Windows Presentation Foundation

C# Programming

April 18

Windows Presentation Foundation

- WPF (code-named "Avalon") is the graphical subsystem of the .NET 3.0 Framework
- It provides a new unified way to develop richer UIs for both desktop and web applications
- Much richer than that `WinForms` – supports advanced 2-D and 3-D graphics, animation, audio, video, and more
- Provides a greater separation of UI and business logic

XAML

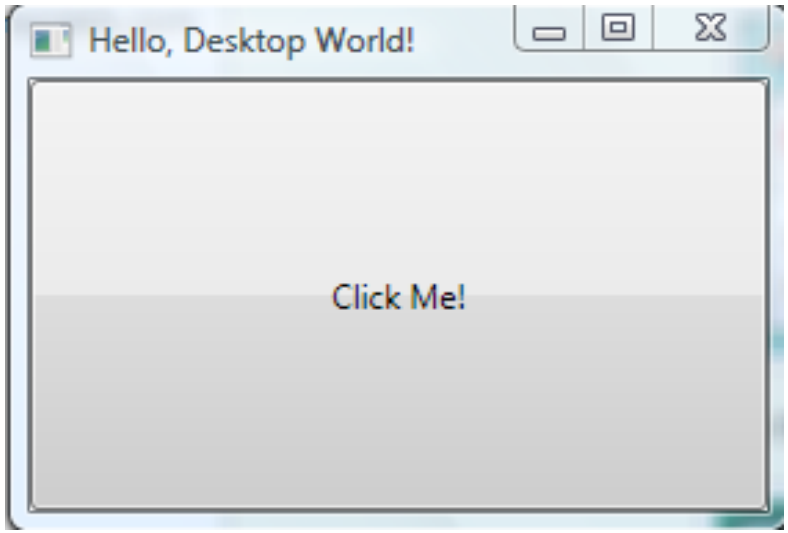
- Extensible Application Markup Language is used to describe the controls on the UI
- Controls on the application's `Window` or `Page` form a element tree hierarchy
- Set properties of controls in XAML
- Also set up event handlers that are defined in the `.cs` files behind the UI

Platforms / Tools

- Installed with Vista
- Can also be added on to XPSP2 and Server 2003
- There are many tools to help develop XAML UIs:
  - XamlPad
  - .NET 3.0 extension for Visual Studio 2005
  - Visual Studio "Orcas"
  - Microsoft Expression Blend (code-named "Sparkle")
- Desktop apps run standalone on .NET 3.0
- Web apps run within the browser with restricted security
- Microsoft Silverlight (formerly known as WPF Everywhere) is a lightweight subset of WPF for mobile devices

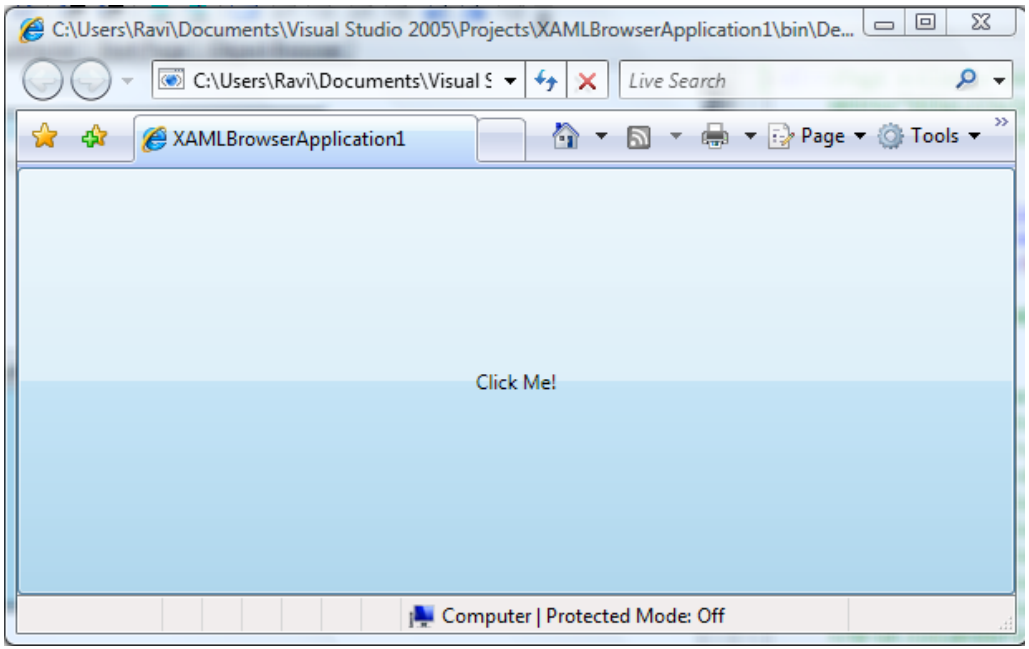
Hello, Desktop World

```
<Window
  xmlns="http://schemas.microsoft.com/winfx/2006/
          xaml/presentation"
  Title="Hello, Desktop World!">
    <Button>Click Me!</Button>
</Window>
```



