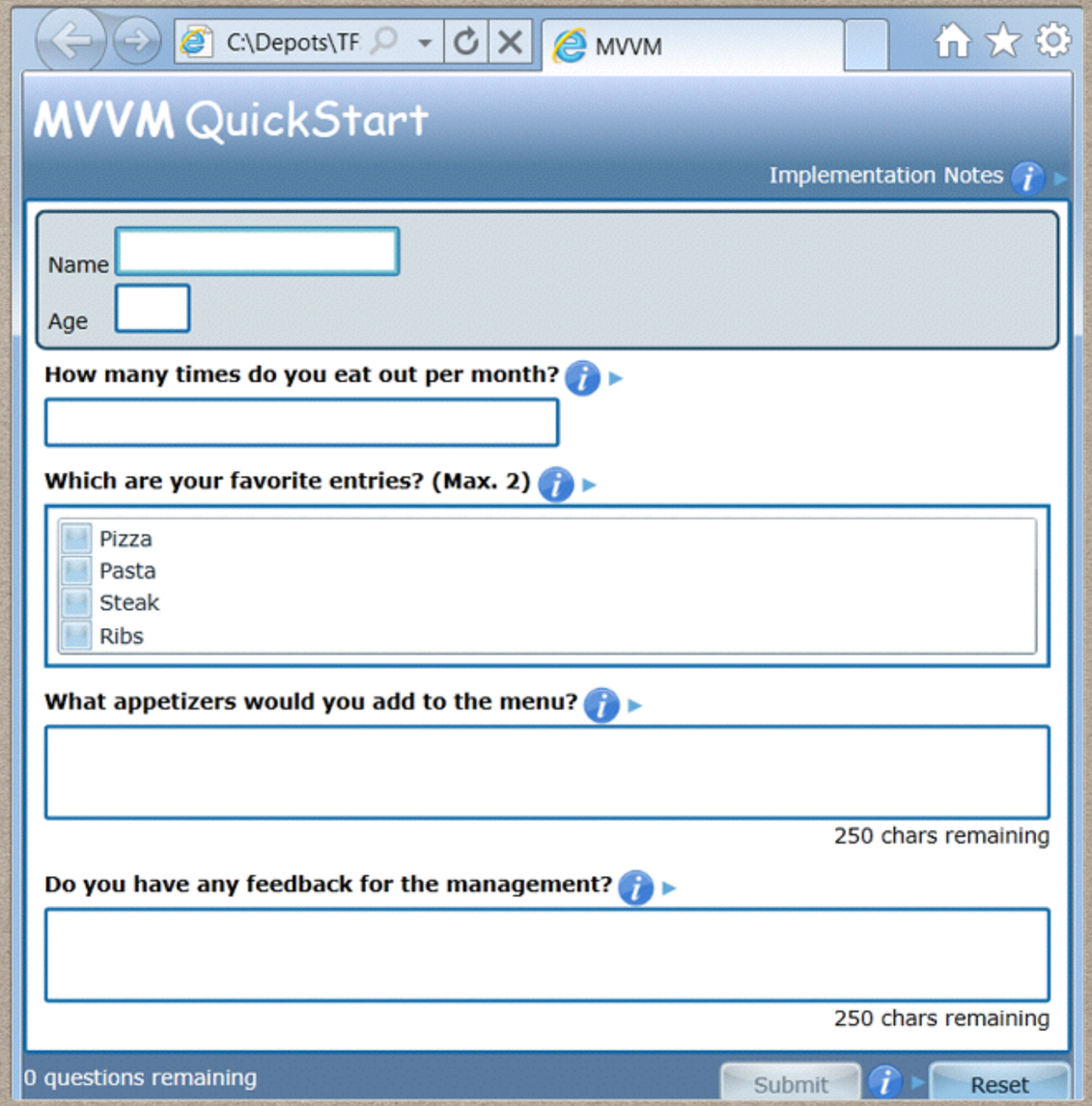
INTRO TO MVVM APPS IN .NET

PREPARED BY
NICOLA B. DIPALMA
FOR LVRUG 4/9/14 MEETUP



