Windows Presentation Foundation

Jim Fawcett

Software Modeling

ENGINEERING@SYRACUSE

References

- *Pro C# 5 and the .Net 4.5 Platform*, Andrew Troelsen, Apress, 2012
- *Programming WPF*, 2nd edition, Sells & Griffiths, O'Reilly, 2007
- *Windows Presentation Foundation Unleashed*, Adam Nathan, SAMS, 2007
- *Essential Windows Presentation Foundation*, Chris Anderson, Addison-Wesley, 2007
- http://msdn2.microsoft.com/en-us/library/aa970268.aspx
- http://msdn2.microsoft.com/en-us/library/ms754130.aspx

WPF Blogs

Josh Smith Blog

WPFpedia

Mike Taulty's Blog

Introduction

- What is WPF?
  - A graphical user interface technology
    - Desktop
    - Little brother Silverlight is used for web applications
  - Uses markup and code
    - Together or separately, much like ASP.Net
  - Easy to produce different styles
    - Web browser like navigation and placement
    - Traditional forms
    - Animated graphics

Markup

- XAML
  - eXtensible Application Markup Language
  - Tags are names of .Net 3.5 classes
  - Attributes are class properties and events

```
  <Grid>
   <Ellipse Fill=“blue” />
   <TextBlock>
      Name: <TextBlock Text=“{Binding Name}” />
   </TextBlock>
 </Grid>
```

Code Behind

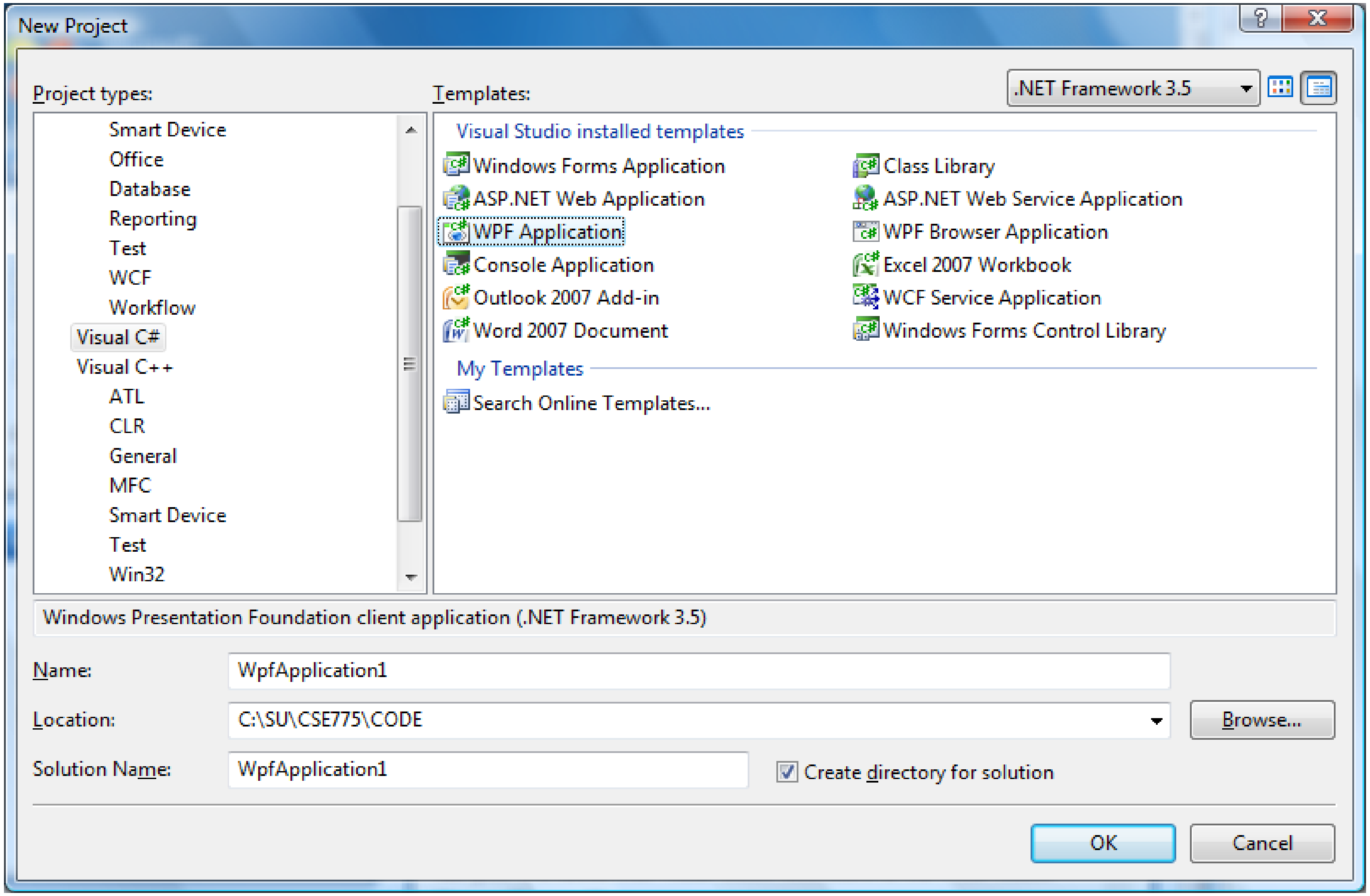
- Often, code provides processing for control events, bound in XAML, like this:
  - XAML in Window.Xaml

```
<Button
  x:Name=“button”
  Width=“200”
  Height=“25”
  Click=“button_Click”>Submit</Button>
```

  - C# code in Window.Xaml.cs

```
Void button_Click(object sender, RoutedEventsArgs e) {
    MessageBox.Show(…) }
```

C# Wizard


New Project

Project types:

- Smart Device
- Office
- Database
- Reporting
- Test
- WCF
- Workflow
- Visual C#
- Visual C++
- ATL
- CLR
- General
- MFC
- Smart Device
- Test
- Win32

Templates: .NET Framework 3.5

Visual Studio installed templates

Windows Forms Application	Class Library
ASP.NET Web Application	ASP.NET Web Service Application
WPF Application	WPF Browser Application
Console Application	Excel 2007 Workbook
Outlook 2007 Add-in	WCF Service Application
Word 2007 Document	Windows Forms Control Library

My Templates

Search Online Templates...

Windows Presentation Foundation client application (.NET Framework 3.5)