Hello, Web World

```
<Page
   xmlns="http://schemas.microsoft.com/winfx/2006/
            xaml/presentation"
   Title="Hello, Web World!">
    <Button>Click Me!</Button>
</Page>
```



Setting Properties and Handlers

```
<Window x:Class="WindowsApplication8.Window1"
    xmlns="http://schemas.microsoft.com/winfx/2006/
              xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/
              xaml"
    Title="WindowsApplication8" Height="200" Width="200">
  <Button Background="Red"
          Height="100"
          Width="100"
          HorizontalAlignment="Center"
          Click="ClickHandler">Click Me</Button>
 </Window>
```

- `ClickHandler` is a method defined in the code-behind file `Window1.xaml.cs`

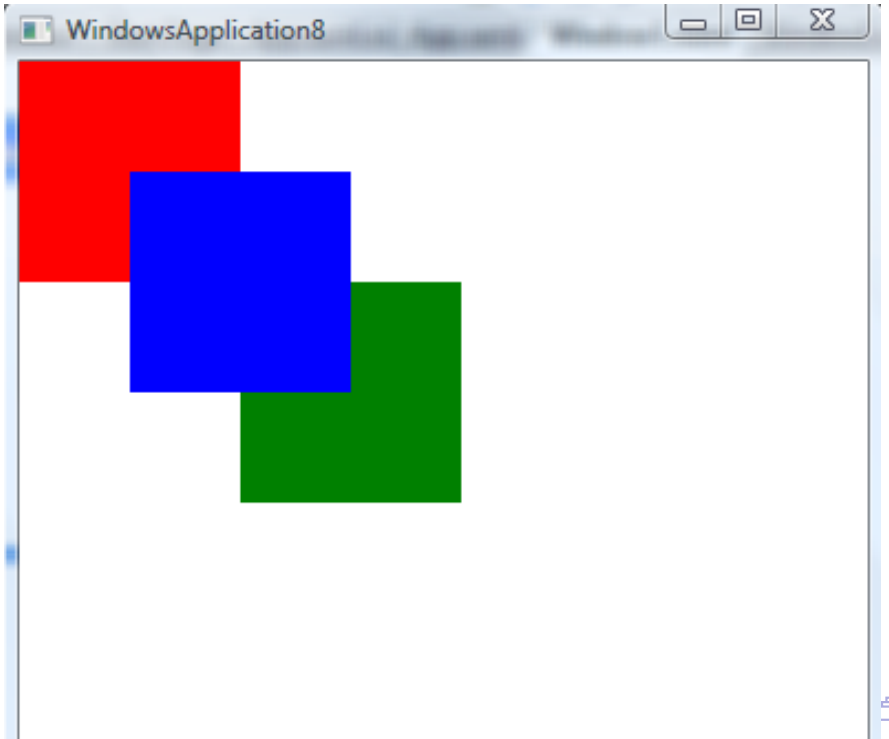
Layout

- The controls on a window or page are organized in an element tree
- There are different kinds of `Panels` for organizing controls in different ways
- By nesting panels, arbitrarily complex UIs can be organized

Canvas

- The most flexible panel is `Canvas`
- It behaves similar to `WinForms` in that controls are specified by absolute x- and y-position relative to the canvas
- Items that appear in last in the tree have the highest z-order

