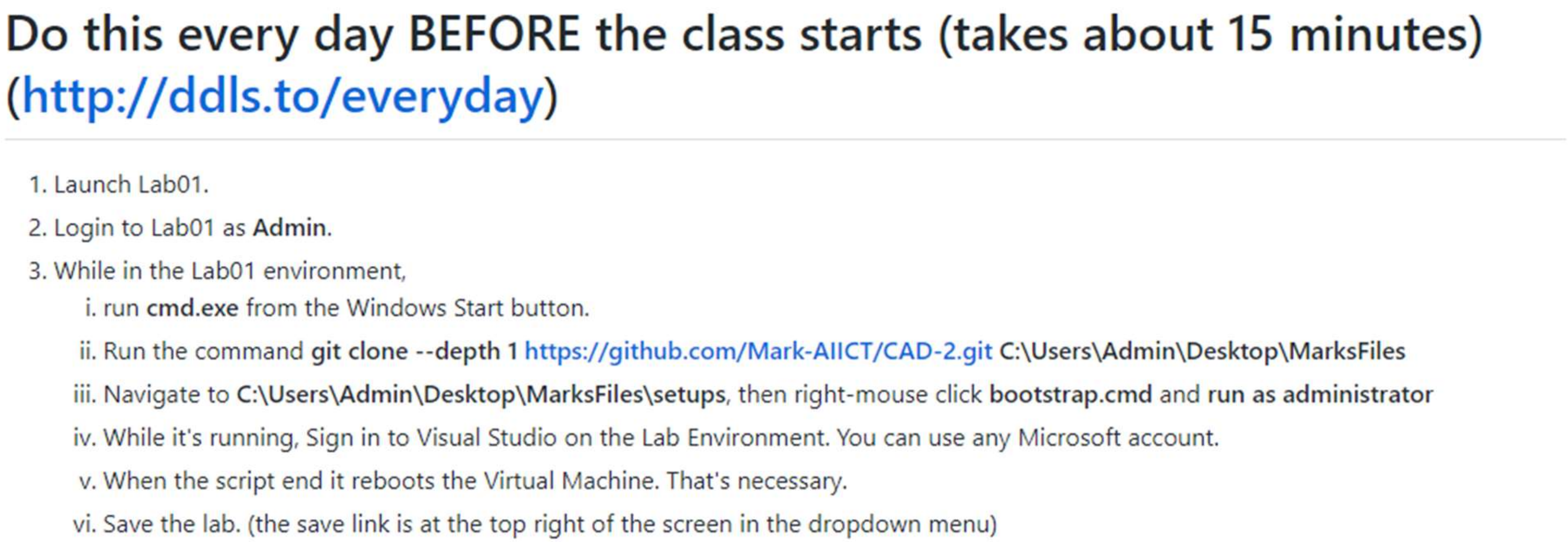
Day 9

The one link you need to recall

https://ddls.to/20483

Ready?