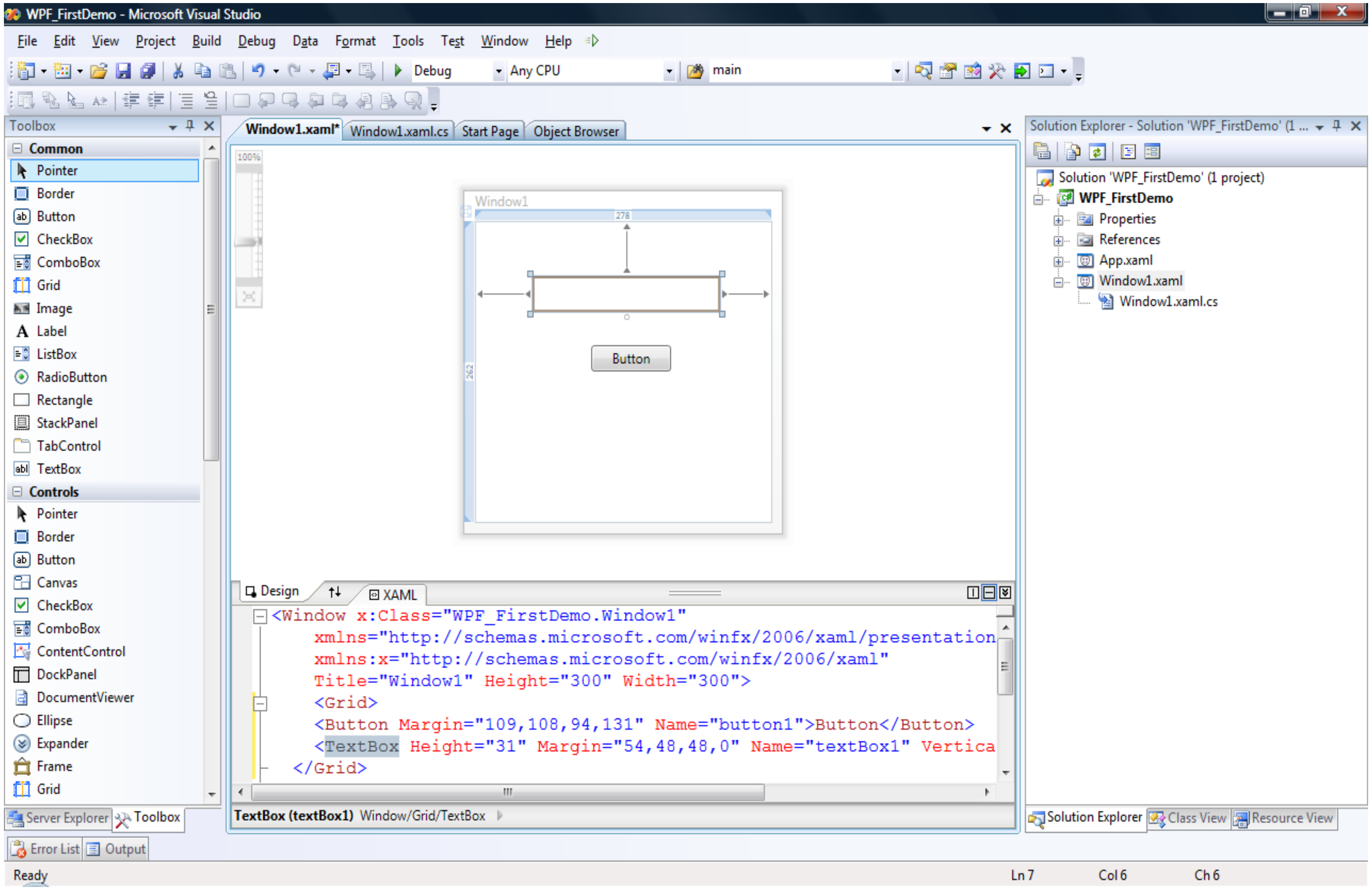
Name: WpfApplication1

Location: C:\SU\CSE775\CODE

Solution Name: WpfApplication1

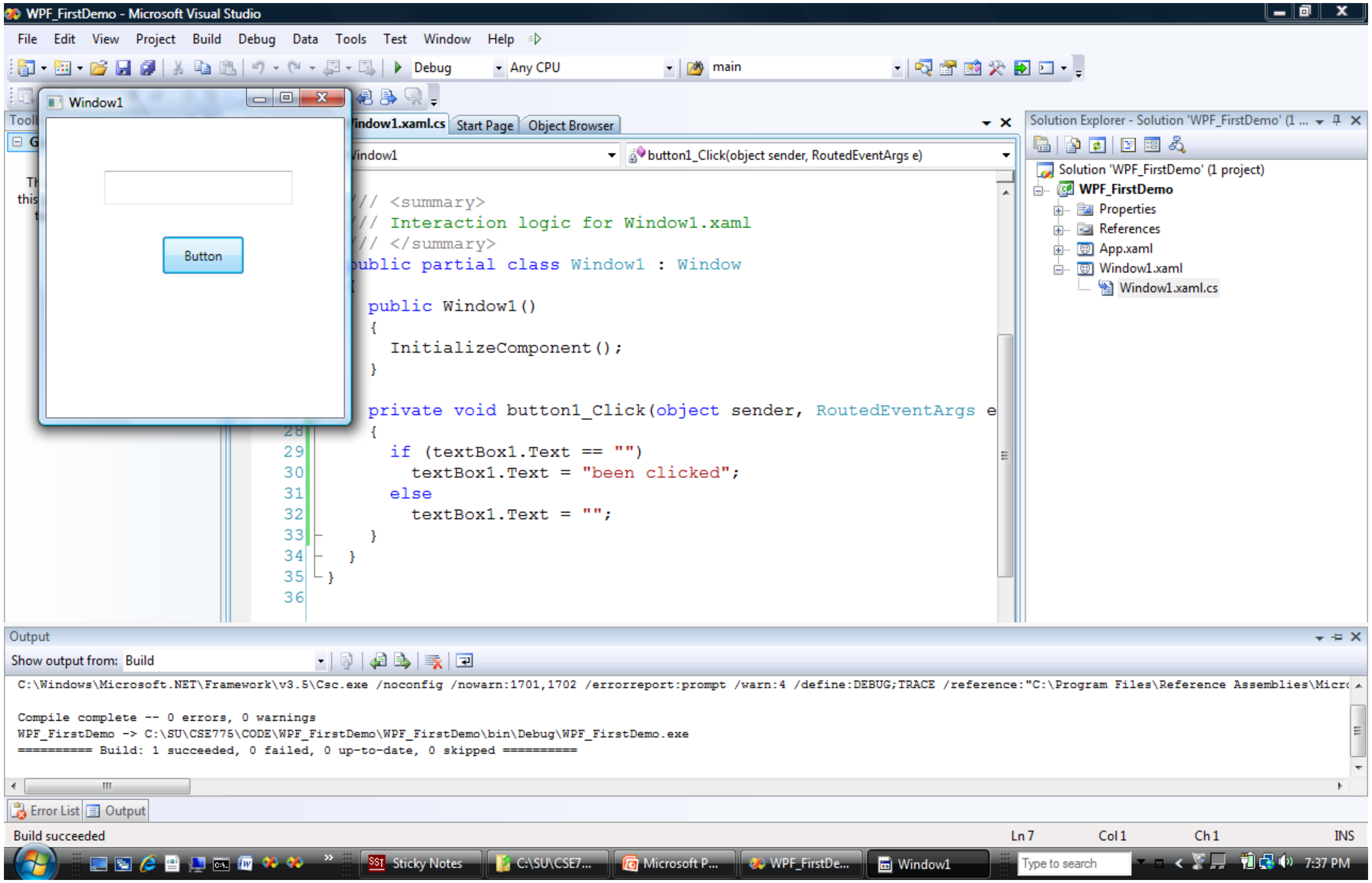
[✓] Create directory for solution

Default Grid Panel



```
<Window x:Class="WPF_FirstDemo.Window1"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Window1" Height="300" Width="300">
    <Grid>
    <Button Margin="109,108,94,131" Name="button1">Button</Button>
    <TextBox Height="31" Margin="54,48,48,0" Name="textBox1" Vertica
  </Grid>
```

