

Course 2D_WPF: 2D-Computer Graphics with C# + WPF

Chapter C1: The Complete Code of the Intro Project

Copyright © by V. Miszalok, last update: 02-02-2008

- ↓ [Preliminaries](#)
- ↓ [intro1.cs driven by a DispatcherTimer Object](#)
- ↓ [intro1.cs driven by a DoubleAnimation Object](#)

Preliminaries

Guidance for **Visual Studio 2008**:

1) Main Menu after start of VS 2008: Tools → Options → check lower left checkbox: Show all Settings → Projects and Solutions → Visual Studio projects location: → C:\temp

2) Main Menu after start of VS 2008: File → New Project... → Visual Studio installed templates: Empty project
Name: intro1 → Location: C:\temp → Create directory for solution: switch off → OK.

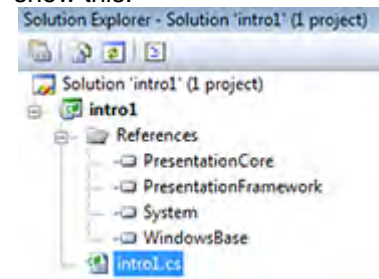
3) In the window titled: Solution Explorer -Solution 'intro1' (1 project) we have to add 4 References and an intro1.cs file:

3.1 **Right-click** the branch References. A drop-down menu appears. Click Add Reference... An Add Reference- window appears. Scroll down until You detect the Component Names Presentation Core and Presentation Framework and select them by Strg+click. Continue scrolling and Strg+click two more Component Names: System and WindowsBase. Quit the Add Reference- window with the button OK.

3.2 **Right-click** the branch **intro1**. A drop-down menu appears. Click Add and select New Item... An Add New Item - intro1- window appears. Select the template Code File and give it the Name: intro1.cs.

Quit the Add New Item - intro1- window with the button Add.

The Solution Explorer should show this:



4) Main menu of Visual Studio 2008 → Project → intro1 Properties... → Application → Output type: Change from Console Application to Windows Application.

5) Main menu of Visual Studio 2008 → Tools → Options... → An Options-window appears.

Double-click the branch Text Editor. **Double-click** C#. **Double-click** Formatting.

Click General. Uncheck all three check boxes.

Click Indentation. Uncheck all four check boxes.

Click New Lines. Uncheck all thirteen check boxes.

Click Spacing. Uncheck all twenty three check boxes.

Click IntelliSense. Uncheck all six check boxes.

Quit the Options- window with the OK-button.

intro1.cs driven by a DispatcherTimer Object

```

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

public class window1 : Window
{
    [STAThread] static void Main() { new Application().Run( new window1() ); }

    Canvas    myCanvas = new Canvas();
    Line      line1    = new Line();
    Line      line2    = new Line();
    Rectangle rect     = new Rectangle();
    Ellipse   elli     = new Ellipse();
    DockPanel myPanel  = new DockPanel();
    TextBox   left     = new TextBox();
    TextBox   top      = new TextBox();
    TextBox   right    = new TextBox();
    TextBox   bottom   = new TextBox();
    TextBox   central  = new TextBox();
    double    zoom     = 1.1;
    double    angle    = 0;
    Random    r        = new Random();
    Byte      r1, g1, b1, r2, g2, b2;

    public window1() //constructor
    {
        Top = Left = 50;
        this.Width = this.Height = 500;
        this.Title = "intro1";
        myCanvas.Children.Add( line1 );
        myCanvas.Children.Add( line2 );
        myCanvas.Children.Add( rect );
        myCanvas.Children.Add( elli );
        myCanvas.Children.Add( myPanel );
        myPanel.Children.Add( top ); DockPanel.SetDock( top, Dock.Top ); top.Text = "top";
        myPanel.Children.Add( bottom ); DockPanel.SetDock( bottom, Dock.Bottom ); bottom.Text = "bottom";
        myPanel.Children.Add( left ); DockPanel.SetDock( left, Dock.Left ); left.Text = "left";
        myPanel.Children.Add( right ); DockPanel.SetDock( right, Dock.Right ); right.Text = "right";
        myPanel.Children.Add( central );
        //Background = new SolidColorBrush( Colors.Cornsilk );
        Background = new LinearGradientBrush( Colors.Red, Colors.Blue, 90 );
        Foreground = new SolidColorBrush( Color.FromRgb( 0, 0, 200 ) );
        FontFamily = new FontFamily( "Courier New" );
        FontSize = 12;
        foreach( TextBox text in myPanel.Children )
        {
            text.HorizontalAlignment = HorizontalAlignment.Center;
            text.VerticalAlignment = VerticalAlignment.Center;
        }
        foreach( Object obj in myCanvas.Children )
        {
            if ( obj.GetType() == typeof(DockPanel) ) continue;
            ((Shape)obj).Stroke = Brushes.Black;
            ((Shape)obj).StrokeThickness = 5;
        }
        rect.Fill = Brushes.White;
        r1 = (Byte)r.Next(255); g1 = (Byte)r.Next(255); b1 = (Byte)r.Next(255);
        r2 = (Byte)r.Next(255); g2 = (Byte)r.Next(255); b2 = (Byte)r.Next(255);
        DispatcherTimer myTimer = new DispatcherTimer();
        myTimer.Interval = TimeSpan.FromMilliseconds( 40 );
        myTimer.Tick += TimerOnTick;
        myTimer.Start();
    }

    void TimerOnTick( Object sender, EventArgs args )
    {
        if ( myCanvas.ActualWidth < 200 ) zoom = 1.1;
        if ( myCanvas.ActualWidth > 800 ) zoom = 0.99;
        this.Width *= zoom;
        this.Height *= zoom;
        this.FontSize *= zoom;
    }
}