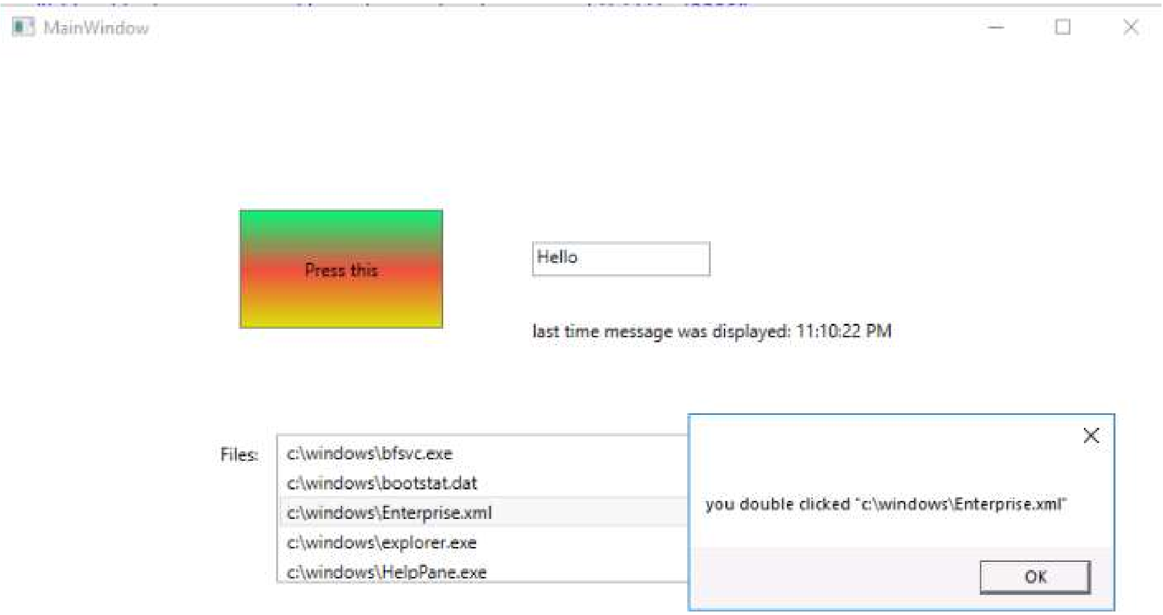


Do this every day BEFORE the class starts (takes about 15 minutes) (http://ddls.to/everyday)

1. Launch Lab01.
2. Login to Lab01 as **Admin**.
3. While in the Lab01 environment,
   i. run **cmd.exe** from the Windows Start button.
   ii. Run the command **git clone --depth 1** https://github.com/Mark-AIICT/CAD-2.git **C:\Users\Admin\Desktop\MarksFiles**
   iii. Navigate to **C:\Users\Admin\Desktop\MarksFiles\setups**, then right-mouse click **bootstrap.cmd** and **run as administrator**
   iv. While it's running, Sign in to Visual Studio on the Lab Environment. You can use any Microsoft account.
   v. When the script end it reboots the Virtual Machine. That's necessary.
   vi. Save the lab. (the save link is at the top right of the screen in the dropdown menu)

Course Outline

- Module 1: Review of Visual C# Syntax
- Module 2: Creating Methods, Handling Exceptions, and Monitoring Applications
- Module 3: Basic Types and Constructs of Visual C#
- Module 4: Creating Classes and Implementing Type-Safe Collections
- Module 5: Creating a Class Hierarchy by Using Inheritance
- Module 6: Reading and Writing Local Data
- Module 7: Accessing a Database
- Module 8: Accessing Remote Data (I’m replacing this with a better module)
- Module 9: Designing the User Interface for a Graphical Application
- Module 10: Improving Application Performance and Responsiveness
- Module 11: Integrating with Unmanaged Code
- Module 12: Creating Reusable Types and Assemblies
- Module 13: Encrypting and Decrypting Data



Hello

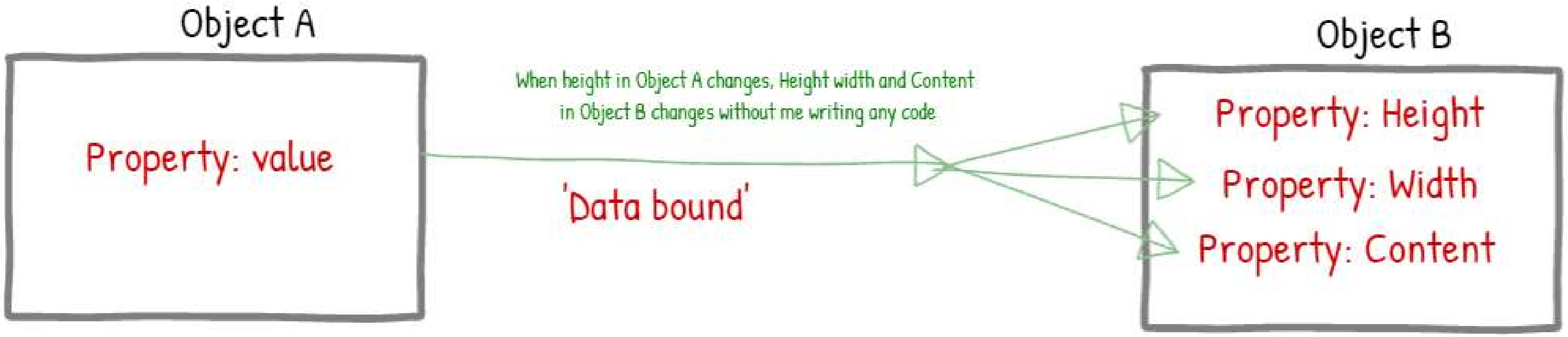
last time message was displayed: 11:10:22 PM