Like WinForms, But …

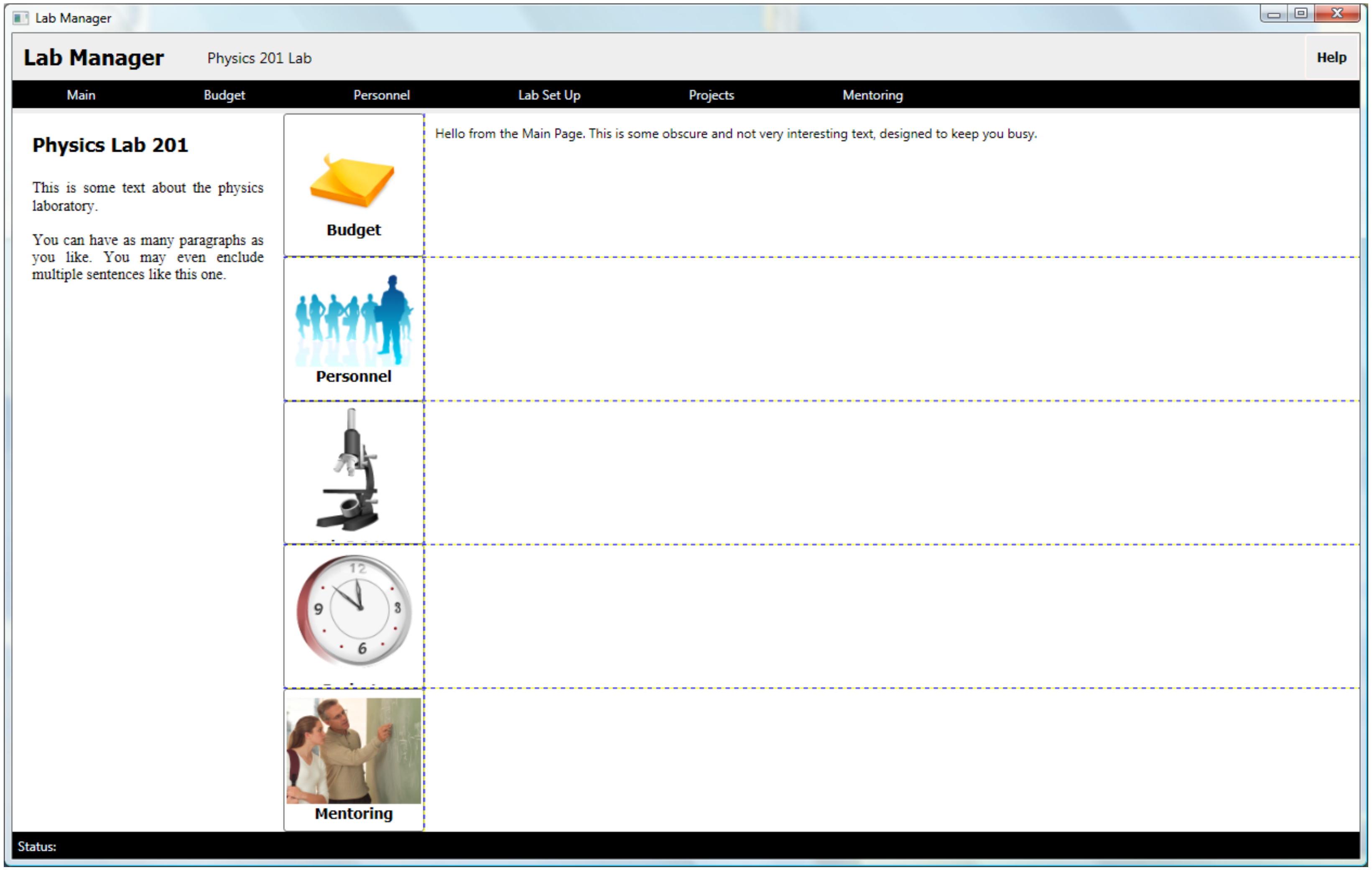


```
/// <summary>
/// Interaction logic for Window1.xaml
/// </summary>
ublic partial class Window1 : Window

  public Window1()
  {
    InitializeComponent();
  }

  private void button1_Click(object sender, RoutedEventArgs e
  {
    if (textBox1.Text == "")
      textBox1.Text = "been clicked";
    else
      textBox1.Text = "";
  }
 }
}
```

```
C:\Windows\Microsoft.NET\Framework\v3.5\Csc.exe /noconfig /nowarn:1701,1702 /errorreport:prompt /warn:4 /define:DEBUG;TRACE /reference:"C:\Program Files\Reference Assemblies\Micr

Compile complete -- 0 errors, 0 warnings
WPF_FirstDemo -> C:\SU\CSE775\CODE\WPF_FirstDemo\WPF_FirstDemo\bin\Debug\WPF_FirstDemo.exe
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

ENGINEERING@SYRACUSE

It’s Easy to Do More Interesting Things



Lab Manager

Lab Manager   Physics 201 Lab   Help

Physics Lab 201

This is some text about the physics laboratory.

You can have as many paragraphs as you like. You may even enclude multiple sentences like this one.

Hello from the Main Page. This is some obscure and not very interesting text, designed to keep you busy.

Status:

ENGINEERING@SYRACUSE

Panels

- Layouts, like the previous page can use:
  - Canvas
    - Simplest, placement relative to two edges
  - StackPanel
    - Horizontal or vertical stacking