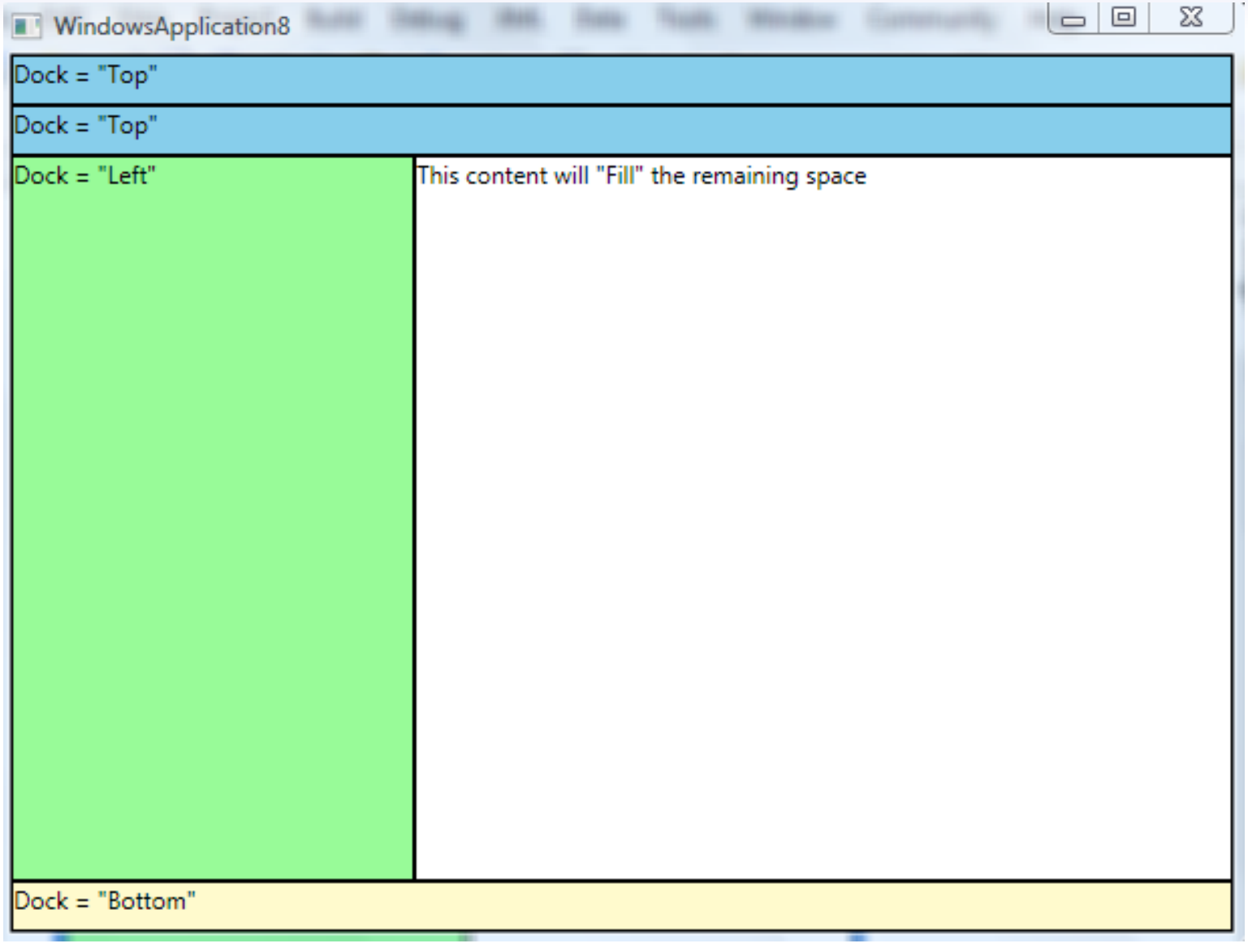
```
<Canvas>
    <Canvas Height="100" Width="100" Top="0"
       Left="0" Background="Red"/>
    <Canvas Height="100" Width="100" Top="100"
       Left="100" Background="Green"/>
    <Canvas Height="100" Width="100" Top="50"
       Left="50" Background="Blue"/>
</Canvas>
```