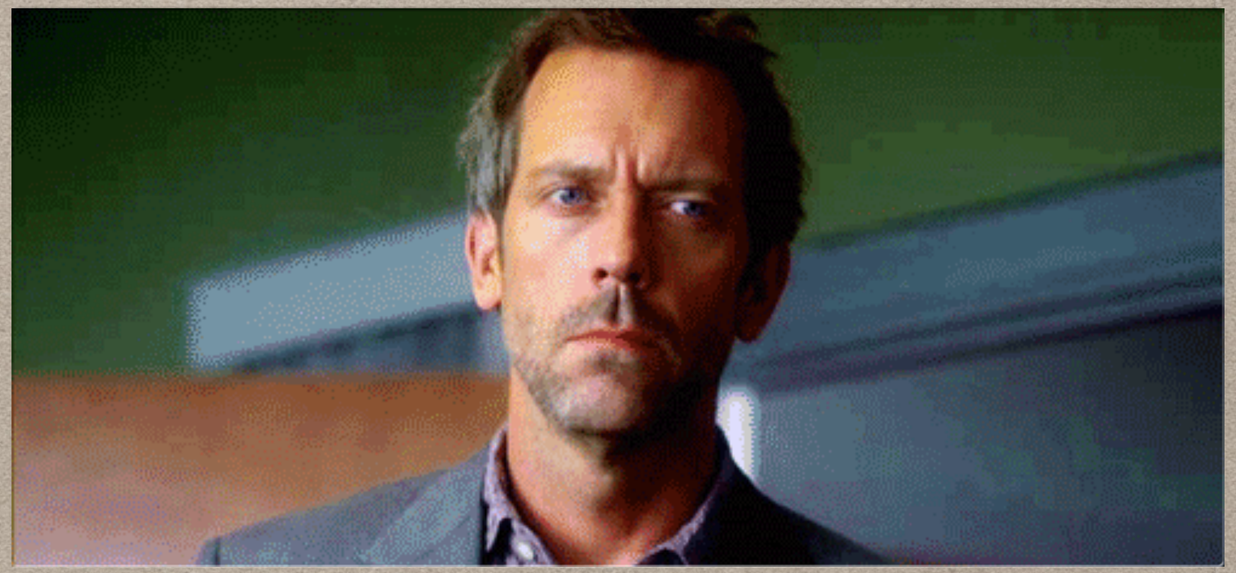
GUI for MVVM example from MSDN < http://msdn.microsoft.com/en-us/library/gg430869(v=PandP.40).aspx >