  - Grid
    - Uses rows and columns
  - DockPanel
    - Dock to top, right, bottom, left, and all else fills remaining space
  - WrapPanel
    - Horizontal stacking with wrap on overflow
  - All of these can be nested, any one in another

Vector Graphics

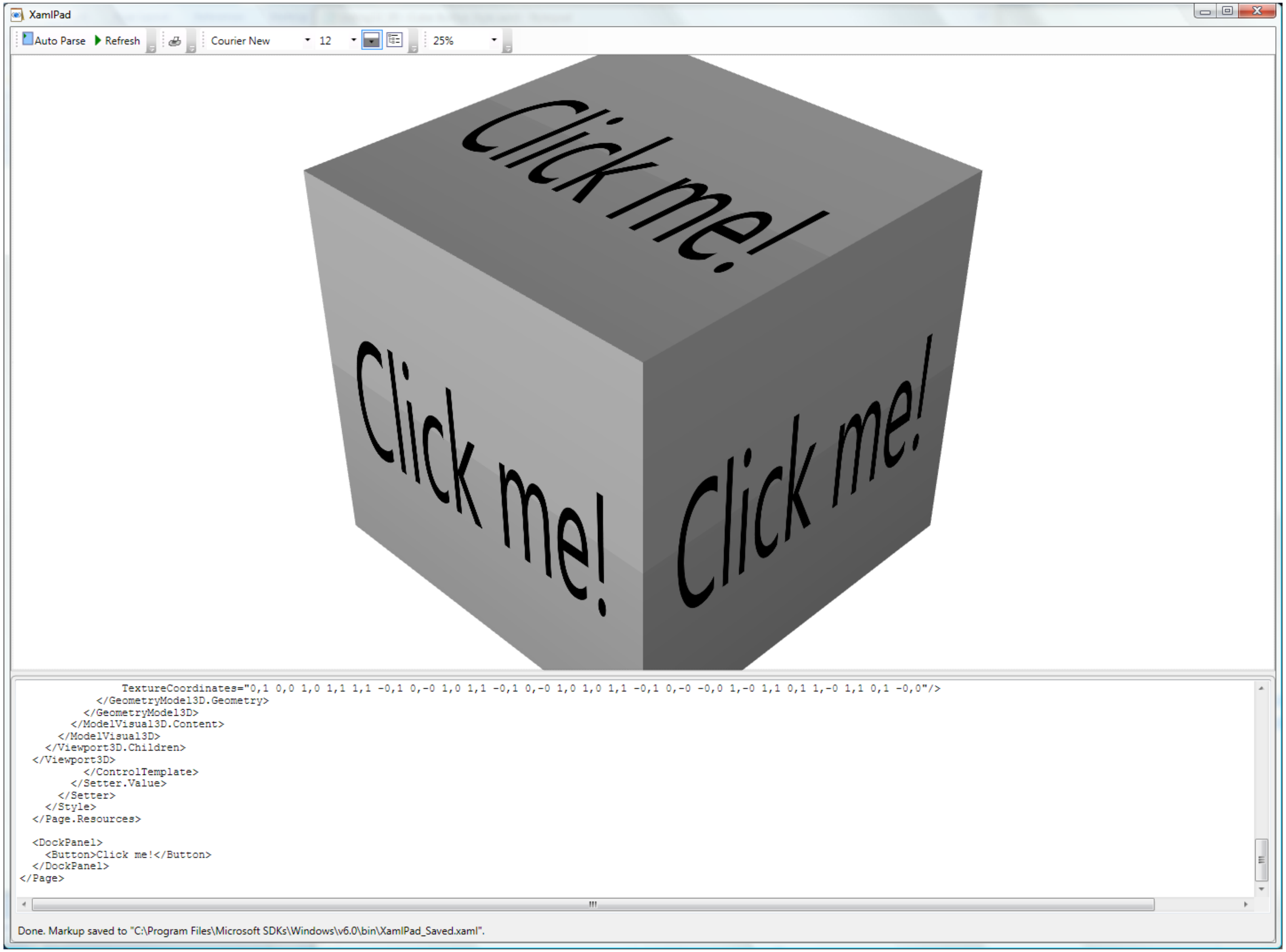
- In WPF there is only (usually) one window
  - Controls are not windows!
  - No handles—really, no handles
  - A button is a shape with border, fill, text, animation, and events, like click.
  - There is a Button class, but it is not a .Net control in the traditional sense nor an ActiveX control.
    - Just markup, lines, fills, and events.

Parse Tree

- XAML gets rendered into a parse tree, just like XML—it is XML
  - Inherited properties are based on parent-child relationships in the markup tree
  - Events bubble based on those relationships as well
  - You have direct and simple control over that structure
    - The world is yours!

What Makes WPF Unique?

- Vector graphics with parse-tree structure derived from markup
- Routed events bubble up the parse tree
- Pervasive publish-and-subscribe model
  - Data binding
  - Dependency properties
- Layered on top of DirectX
  - Strong 2D and 3D graphics
  - Animation
- Layout and styles model similar to the best of the web

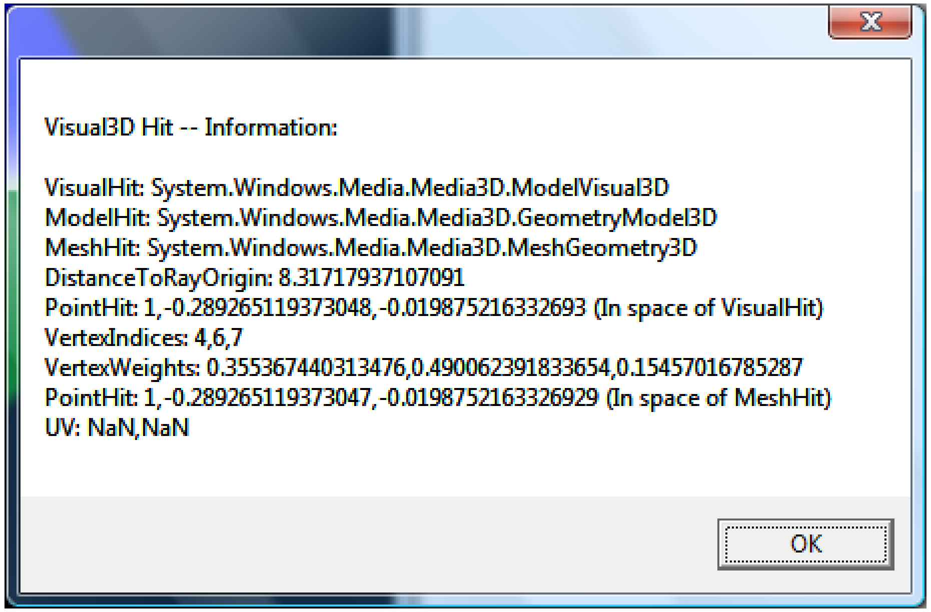