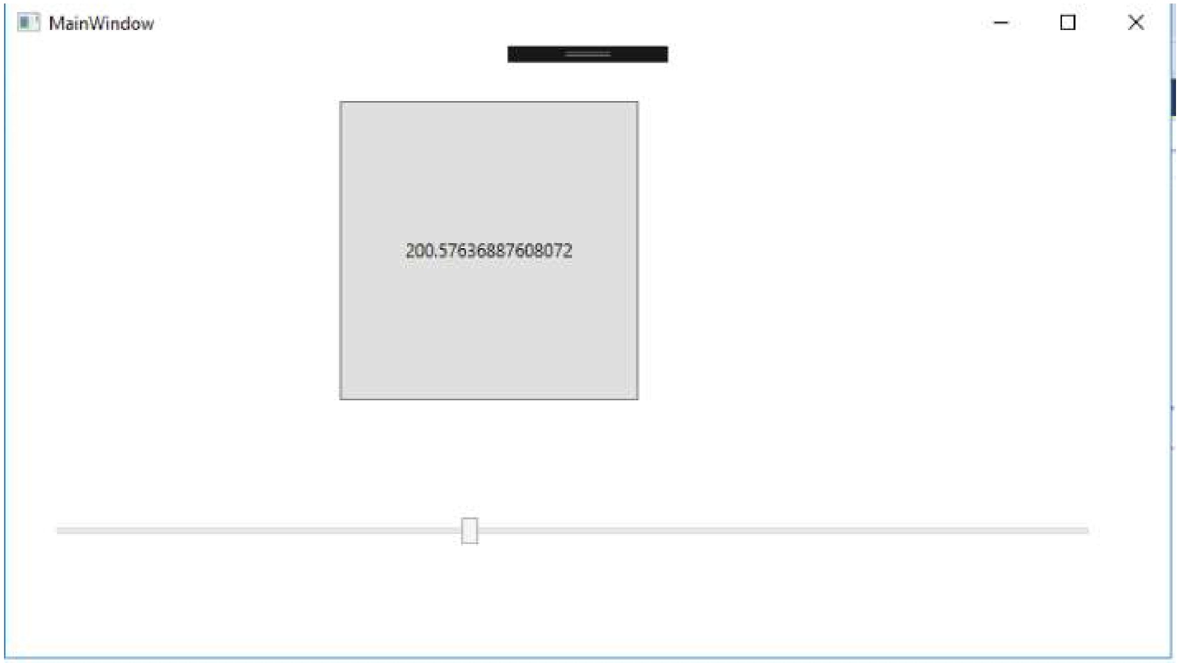
```
c:\windows\bfsvc.exe
c:\windows\bootstat.dat
c:\windows\Enterprise.xml
c:\windows\explorer.exe
c:\windows\HelpPane.exe
```

you double clicked "c:\windows\Enterprise.xml"

Lesson 1: Using XAML to Design a User Interface

- Introducing XAML
- Common Controls
- Setting Control Properties
- Handling Events
- Using Layout Controls
- Creating User Controls

