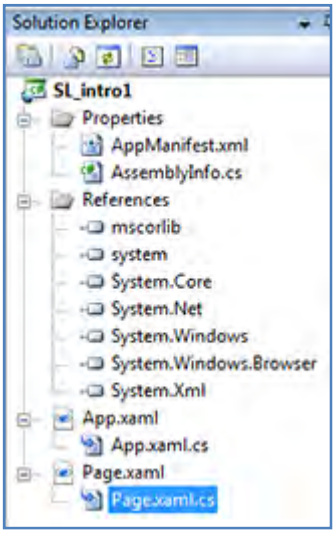
Install 1) **Visual Web Developer 2008 Express Edition with Service Pack 1 English**
and 2) **Silverlight Tools for Visual Studio 2008 SP1**.

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_intro1` → `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`.
The `Solution Explorer - SL_intro1` window appears
as shown on the right.



Replace the default code of `Page.xaml` and `Page.xaml.cs` by
the codes of **Page.xaml** and **Page.xaml.cs** which follow.

Page.xaml

```
<UserControl x:Class="SL_intro1.Page"
 xmlns="http://schemas.microsoft.com/client/2007"
 xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
 Width="380" Height="380" x:Name="user_control">
 <Grid Background="Cornsilk">
  <Grid.ColumnDefinitions>
   <ColumnDefinition x:Name="left_column" Width="90"/>
   <ColumnDefinition Width="Auto"/>
  </Grid.ColumnDefinitions>
  <Button x:Name="draw_area" Grid.Column="1" Background="White"
          Padding="0,0,0,0"  Loaded="on_draw_area_loaded">
   <Polygon x:Name="polygon" Stroke="Black" Fill="Black"/>
  </Button>
  <StackPanel Grid.Column="0" Orientation="Vertical" Margin="3">
   <Button x:Name="start_button" Content="Start" Click="on_start_button_click"/>
   <Button x:Name="stop_button"  Content="Stop"  Click="on_stop_button_click"/>
   <Button x:Name="clear_button" Content="Clear" Click="on_clear_button_click"/>
   <TextBlock Text="Speed" FontSize="10" HorizontalAlignment="Center"/>
   <Slider x:Name="time_interval_slider" Minimum="0" Maximum="1000" Value="0"
           ValueChanged="on_time_interval_slider_value_changed"
           IsDirectionReversed="True"/>
   <TextBlock Text="Opacity" FontSize="10" HorizontalAlignment="Center" />
   <Slider x:Name="opacity_slider" Minimum="0" Maximum="1" Value="1"
           ValueChanged="on_opacity_slider_value_changed" />
   <StackPanel Height="10" /><!--empty space-->
   <StackPanel Background="red" >
    <Slider x:Name="red_slider" Minimum="0" Maximum="255" Value="0"
            ValueChanged="on_color_slider_value_changed" />
   </StackPanel>
   <StackPanel Background="green" >
    <Slider x:Name="green_slider" Minimum="0" Maximum="255" Value="0"
            ValueChanged="on_color_slider_value_changed" />
   </StackPanel>
   <StackPanel Background="blue" >
    <Slider x:Name="blue_slider" Minimum="0" Maximum="255" Value="0"
            ValueChanged="on_color_slider_value_changed" />
   </StackPanel>
   <StackPanel Height="10" /><!--empty space-->
   <TextBlock x:Name="textblock_polygon_points" Text="Vertices: 100"
                      FontSize="10" HorizontalAlignment="Center" />
   <Slider x:Name="no_of_polygon_points_slider"
           Minimum="3" Maximum="200" Value="100"
           ValueChanged="on_no_of_polygon_points_slider_value_changed" />
   <StackPanel Height="10" /><!--empty space-->
   <RadioButton Content="Wire Frame"
                Click="on_radio_button_fill_mode_wire_frame_click" />
   <RadioButton Content="Filled" IsChecked="True"
                Click="on_radio_button_fill_mode_filled_click" />
  </StackPanel>
 </Grid>
</UserControl>
```