```
                TextureCoordinates="0,1 0,0 1,0 1,1 1,1 -0,1 0,-0 1,0 1,1 -0,1 0,-0 1,0 1,1 -0,1 0,-0 -0,0 1,-0 1,1 0,1 1,-0 1,1 0,1 -0,0"/>
            </GeometryModel3D.Geometry>
          </GeometryModel3D>
        </ModelVisual3D.Content>
      </ModelVisual3D>
    </Viewport3D.Children>
  </Viewport3D>
          </ControlTemplate>
        </Setter.Value>
      </Setter>
    </Style>
  </Page.Resources>

  <DockPanel>
    <Button>Click me!</Button>
  </DockPanel>
</Page>
```

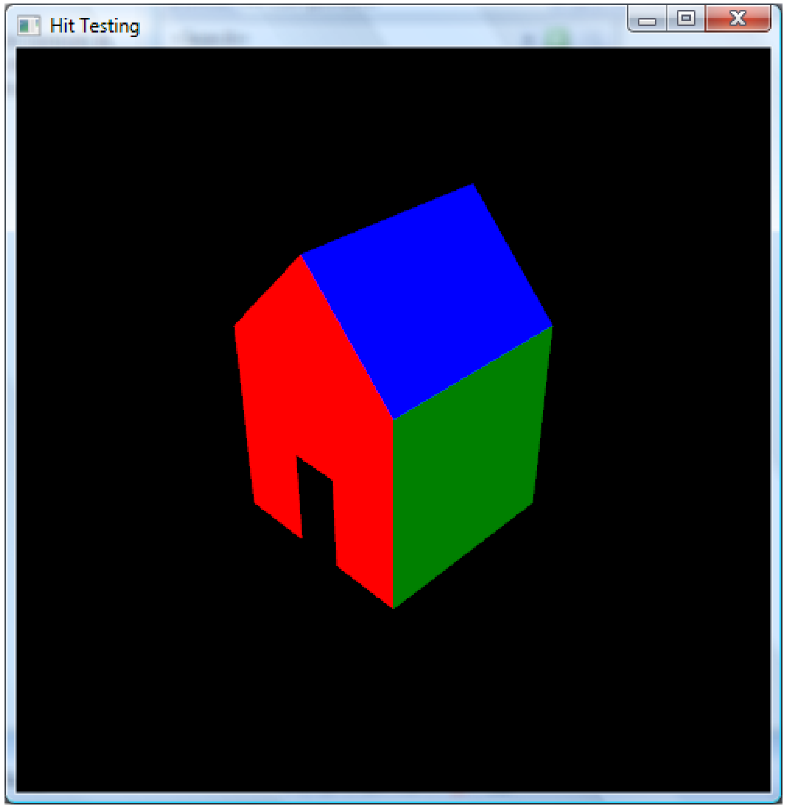
Done. Markup saved to "C:\Program Files\Microsoft SDKs\Windows\v6.0\bin\XamlPad_Saved.xaml".

ENGINEERING@SYRACUSE

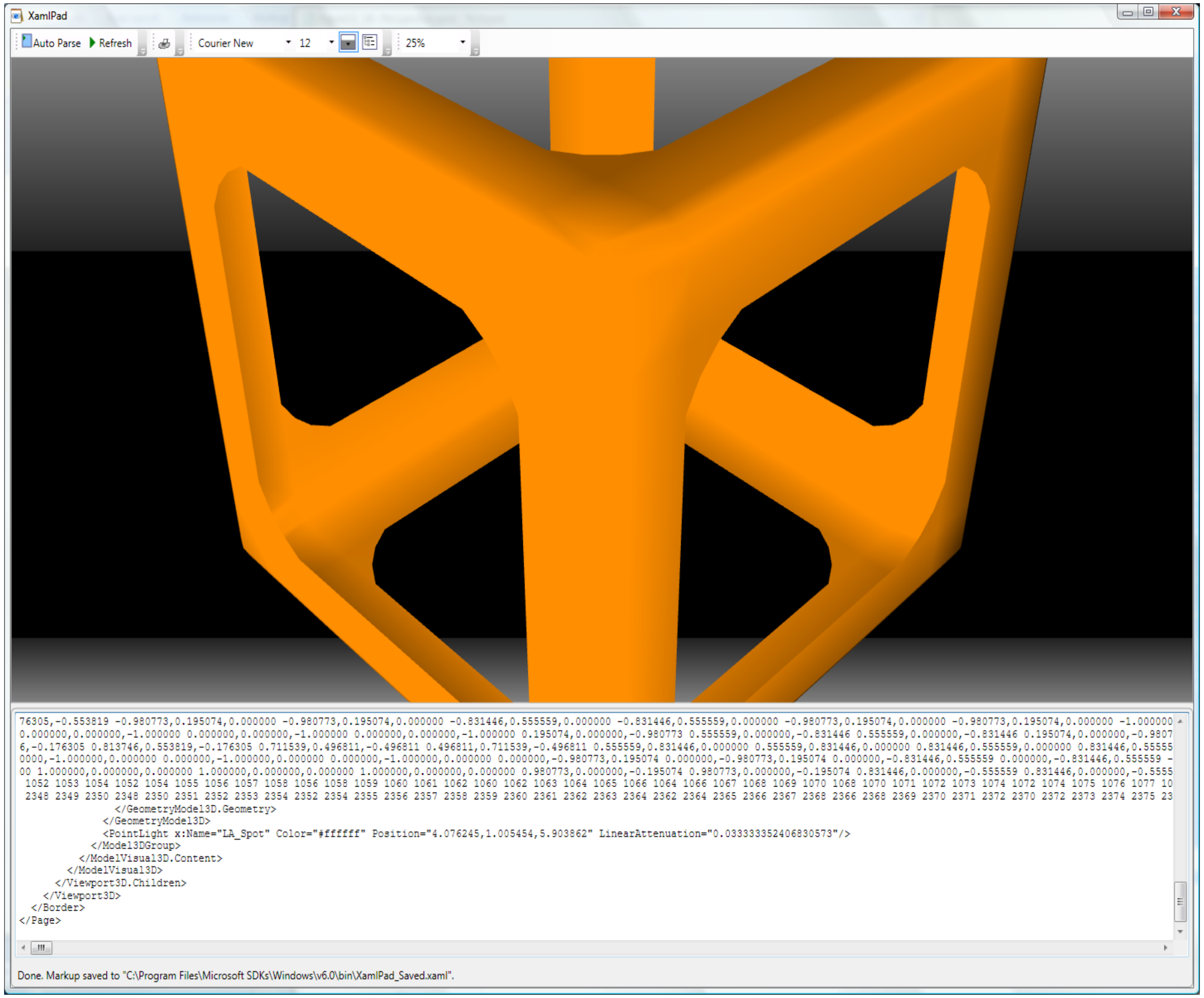
3D Hit Testing



```
Visual3D Hit -- Information:

VisualHit: System.Windows.Media.Media3D.ModelVisual3D
ModelHit: System.Windows.Media.Media3D.GeometryModel3D
MeshHit: System.Windows.Media.Media3D.MeshGeometry3D
DistanceToRayOrigin: 8.31717937107091
PointHit: 1,-0.289265119373048,-0.019875216332693 (In space of VisualHit)
VertexIndices: 4,6,7
VertexWeights: 0.355367440313476,0.490062391833654,0.15457016785287
PointHit: 1,-0.289265119373047,-0.0198752163326929 (In space of MeshHit)
UV: NaN,NaN
```



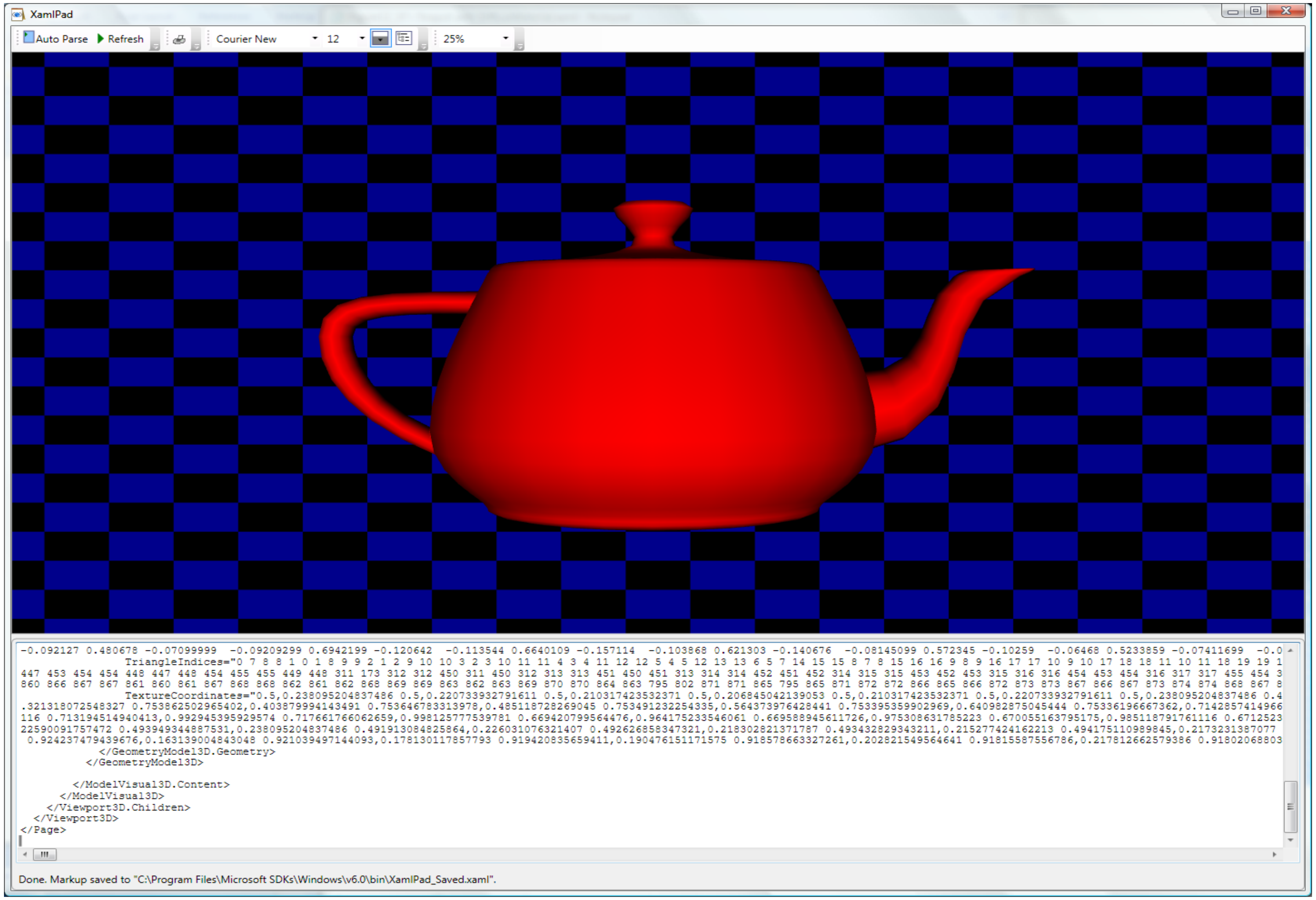
3D Perspective Camera