```

```

protected override void OnRenderSizeChanged( SizeChangedEventArgs sizeInfo )
{
    String s1 = "Hello World " + DateTime.Now.ToString() + "\n";
    int width = Convert.ToInt32( this.Width );
    int height = Convert.ToInt32( this.Height );
    String s2 = "Window Size = " + width.ToString() + " x " + height.ToString() + "\n";
    width = Convert.ToInt32( myCanvas.ActualWidth );
    height = Convert.ToInt32( myCanvas.ActualHeight );
    String s3 = "Client Size = " + width.ToString() + " x " + height.ToString() + "\n";
    String s4 = String.Format( "Font Size = {0,2:F1}", this.FontSize );
    central.Text = s1 + s2 + s3 + s4;
    line1.X1 = 0; line1.Y1 = 0; line1.X2 = myCanvas.ActualWidth; line1.Y2 = myCanvas.ActualHeight;
    line2.X1 = myCanvas.ActualWidth; line2.Y1 = 0; line2.X2 = 0; line2.Y2 = myCanvas.ActualHeight;
    Canvas.SetLeft( rect, myCanvas.ActualWidth / 5 );
    Canvas.SetLeft( elli, myCanvas.ActualWidth / 5 );
    Canvas.SetTop ( rect, myCanvas.ActualHeight / 5 );
    Canvas.SetTop ( elli, myCanvas.ActualHeight / 5 );
    rect.Width = elli.Width = 3 * myCanvas.ActualWidth / 5;
    rect.Height = elli.Height = 3 * myCanvas.ActualHeight / 5;
    Color color1 = Color.FromRgb( r1++, g1++, b1++ );
    Color color2 = Color.FromRgb( r2++, g2++, b2++ );
    RadialGradientBrush brush = new RadialGradientBrush( color1, color2 );
    brush.SpreadMethod = GradientSpreadMethod.Repeat;
    brush.RadiusX = brush.RadiusY = 0.1;
    brush.GradientOrigin = new Point( 0.5+0.05*Math.Cos(angle), 0.5+0.05*Math.Sin(angle) );
    elli.Fill = brush;
    angle += Math.PI / 32;
    myPanel.Width = myCanvas.ActualWidth;
    myPanel.Height = myCanvas.ActualHeight;
    Content = myCanvas;
}
}
}

```

intro1.cs driven by a DoubleAnimation Object

Main menu of Visual Studio 2008 → Project → intro1 Properties... → Application → Output type: Change from Windows Application to Console Application, otherwise the animation could never be stopped.

```

using System;
using System.Windows;
using System.Windows.Media;
using System.Windows.Controls;
using System.Windows.Shapes;
using System.Windows.Media.Animation;

public class window1 : Window
{
    [STAThread] static void Main() { new Application().Run( new window1() ); }
    Canvas myCanvas = new Canvas();
    Line line1 = new Line();
    Line line2 = new Line();
    Rectangle rect = new Rectangle();
    Ellipse elli = new Ellipse();
    DockPanel myPanel = new DockPanel();
    TextBox left = new TextBox();
    TextBox top = new TextBox();
    TextBox right = new TextBox();
    TextBox bottom = new TextBox();
    TextBox central = new TextBox();

    public window1() //constructor
    {
        Top = Left = 50;
        this.Width = this.Height = 500;
        this.Title = "intro1";
        myCanvas.Children.Add( line1 ); myCanvas.Children.Add( line2 );
        myCanvas.Children.Add( rect ); myCanvas.Children.Add( elli );
        myCanvas.Children.Add( myPanel );
        myPanel.Children.Add( top ); DockPanel.SetDock( top, Dock.Top ); top.Text = "top";
        myPanel.Children.Add( bottom ); DockPanel.SetDock( bottom, Dock.Bottom ); bottom.Text = "bottom";
        myPanel.Children.Add( left ); DockPanel.SetDock( left, Dock.Left ); left.Text = "left";
        myPanel.Children.Add( right ); DockPanel.SetDock( right, Dock.Right ); right.Text = "right";
        myPanel.Children.Add( central );
        //Background = new SolidColorBrush( Colors.Cornsilk );
        Background = new LinearGradientBrush( Colors.Red, Colors.Blue, 90 );
        Foreground = new SolidColorBrush( Color.FromRgb( 0, 0, 200 ) );
        FontFamily = new FontFamily( "Courier New" );
        FontSize = 12;
    }
}