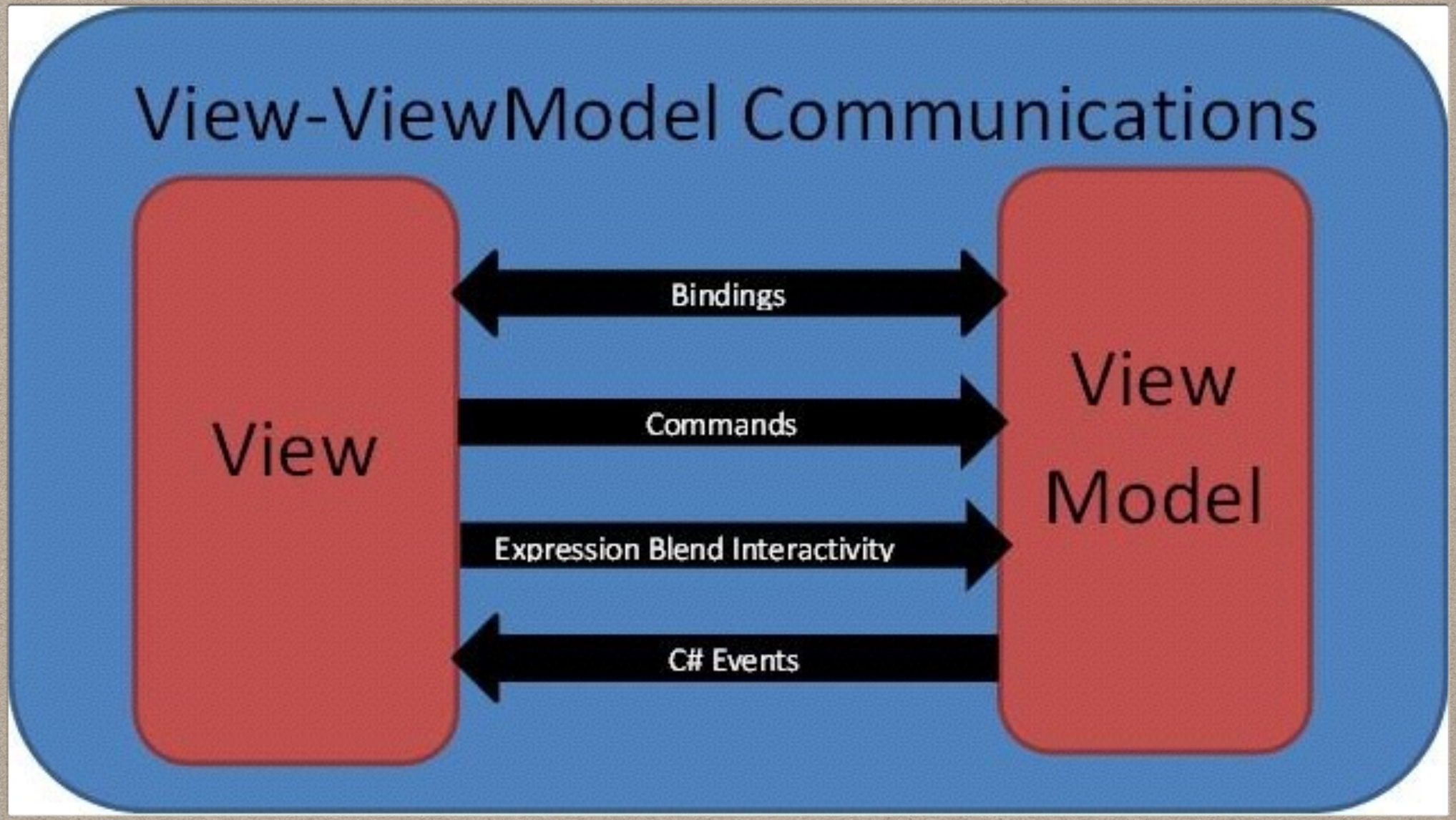
WHAT IS MVVM?

- MVVM stands for “Model-View-ViewModel”
- It’s an MVC variation based on Microsoft’s Model-View-Presenter (i.e. Presentation Model) pattern.
- The identifying factor between this and run-of-the-mill MVC is the ViewModel, which acts as both a controller and a logical representation of the UI.

WHY IS THIS USEFUL?

- Separates boundary and controller classes by mapping GUI markup directly to ViewModel objects.
- ViewModel properties are bound directly to the interface.
- As long as the View and ViewModel are bound by the same properties, they can be developed completely independent of one another.





MVVM communication diagram excerpted from: http://www.codeproject.com/Articles/278901/MVVM-Pattern-Made-Simple

(VISUAL-) BASICALLY:

- Minimizes coupling between application layers.
- Maximizes the degree of encapsulation that can be implemented separately in View-specific and Controller-specific functionality.
- Allows developers an designers to pour out their skills separately on a project without having to understand each other.
- Allows developers and designers to prototype and deploy applications quickly without being at each others necks.

THE FRONT-END (VIEW)

"THE SHOW"

WINFORMS OR XAML?