Page.xaml.cs

```
using System;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Shapes;
using System.Windows.Threading;
using System.Windows.Media;

namespace SL_intro1 {
public partial class Page : UserControl
{ Random r = new Random();
  int no_of_polygon_points = 100;
  DispatcherTimer timer = new DispatcherTimer();
  double w, h;
  public Page()
  { InitializeComponent();
    timer.Tick += new EventHandler(on_timer);
  }

  private void on_draw_area_loaded(object sender, EventArgs e)
  { draw_area.Width  = user_control.Width - 90;
    draw_area.Height = user_control.Height;
    w = draw_area.Width  - draw_area.Padding.Left
                         - draw_area.Padding.Right;
    h = draw_area.Height - draw_area.Padding.Top
                         - draw_area.Padding.Bottom;
    for (int i = 0; i < no_of_polygon_points; i++)
        polygon.Points.Add(new Point(w * r.NextDouble(),
                                     h * r.NextDouble()));
  }

  private void on_start_button_click(object sender, EventArgs e)
  { timer.Start();
  }

  private void on_stop_button_click(object sender, EventArgs e)
  { timer.Stop();
  }

  private void on_clear_button_click(object sender, EventArgs e)
  { timer.Stop();
    draw_area.Content = null;
  }

  public void on_timer(object sender, EventArgs e)
  { int i = r.Next(no_of_polygon_points);
    polygon.Points.RemoveAt(i);
    polygon.Points.Insert(i, new Point(w * r.NextDouble(),
                                       h * r.NextDouble()));
    draw_area.Content = 0;
    draw_area.Content = polygon;
  }

  public void on_time_interval_slider_value_changed
              (object sender, EventArgs e)
  { int milliseconds = (int)time_interval_slider.Value;
    timer.Interval = new TimeSpan(0, 0, 0, 0, milliseconds);
  }

  public void on_opacity_slider_value_changed(object sender, EventArgs e)
  { try {
    polygon.Stroke.Opacity = opacity_slider.Value;
    polygon.Fill.Opacity = opacity_slider.Value;
    } catch { }
  }
```

```
  public void on_color_slider_value_changed (object sender, EventArgs e)
  { SolidColorBrush brush = (SolidColorBrush)polygon.Fill;
    Color c = brush.Color;
    switch (((Slider)sender).Name)
    { case "red_slider"  : c.R = (byte)red_slider.Value;   break;
      case "green_slider": c.G = (byte)green_slider.Value; break;
      case "blue_slider" : c.B = (byte)blue_slider.Value;  break;
    }
    brush = new SolidColorBrush(c);
    polygon.Stroke = polygon.Fill = brush;
    draw_area.Content = 0;
    draw_area.Content = polygon;
  }

  public void on_no_of_polygon_points_slider_value_changed
              (object sender, EventArgs e)
  { try {
    no_of_polygon_points = (int)no_of_polygon_points_slider.Value;
    textblock_polygon_points.Text = "Vertices: " +
                                    no_of_polygon_points.ToString();
    polygon.Points.Clear();
    w = draw_area.ActualWidth - 8;
    h = draw_area.ActualHeight - 8;
    for (int i = 0; i < no_of_polygon_points; i++)
        polygon.Points.Add(new Point(w * r.NextDouble(),
                                     h * r.NextDouble()));
    draw_area.Content = 0;
    draw_area.Content = polygon;
    } catch { }
  }

  public void on_radio_button_fill_mode_wire_frame_click
         (object sender, EventArgs e)
  { polygon.Fill = draw_area.Background;
    draw_area.Content = 0;
    draw_area.Content = polygon;
  }

  public void on_radio_button_fill_mode_filled_click
              (object sender, EventArgs e)
  { polygon.Fill = polygon.Stroke;
    draw_area.Content = 0;
    draw_area.Content = polygon;
  }
}
} //end of namespace
```