```
76305,-0.553819 -0.980773,0.195074,0.000000 -0.980773,0.195074,0.000000 -0.831446,0.555559,0.000000 -0.831446,0.555559,0.000000 -0.980773,0.195074,0.000000 -0.980773,0.195074,0.000000 -1.000000
0.000000,0.000000,-1.000000 0.000000,0.000000,-1.000000 0.000000,0.000000,-1.000000 0.195074,0.000000,-0.980773 0.555559,0.000000,-0.831446 0.555559,0.000000,-0.831446 0.195074,0.000000,-0.9807
6,-0.176305 0.813746,0.553819,-0.176305 0.711539,0.496811,-0.496811 0.496811,0.711539,-0.496811 0.555559,0.831446,0.000000 0.555559,0.831446,0.000000 0.831446,0.555559,0.000000 0.831446,0.55555
0000,-1.000000,0.000000 0.000000,-1.000000,0.000000 0.000000,-1.000000,0.000000 0.000000,-0.980773,0.195074 0.000000,-0.980773,0.195074 0.000000,-0.831446,0.555559 0.000000,-0.831446,0.555559 -
00 1.000000,0.000000,0.000000 1.000000,0.000000,0.000000 1.000000,0.000000,0.000000 0.980773,0.000000,-0.195074 0.980773,0.000000,-0.195074 0.831446,0.000000,-0.555559 0.831446,0.000000,-0.5555
 1052 1053 1054 1052 1054 1055 1056 1057 1058 1056 1058 1059 1060 1061 1062 1060 1062 1063 1064 1065 1066 1064 1066 1067 1068 1069 1070 1068 1070 1071 1072 1073 1074 1072 1074 1075 1076 1077 10
 2348 2349 2350 2348 2350 2351 2352 2353 2354 2352 2354 2355 2356 2357 2358 2359 2360 2361 2362 2363 2364 2362 2364 2365 2366 2367 2368 2366 2368 2369 2370 2371 2372 2370 2372 2373 2374 2375 23
                </GeometryModel3D.Geometry>
              </GeometryModel3D>
              <PointLight x:Name="LA_Spot" Color="#ffffff" Position="4.076245,1.005454,5.903862" LinearAttenuation="0.0333333352406830573"/>
            </Model3DGroup>
          </ModelVisual3D.Content>
        </ModelVisual3D>
      </Viewport3D.Children>
    </Viewport3D>
  </Border>
</Page>
```

Done. Markup saved to "C:\Program Files\Microsoft SDKs\Windows\v6.0\bin\XamlPad_Saved.xaml".

Famous Teapot



```
-0.092127 0.480678 -0.07099999  -0.09209299 0.6942199 -0.120642  -0.113544 0.6640109 -0.157114  -0.103868 0.621303 -0.140676  -0.08145099 0.572345 -0.10259  -0.06468 0.5233859 -0.07411699  -0.0
                TriangleIndices="0 7 8 8 1 0 1 8 9 9 2 1 2 9 10 10 3 2 3 10 11 11 4 3 4 11 12 12 5 4 5 12 13 13 6 5 7 14 15 15 8 7 8 15 16 16 9 8 9 16 17 17 10 9 10 17 18 18 11 10 11 18 19 19 1
447 453 454 454 448 447 448 454 455 455 449 448 311 173 312 312 450 311 450 312 313 313 451 450 451 313 314 314 452 451 452 314 315 315 453 452 453 315 316 316 454 453 454 316 317 317 455 454 3
860 866 867 867 861 860 861 867 868 868 862 861 862 868 869 869 863 862 863 869 870 870 864 863 795 802 871 871 865 795 865 871 872 872 866 865 866 872 873 873 867 866 867 873 874 874 868 867 8
                TextureCoordinates="0.5,0.238095204837486 0.5,0.220733932791611 0.5,0.210317423532371 0.5,0.206845042139053 0.5,0.210317423532371 0.5,0.220733932791611 0.5,0.238095204837486 0.4
.321318072548327 0.753862502965402,0.403879994143491 0.753646783313978,0.485118728269045 0.753491232254335,0.564373976428441 0.753395359902969,0.640982875045444 0.753361966667362,0.7142857414966
116 0.713194514940413,0.992945395929574 0.717661766062659,0.998125777539781 0.669420799564476,0.964175233546061 0.669588945611726,0.975308631785223 0.670055163795175,0.985118791761116 0.6712523
22590091757472 0.493949344887531,0.238095204837486 0.491913084825864,0.226031076321407 0.492626858347321,0.218302821371787 0.493432829343211,0.215277424162213 0.494175110989845,0.2173231387077
 0.924237479439676,0.163139004843048 0.921039497144093,0.178130117857793 0.919420835659411,0.190476151171575 0.918578663327261,0.202821549564641 0.91815587556786,0.217812662579386 0.91802068803
            </GeometryModel3D.Geometry>
          </GeometryModel3D>