- Desktop interfaces can be developed in Winforms or via XAML markup using this pattern - Mobile and Web interfaces can be developed via HTML/CSS.
- XAML is tailored for use with this pattern and is much more flexible (my opinion).
- Implementations in Winforms are possible due to data-binding, but somewhat difficult.
- Winforms implementations typically go back to the “Presentation Model” pattern and have a testable “Presenter” class behind the view.
- Let’s talk more about XAML…

XAML

- Stands for “Extensible Application Markup Language”
- Developed as part of Project “Avalon”
- Released in June 2008.
- Markup files are linked to a C# backend, and markup objects can be instantiated as classes in C# within the back-end when creating new windows.

```
<!--
 Application Name: HiPSEC Shock Lab Tools
 Application Purpose: Provides the researcher (user) with tools and a library of functions to calculate high pressure shoc
 Developer: Nicola B. DiPalma
 Date Created: June 17th, 2013 (Version 3.0)
 Supervisor: Dr. Oliver Tschauner, Associate researcher at UNLV High Pressure Science and Engineering Center
 Program Name: HiPSEC Shock Experiment Simulator
 Reference: Shock Wave Techniques for Geophysics and Planetary Physics
            Thomas J. Ahrens
            Seismological laboratory   California Institute of Technology   Pasadena California 91125
            Published in "Methods of Experimental Physics" - Vol 24, Part A, Page 185.
            Copyright 1987 by Academic Press, Inc.
            Full text can be viewed at: http://www.fas.harvard.edu/~planets/pstewart/shrmex/index.html (Item#: 174)
 EULA: GNU General Public License v3
       Released June 29th, 2007
       Full text can be viewed at: http://www.gnu.org/licenses/gpl.html
 -->
<Window x:Class="HiPSEC_ShockLab_Tools.View.Windows.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Left="0"
        MinHeight="500"
        MinWidth="900"
        Title="HiPSEC Shock Lab Tools"
        Top="0"
        WindowStartupLocation="Manual"
        WindowState="Maximized">

    <Window.Resources>
        <ResourceDictionary Source="MainWindowResources.xaml"/>
    </Window.Resources>

    <DockPanel>

        <ScrollViewer Background="Gray" DockPanel.Dock="Left">
            <StackPanel>

                <!--AVAILABLE EXPERIMENT RECORDS LISTING (BEGIN)-->
                <ContentControl DockPanel.Dock="Left"
                                Content="{Binding Path=ExperimentsWorkspace, UpdateSourceTrigger=PropertyChanged}">
                </ContentControl>
                <!--AVAILABLE EXPERIMENT RECORDS LISTING (END)-->

                <!  AVAILABLE MATERIALS LISTING (BEGIN)  >
                <ContentControl DockPanel.Dock="Left"
                                Content="{Binding Path=MaterialsWorkspace, UpdateSourceTrigger=PropertyChanged}">
                </ContentControl>
                <!--AVAILABLE MATERIALS LISTING (END)-->

            </StackPanel>
        </ScrollViewer>

        <!--WORKSPACES (BEGIN)-->
        <ContentControl DockPanel.Dock="Left"
                        Content="{Binding Path=VisibleWorkspaces}"
                        ContentTemplate="{StaticResource WorkspacesTemplate}">
        </ContentControl>
        <!--WORKSPACES (END)-->

    </DockPanel>
</Window>
```

XAML example from HiPSEC Shocklab Tools.

THE BACK-END (VIEWMODEL)

"THE BUSINESS"

C# .NET

- Basically, it’s C/C++ without pointers - Microsoft’s version of Java.
- Shares a lot of functionality and syntax with Java.
- Runs in the .Net virtual environment in Windows applications, but many ports are available for others.
- Runs slower than C/C++ due to some overhead.
- Designed to allow developers to write functional as well as object-oriented applications.

VISUAL BASIC .NET

- Derives functionality and syntax from its predecessor, classic Visual Basic (last stable release: version 6.0 - late 1998).
- Is event-driven, just like its predecessor.
- More commonly used with Winforms interfaces.

THE MARRIAGE

"THE ART"

DATA-BINDING

- Mechanism that maps ViewModel properties to appropriately labeled {Binding} commands in the View at runtime.
- Paths are specified within the binding with the names of ViewModel properties and indexers.
- Bindings can be equipped with PropertyChanged event notifiers so that any changes in the ViewModel induce view updates.