DockPanel

- `DockPanel` is a container that positions its elements along the edges of the container
- Elements can be position at the `Top`, `Bottom`, `Left`, or `Right`
- You can also have the last element added fill the remaining space of the container

```
<DockPanel LastChildFill="True">
  <Border Height="25" Background="SkyBlue" BorderBrush="Black"
   BorderThickness="1" DockPanel.Dock="Top">
    <TextBlock Foreground="Black">Dock = "Top"</TextBlock>
  </Border>
  <Border Height="25" Background="SkyBlue" BorderBrush="Black"
   BorderThickness="1" DockPanel.Dock="Top">
    <TextBlock Foreground="Black">Dock = "Top"</TextBlock>
  </Border>
  <Border Height="25" Background="LemonChiffon" BorderBrush="Black"
   BorderThickness="1" DockPanel.Dock="Bottom">
    <TextBlock Foreground="Black">Dock = "Bottom"</TextBlock>
  </Border>
  <Border Width="200" Background="PaleGreen" BorderBrush="Black"
   BorderThickness="1" DockPanel.Dock="Left">
    <TextBlock Foreground="Black">Dock = "Left"</TextBlock>
  </Border>
  <Border Background="White" BorderBrush="Black"
   BorderThickness="1">
    <TextBlock Foreground="Black">This content will "Fill" the remainin
 </Border>
</DockPanel>
```


WindowsApplication8
Dock = "Top"
Dock = "Top"
Dock = "Left"
This content will "Fill" the remaining space
Dock = "Bottom"

More Panels

- The other two major types of panels are `StackPanel` and `Grid`
- `StackPanels` position all their children from top to bottom or left to right
- `Grids` break up the display into rows and columns that contain elements
- You can use a `Grid` to specify that a particular row or column should occupy a fixed percentage of the window size

Resources

- For each element you can provide a resource dictionary
- This contains values that can be used within the scope of the element

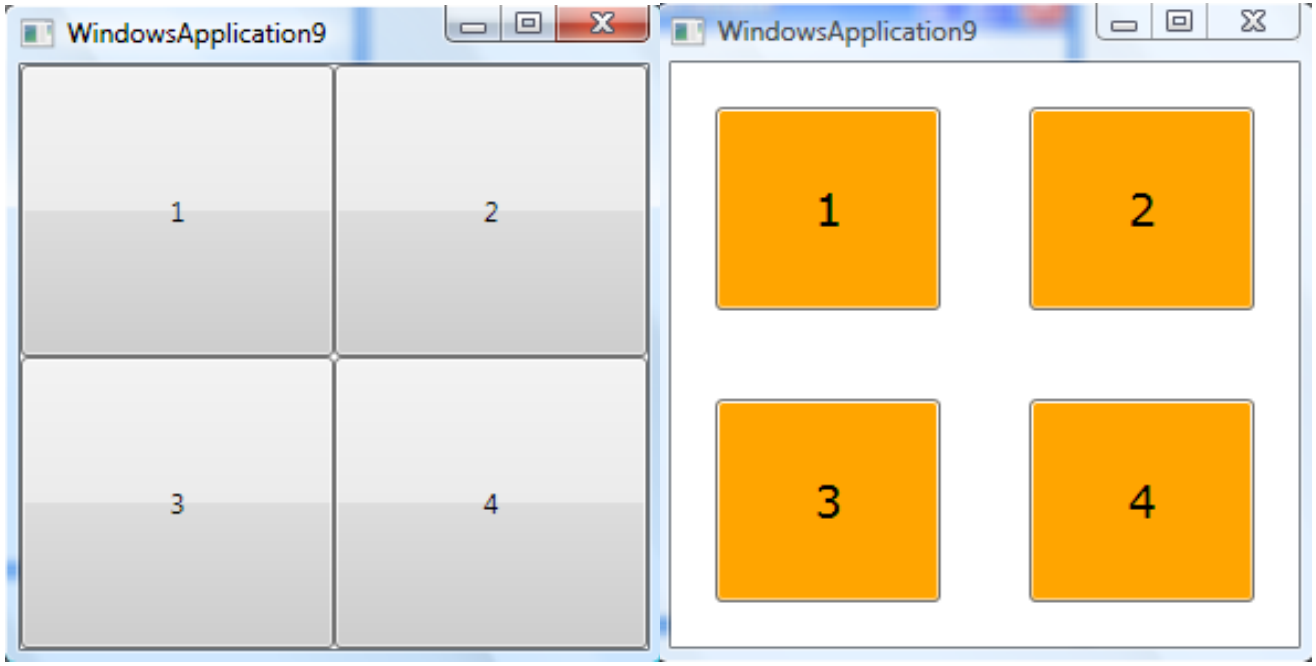
```
<Window>
    <Window.Resources>
        <Color x:Key="MyColor" R="10" G="3" B="100" />
    </Window.Resources>
    <Button Background="{StaticResource MyColor}" />
</Window>
```