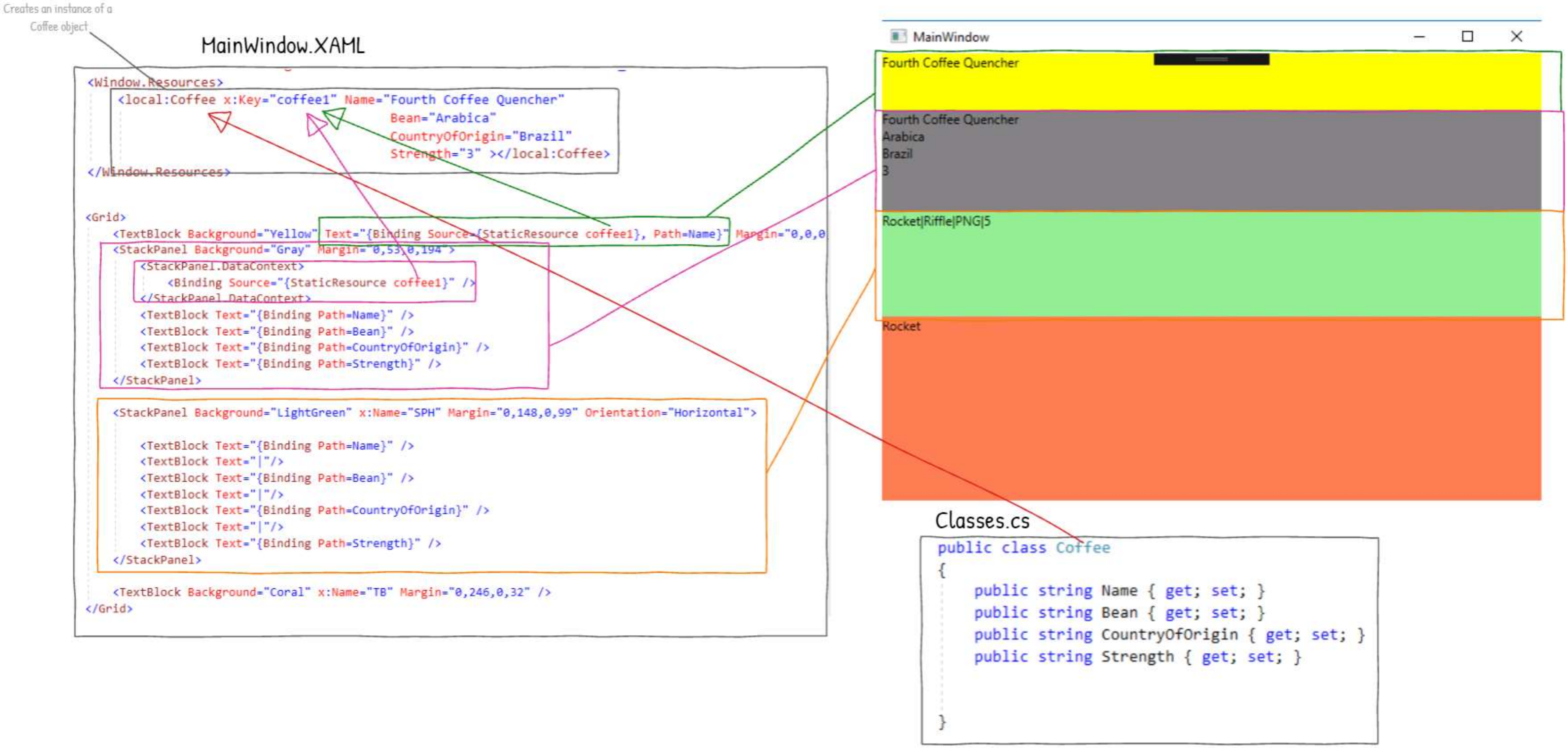

Object A
Property: value
When height in Object A changes, Height width and Content
in Object B changes without me writing any code
'Data bound'
Object B
Property: Height
Property: Width
Property: Content



200.57636887608072

Lesson 2: Binding Controls to Data

- Intoduction to Data Binding
- Binding Controls to Data in XAML
- Binding Controls to Data in Code
- Binding Controls to Collections
- Creating Data Templates