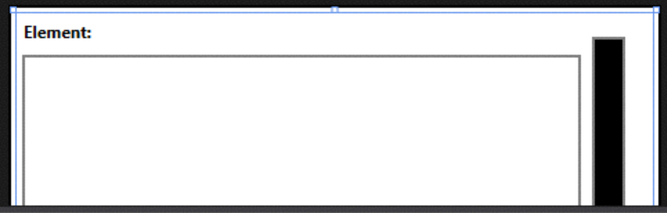
XRF Data Utilities - Microsoft Visual Studio

Quick Launch (Ctrl+Q)

EDIT VIEW PROJECT BUILD DEBUG TEAM SQL DESIGN FORMAT TOOLS TEST ARCHITECTURE DRIVER ANALYZE WINDOW HELP

Start Debug

ColorFromRamp.cs | MainWindowViewModel.cs | DumpFileHandler.cs | DoubleToBrushConverter.cs | DataRenderingWorkspace.xaml | DataRenderingWorkspaceViewModel.cs | DataRenderingWork...aceResources.xaml



100% Design XAML

```
<UserControl x:Class="XRF_Data_Utilities.View.Workspaces.DataRenderingWorkspace"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
    <UserControl.Resources>
        <ResourceDictionary Source="DataRenderingWorkspaceResources.xaml"/>
    </UserControl.Resources>
    <Border Background="White" BorderBrush="Black" BorderThickness="1" Padding="4">
        <StackPanel Orientation="Horizontal">
            <StackPanel>
                <StackPanel Orientation="Horizontal">
                    <Label FontWeight="Bold" Margin="0,0,-8,0">Element:</Label>
                    <Label Content="{Binding Path=ElementName, UpdateSourceTrigger=PropertyChanged}" FontWeight="Bold"/>
                </StackPanel>
                <Border BorderBrush="Gray" BorderThickness="2" Margin="4">
                    <ContentControl Content="{Binding Path=RenderedImage, UpdateSourceTrigger=PropertyChanged}" Height="400" Width="400" RenderOptions.BitmapScalingMode="NearestNeighbor" SnapsToDevicePix
                </Border>
            </StackPanel>
            <StackPanel Margin="4" Orientation="Horizontal">
                <Border BorderBrush="Gray" BorderThickness="2" Height="404">
                    <Canvas Background="Black" Height="400" Width="20">
                        <Rectangle Fill="{Binding Path=ColorRamp}" Height="400" Width="20"/>
                    </Canvas>
                </Border>
                <Grid>
                    <Label Content="{Binding Path=MaxCounts}" HorizontalAlignment="Left" VerticalAlignment="Top"/>
                    <Label HorizontalAlignment="Left" VerticalAlignment="Bottom">0</Label>
                </Grid>
            </StackPanel>
        </StackPanel>
    </Border>
</UserControl>
```

100 %

Error List Output

Solution Explorer

Search Solution Explorer (Ctrl+;)

- XmlFileHandler.cs
- WPFUtilities
  - Properties
  - References
  - Command
  - Model
  - PropertyChanged
  - View
  - ViewModel
  - WindowInitializer.cs
- XRF Data Utilities
  - Properties
  - References
  - Files
    - DumpFileHandler.cs
  - Model
    - coordinate.cs
    - motorSettings.cs
    - pixel.cs
    - xrfSample.cs
  - Utilities
    - ColorFromRamp.cs
    - DoubleToBrushConverter.cs
  - View
    - Windows
      - MainWindow.xaml
      - MainWindowResources.xaml
    - Workspaces
      - DataRenderingWorkspace.xaml
      - DataRenderingWorkspaceResources.xaml
    - XrfPixel.xaml
  - ViewModel
    - Windows
      - MainWindowViewModel.cs
    - Workspaces
      - DataRenderingWorkspaceViewModel.cs
    - XrfPixelViewModel.cs
  - app.config
  - App.xaml
- XRF Data Utilities Test
  - Properties
  - References
  - Files
    - DumpFileHandlerTest.cs

Solution Explorer | Team Explorer

XRF Data Utilities - Microsoft Visual Studio

Quick Launch (Ctrl+Q)

EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ARCHITECTURE DRIVER ANALYZE WINDOW HELP

Start Debug

ColorFromRamp.cs | MainWindowViewModel.cs | DumpFileHandler.cs | DataRenderingWorkspace.xaml | DataRenderingWorkspaceViewModel.cs | DataRenderingWork...aceResources.xaml | pixel.cs

XRF_Data_Utilities.ViewModel.Workspaces.DataRenderingWorkspaceViewModel

RenderImage(pixel[][] imageGrid, double imageSize)

```
19          private Canvas _renderedImage;
20
21          // Render-specific
22          private GradientStopCollection _ramp;
23
24          #endregion
25
26          ////////////////////////////////////
27          #region Properties
28
29          public LinearGradientBrush ColorRamp
30          {
31              get
32              {
33                  return new LinearGradientBrush(_ramp);
34              }
35          }
36
37          public string ElementName
38          {
39              get
40              {
41                  return _elementName;
42              }
43              set
44              {
45                  _elementName = value;
46                  OnPropertyChanged("ElementName");
47              }
48          }
49
50          public int MaxCounts
51          {
52              get;
53              set;
54          }
55
56          public Canvas RenderedImage
57          {
58              get
59              {
60                  return _renderedImage;
61              }
62              set
63              {
64                  _renderedImage = value;
65                  OnPropertyChanged("RenderedImage");
66              }
67          }
68
69          #endregion
70
71          ////////////////////////////////////
```

100 %

Solution Explorer

Search Solution Explorer (Ctrl+;)

- XmlFileHandler.cs
- WPFUtilities
  - Properties
  - References
  - Command
  - Model
  - PropertyChanged
  - View
  - ViewModel
  - WindowInitializer.cs
- **XRF Data Utilities**
  - Properties
  - References
  - Files
    - DumpFileHandler.cs
  - Model
    - coordinate.cs
    - motorSettings.cs
    - pixel.cs
    - xrfSample.cs
  - Utilities
    - ColorFromRamp.cs
    - DoubleToBrushConverter.cs
  - View
    - Windows
      - MainWindow.xaml
      - MainWindowResources.xaml
    - Workspaces
      - DataRenderingWorkspace.xaml
      - DataRenderingWorkspaceResources.xaml
    - XrfPixel.xaml
  - ViewModel
    - Windows
      - MainWindowViewModel.cs
    - Workspaces
      - DataRenderingWorkspaceViewModel.cs
    - XrfPixelViewModel.cs
  - app.config
  - App.xaml
- XRF Data Utilities Test
  - Properties
  - References
  - Files
    - DumpFileHandlerTest.cs

Solution Explorer | Team Explorer

Error List | Output

Ln 137 Col 13 Ch 13

DATA CONTEXT

- A template for interpreting the data-bound markup is established by matching the “DataContext” property of the window object.
- When assigned via a resource dictionary, ViewModel objects are given a View mapping automatically when bound to View controls as content.