Styles

- A very useful resource is a `Style`
- A `Style` can be used to set properties of several controls in a concise and organized way
- `Styles` also make it very easy to change the look of the UI, like CSS does for webpages

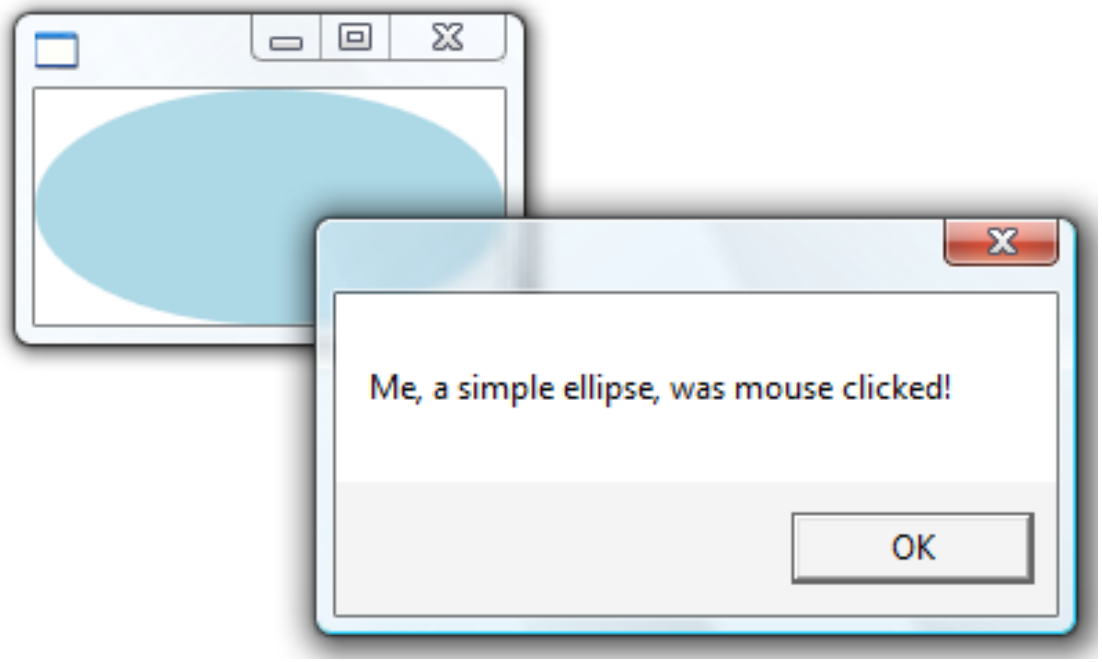
```
<Window xmlns="...">
  <Window.Resources>
    <Style TargetType="Button">
      <Setter Property="Background" Value="Orange" />
      <Setter Property="Margin" Value="10, 10, 10, 10" />
      <Setter Property="FontFamily" Value="Verdana" />
      <Setter Property="FontSize" Value="20" />
    </Style>
  </Window.Resources>
    <Grid>
      <Grid.RowDefinitions><RowDefinition/>
        <RowDefinition/></Grid.RowDefinitions>
      <Grid.ColumnDefinitions>
        <ColumnDefinition />
        <ColumnDefinition /></Grid.ColumnDefinitions>
      <Button Grid.Column="0" Grid.Row="0">1</Button>
      <Button Grid.Column="1" Grid.Row="0">2</Button>
      <Button Grid.Column="0" Grid.Row="1">3</Button>
      <Button Grid.Column="1" Grid.Row="1">4</Button>
    </Grid>
</Window>
```

Styles



2-D Graphics

- Common shapes are pre-defined elements
- Can define custom vector shapes
- These elements, like any other, can received keyboard and mouse input!
- ...this would have been nice for the Vector Graphics assignment :-)



3-D Graphics and Animation

[Samples]

Other Features

- Data binding
- Video and audio elements
- Fixed and flow documents

Bottom line: You know how to design GUIs using WinForms, but there are a lot of features to learn in WPF that help create better looking user interfaces!