        </ModelVisual3D.Content>
      </ModelVisual3D>
    </Viewport3D.Children>
  </Viewport3D>
</Page>
```

Done. Markup saved to "C:\Program Files\Microsoft SDKs\Windows\v6.0\bin\XamlPad_Saved.xaml".

ENGINEERING@SYRACUSE

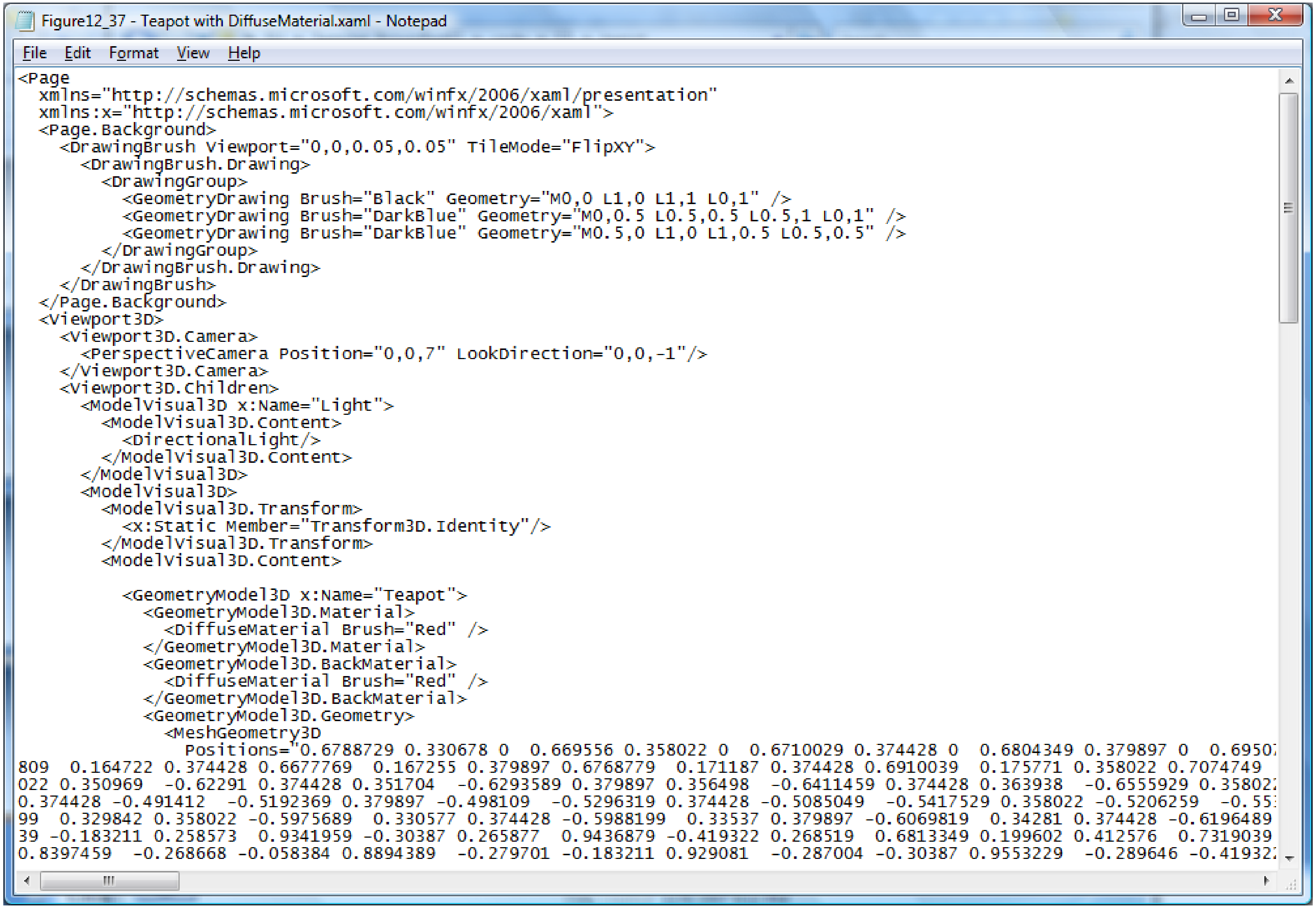


Figure12_37 - Teapot with DiffuseMaterial.xaml - Notepad

```
<Page
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
  <Page.Background>
    <DrawingBrush Viewport="0,0,0.05,0.05" TileMode="FlipXY">
      <DrawingBrush.Drawing>
        <DrawingGroup>
          <GeometryDrawing Brush="Black" Geometry="M0,0 L1,0 L1,1 L0,1" />
          <GeometryDrawing Brush="DarkBlue" Geometry="M0,0.5 L0.5,0.5 L0.5,1 L0,1" />
          <GeometryDrawing Brush="DarkBlue" Geometry="M0.5,0 L1,0 L1,0.5 L0.5,0.5" />
        </DrawingGroup>
      </DrawingBrush.Drawing>
    </DrawingBrush>
  </Page.Background>
  <Viewport3D>
    <Viewport3D.Camera>
      <PerspectiveCamera Position="0,0,7" LookDirection="0,0,-1"/>
    </Viewport3D.Camera>
    <Viewport3D.Children>
      <ModelVisual3D x:Name="Light">
        <ModelVisual3D.Content>
          <DirectionalLight/>
        </ModelVisual3D.Content>
      </ModelVisual3D>
      <ModelVisual3D>
        <ModelVisual3D.Transform>
          <x:Static Member="Transform3D.Identity"/>
        </ModelVisual3D.Transform>
        <ModelVisual3D.Content>

          <GeometryModel3D x:Name="Teapot">
            <GeometryModel3D.Material>
              <DiffuseMaterial Brush="Red" />
            </GeometryModel3D.Material>
            <GeometryModel3D.BackMaterial>
              <DiffuseMaterial Brush="Red" />
            </GeometryModel3D.BackMaterial>
            <GeometryModel3D.Geometry>
              <MeshGeometry3D
                Positions="0.6788729 0.330678 0  0.669556 0.358022 0  0.6710029 0.374428 0  0.6804349 0.379897 0  0.6950
809  0.164722 0.374428 0.6677769  0.167255 0.379897 0.6768779  0.171187 0.374428 0.6910039  0.175771 0.358022 0.7074749
022 0.350969  -0.62291 0.374428 0.351704  -0.6293589 0.379897 0.356498  -0.6411459 0.374428 0.363938  -0.6555929 0.35802
0.374428 -0.491412  -0.5192369 0.379897 -0.498109  -0.5296319 0.374428 -0.5085049  -0.5417529 0.358022 -0.5206259  -0.55
99  0.329842 0.358022 -0.5975689  0.330577 0.374428 -0.5988199  0.33537 0.379897 -0.6069819  0.34281 0.374428 -0.6196489
39 -0.183211 0.258573  0.9341959 -0.30387 0.265877  0.9436879 -0.419322 0.268519  0.6813349 0.199602 0.412576  0.7319039
0.8397459  -0.268668 -0.058384 0.8894389  -0.279701 -0.183211 0.929081  -0.287004 -0.30387 0.9553229  -0.289646 -0.41932
```

ENGINEERING@SYRACUSE

Routed Events

- WPF maps markup elements to UIElements, which derive from ContentControl
  - That means that almost everything can hold content—only one thing unless it’s a panel.
  - How does a mouse click ***event*** on any one of a control’s content elements get ***routed*** to the control?
    - By walking the XAML parse tree until it finds a parent that handles that event.

Adding Event Handlers

- You will find that property sheets show events as well as properties
  - Click on the lightning bolt to see the event sheet.
  - You subscribe by clicking on an event entry.
- You can also add event handlers quickly in XAML:
  - Go to the XAML, type a space after the tag for the element you want to handle the event
    - That gets you a context menu (via IntelliSense) and you just double-click on the desired event, which adds an event attribute

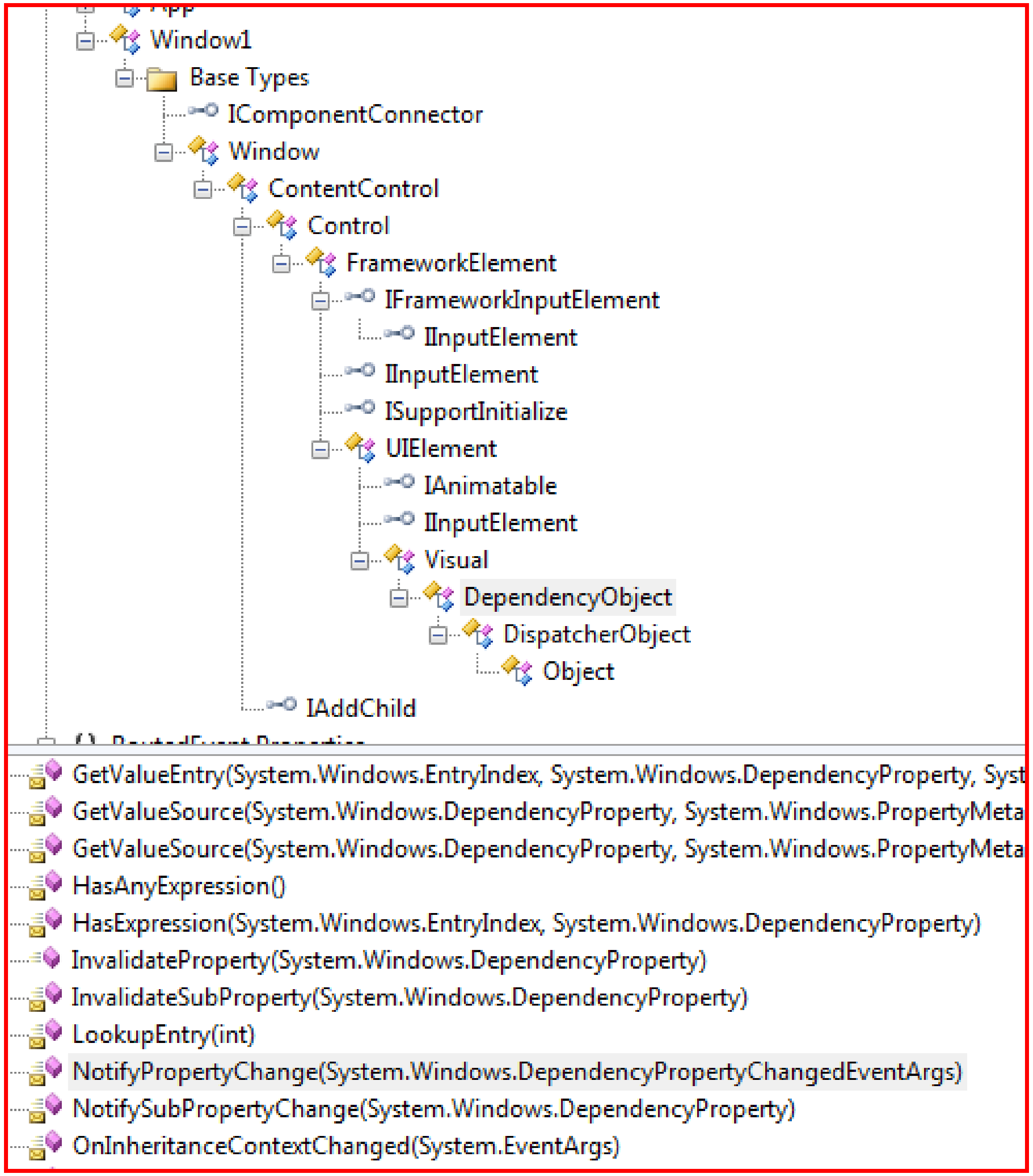
Attached Properties

- Buttons, ListBoxes, Images, etc., do not have Dock properties.
- However, when you place one of these in a DockPanel, you find that it has had Dock ***properties attached***.

<Image Source="./help.png" ***DockPanel.Dock="Top"*** Height="213" ImageFailed="Image_ImageFailed" />

DependencyObject Class

- Attached properties work because all WPF controls derive from the DependencyObject class.
  - DependencyObject class supports adding an arbitrary number of dependency properties.