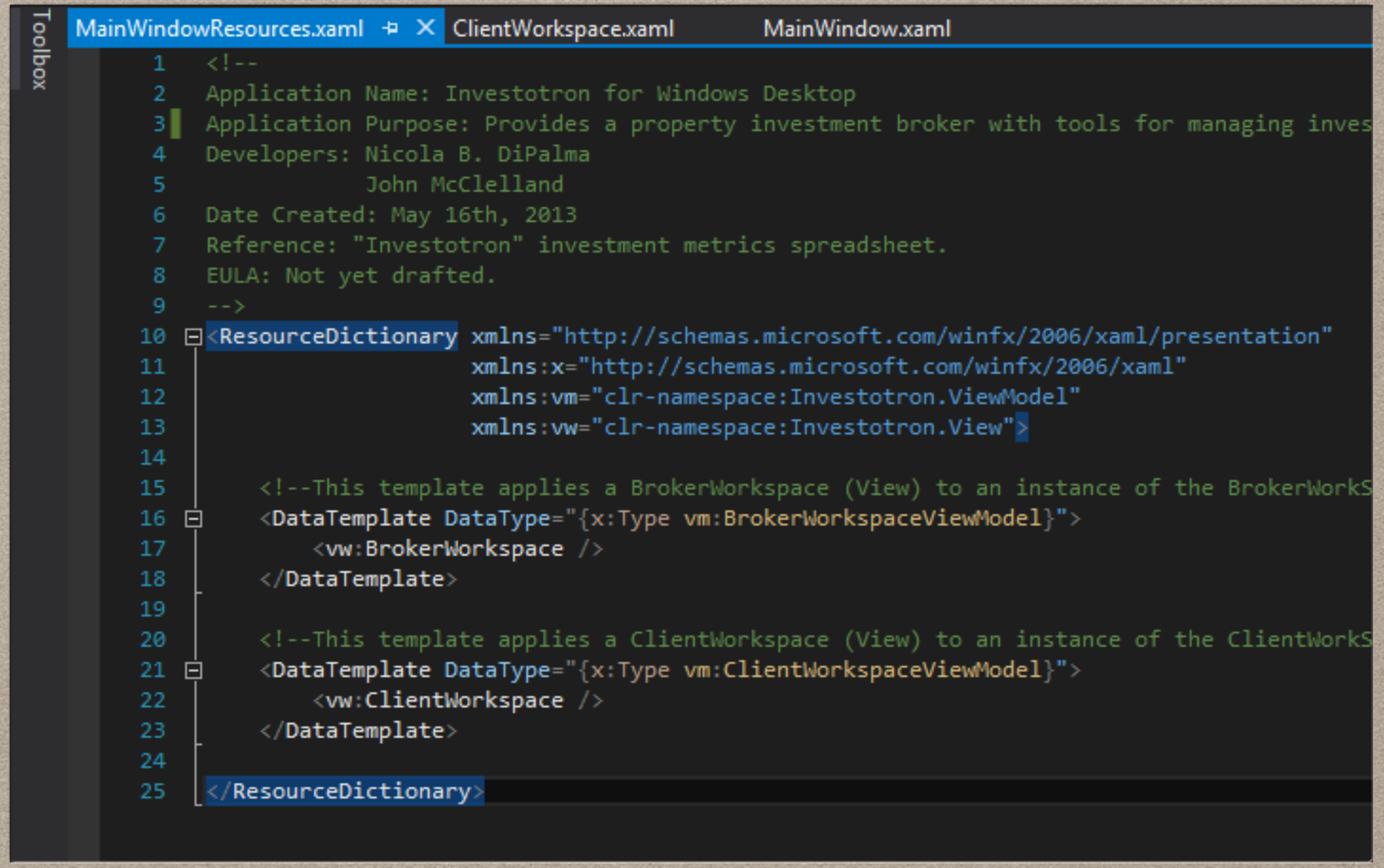
```
/// <param name="windowViewModel"></param>
public WindowInitializer(Window window, IWindow windowViewModel)
{
    _window = window;
    var viewModel = windowViewModel;
    viewModel.PropertyChanged += windowViewModel_PropertyChanged;
    window.DataContext = viewModel;
}

#endregion
```

Example of a window object and its backend instantiated in the MVVM pattern, from WPFUtilities library.

RESOURCE DICTIONARIES

- Enable additional markup files to be mapped directly to objects bound to the view.
- Reduce complexity of interface markup by enabling compartmentalization of its components.



Example of a resource dictionary from Investotron REIT software.

BUILD AN APP (TIME PERMITTING)?

OUTCOMES

- Gotten a feel for what MVVM is and how it works.
- Gotten a feel for how XAML and C# play a role in implementing the MVVM pattern.
- Acquired a deeper understanding of what data-binding is and how it makes MVVM really work.
- Acquired an understanding of how markup is mapped to various ViewModel objects as Windows and User Controls (if an app was constructed).

QUESTIONS? COMMENTS?

THE BIGGER QUESTION…

- If the engineers at Microsoft know the nature of good architecture, why do the company’s products have SO many bugs?
- I digress…