```

```

foreach( TextBox text in myPanel.Children )
{ text.HorizontalAlignment = HorizontalAlignment.Center;
  text.VerticalAlignment = VerticalAlignment.Center;
}
foreach( Object obj in myCanvas.Children )
{ if ( obj.GetType() == typeof(DockPanel) ) continue;
  ((Shape)obj).Stroke = Brushes.Black;
  ((Shape)obj).StrokeThickness = 5;
}
rect.Fill = Brushes.White;

DoubleAnimation anim = new DoubleAnimation();
anim.Duration= new Duration( TimeSpan.FromSeconds( 3 ) );
anim.AutoReverse = true;
anim.RepeatBehavior = RepeatBehavior.Forever;
anim.From = 100;
anim.To = 600;
this.BeginAnimation( Window.WidthProperty, anim );
this.BeginAnimation( Window.HeightProperty, anim );
anim.From = 100 - 2* SystemParameters.ResizeFrameVerticalBorderWidth;
anim.To = 600 - 2* SystemParameters.ResizeFrameVerticalBorderWidth;
myCanvas.BeginAnimation( Canvas.WidthProperty, anim );
line1 .BeginAnimation( Line.X2Property, anim );
line2 .BeginAnimation( Line.X1Property, anim );
myPanel .BeginAnimation( Panel.WidthProperty, anim );
anim.From = 100 - 2* SystemParameters.ResizeFrameHorizontalBorderHeight
          - SystemParameters.CaptionHeight;
anim.To = 600 - 2* SystemParameters.ResizeFrameHorizontalBorderHeight
          - SystemParameters.CaptionHeight;
myCanvas.BeginAnimation( Canvas.HeightProperty, anim );
line1.BeginAnimation( Line.Y2Property, anim );
line2.BeginAnimation( Line.Y2Property, anim );
myPanel .BeginAnimation( Panel.HeightProperty, anim );
anim.From = ( 100 - 2* SystemParameters.ResizeFrameVerticalBorderWidth ) / 5;
anim.To = ( 600 - 2* SystemParameters.ResizeFrameVerticalBorderWidth ) / 5;
rect.BeginAnimation( Canvas.LeftProperty, anim );
elli.BeginAnimation( Canvas.LeftProperty, anim );
anim.From = ( 100 - 2* SystemParameters.ResizeFrameHorizontalBorderHeight
          - SystemParameters.CaptionHeight ) / 5;
anim.To = ( 600 - 2* SystemParameters.ResizeFrameHorizontalBorderHeight
          - SystemParameters.CaptionHeight ) / 5;
rect.BeginAnimation( Canvas.TopProperty, anim );
elli.BeginAnimation( Canvas.TopProperty, anim );
anim.From = 3 * ( 100 - 2* SystemParameters.ResizeFrameVerticalBorderWidth ) / 5;
anim.To = 3 * ( 600 - 2* SystemParameters.ResizeFrameVerticalBorderWidth ) / 5;
rect.BeginAnimation( Canvas.WidthProperty, anim );
elli.BeginAnimation( Canvas.WidthProperty, anim );
anim.From = 3 * ( 100 - 2* SystemParameters.ResizeFrameHorizontalBorderHeight
          - SystemParameters.CaptionHeight ) / 5;
anim.To = 3 * ( 600 - 2* SystemParameters.ResizeFrameHorizontalBorderHeight
          - SystemParameters.CaptionHeight ) / 5;
rect.BeginAnimation( Canvas.HeightProperty, anim );
elli.BeginAnimation( Canvas.HeightProperty, anim );
anim.From = 2;
anim.To = 20;
myCanvas.BeginAnimation( Window.FontSizeProperty, anim );
Content = myCanvas;
}

protected override void OnRenderSizeChanged( SizeChangedEventArgs sizeInfo )
{ String s1 = "Hello World " + DateTime.Now.ToString() + "\n";
  int width = Convert.ToInt32( this.Width );
  int height = Convert.ToInt32( this.Height );
  String s2 = "Window Size = " + width.ToString() + " x " + height.ToString() + "\n";
  width = Convert.ToInt32( myCanvas.ActualWidth );
  height = Convert.ToInt32( myCanvas.ActualHeight );
  String s3 = "Client Size = " + width.ToString() + " x " + height.ToString();
  central.Text = s1 + s2 + s3;
}
}

```