```
App
Window1
  Base Types
    IComponentConnector
    Window
      ContentControl
        Control
          FrameworkElement
            IFrameworkInputElement
              IInputElement
            IInputElement
            ISupportInitialize
            UIElement
              IAnimatable
              IInputElement
              Visual
                DependencyObject
                  DispatcherObject
                    Object
        IAddChild
```

```
GetValueEntry(System.Windows.EntryIndex, System.Windows.DependencyProperty, Syst
GetValueSource(System.Windows.DependencyProperty, System.Windows.PropertyMeta
GetValueSource(System.Windows.DependencyProperty, System.Windows.PropertyMeta
HasAnyExpression()
HasExpression(System.Windows.EntryIndex, System.Windows.DependencyProperty)
InvalidateProperty(System.Windows.DependencyProperty)
InvalidateSubProperty(System.Windows.DependencyProperty)
LookupEntry(int)
NotifyPropertyChange(System.Windows.DependencyPropertyChangedEventArgs)
NotifySubPropertyChange(System.Windows.DependencyProperty)
OnInheritanceContextChanged(System.EventArgs)
```

Dependency Properties

- A dependency property is a property that is registered with the WPF dependency property system. Two uses:
  - Backing an object property with a dependency property, provides support for data binding, styling, and animation. Examples include Background and FontSize properties.
  - Creating attached properties. Attached properties are properties that can be set on ANY DependencyObject types. An example is the Dock property.
- You can find an example of the definition and use of a custom dependency property here.
- Dependency properties are a publish-and-subscribe system.

Dependency Property Links

Josh Smith's Blog

Switch on the Code Blog

Learn WPF site

Property Syntax

- Two syntax forms:
  - XAML attribute:

```
<button ToolTip=“Button Tip />
```

  - Property element syntax:

```
<Button>
  <Button.Background>
    <SolidColorBrush Color=“#FF4444FF” />
  </Button.Background>
  Some Button Text
</Button>
```

Markup Extensions

- Sometimes you need to assign a property from some source at run-time. For that you use markup extensions:

```
<Button Foreground="{x:static
          SystemColors.ActiveCaptionBrush}" >
     Some text
</Button>
```

Inline Styles

- Collections of property values:
  -
```
<Button.Style>
   <Style>
      <Setter Property=“Button.FontSize”
Value=“32pt” />
      <Setter Property=“Button.FontWeight”
Value=“Bold” />
   </Style>
</Button.Style>
```

Named Styles

- Collections of property values:

```
<Window.Resources>
   <Style x:Key=“myStyle” TargetType=“{x:Type
Control}”>
      <Setter Property=“FontSize” Value=“32pt” />
      <Setter Property=“FontWeight”
Value=“Bold” />
   </Style>
</Window>
```

Binding

- Binding infrastructure allows you to set up a one-way or two-way updating of property values that happens when the source changes.
- This requires two things:
  - A dependency object
    - Has its own dispatcher thread
  - Support for INotifyPropertyChanged interface

Binding

- Objects that implement INotifyPropertyChanged interface raise events when the property has changed.
- Data binding is the process of registering two properties with the data-binding engine and letting the engine keep them synchronized.
- You will find an example in the Wpf_AttachedProperties demo code.

Binding Links

MSDN Article by John Papa

CodeProject article by Josh Smith (part of a tutorial series)

Bea (Costa) Stollnitz

Control Templates

- With control templates you can change the look and feel of existing controls and support making your own controls:

  -
    ```
    <Button.Template>
       <ControlTemplate>
          <Grid><Rectangle /></Grid>
       </ControlTemplate>
    </Button.Template>
    ```

Navigation

- You can use instances of the Page and Frame classes to set up a navigation structure resembling web applications.
  - Pages go in NavigationWindow instances and Frames go in Windows and Pages.
  - This is a good alternative to tabbed displays.

Special Classes

- ContentControl
  - All UIElements derive from this.
  - Content can be text, a tree of elements, or a .Net object which can be displayed using a data template
- Dependency object
  - Derives from DispatcherObject
  - Supports data binding, styling, animation, property inheritance, and property change notifications
- WindowsFormsHost
  - Supports hosting controls based on HWNDs

Special UI Elements

- ViewBox
  - Resizes content to fit available space
- UserControl
  - Way to build custom controls as collections of elements on a panel
- Animatable
  - Provides hooks for DirectX to change elements properties over time, e.g., position, size, color, …
- FlowDocument
  - FlowDocumentScrollViewer
  - FlowDocumentPageViewer
- MediaElement
  - Play media on load or on request, e.g., wma, wmv, mp3, …

ENGINEERING@SYRACUSE