Creates an instance of a Coffee object

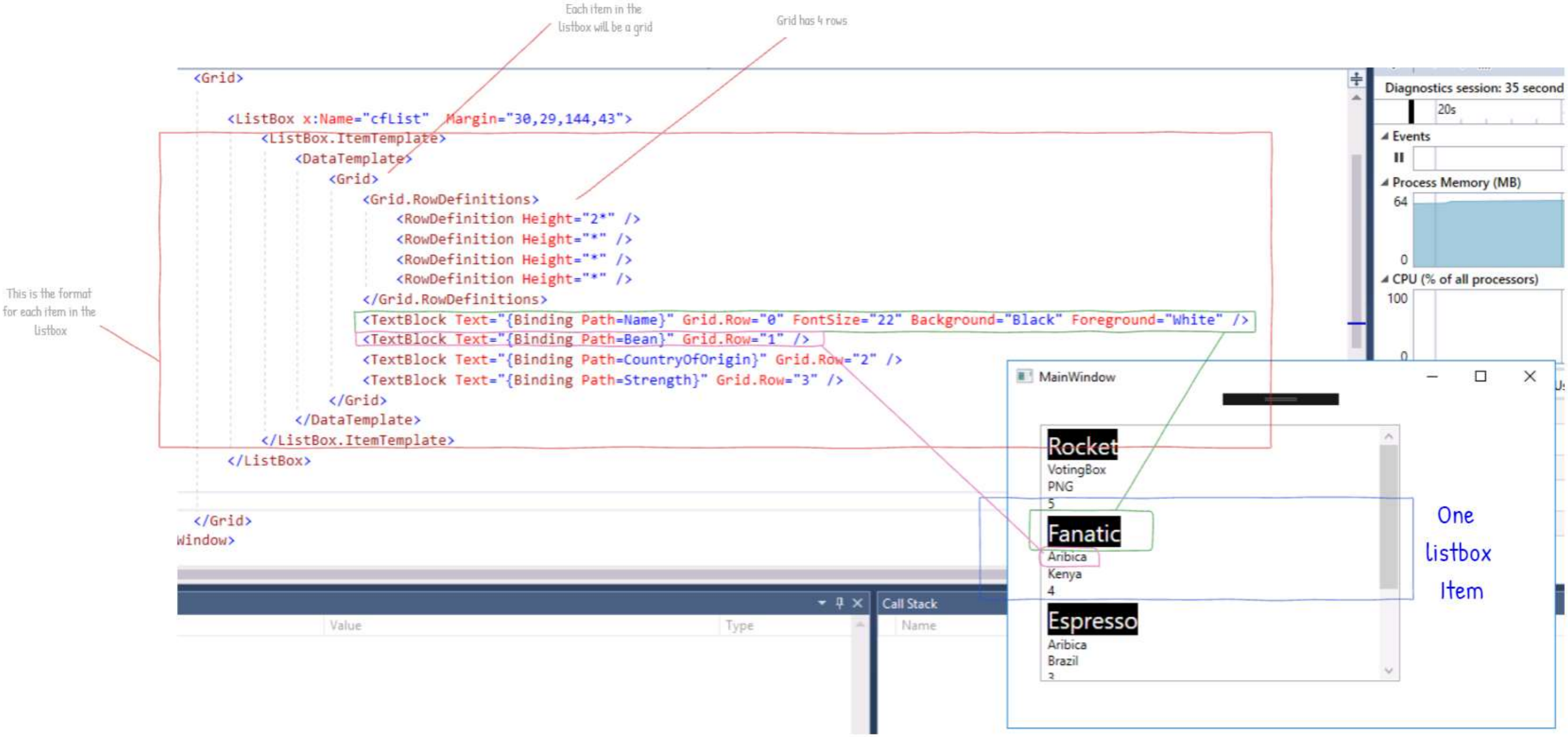
MainWindow.XAML

```xml
<Window.Resources>
    <local:Coffee x:Key="coffee1" Name="Fourth Coffee Quencher"
                                  Bean="Arabica"
                                  CountryOfOrigin="Brazil"
                                  Strength="3" ></local:Coffee>
</Window.Resources>


<Grid>
    <TextBlock Background="Yellow" Text="{Binding Source={StaticResource coffee1}, Path=Name}" Margin="0,0,0
    <StackPanel Background="Gray" Margin="0,53,0,194">
        <StackPanel.DataContext>
            <Binding Source="{StaticResource coffee1}" />
        </StackPanel.DataContext>
        <TextBlock Text="{Binding Path=Name}" />
        <TextBlock Text="{Binding Path=Bean}" />
        <TextBlock Text="{Binding Path=CountryOfOrigin}" />
        <TextBlock Text="{Binding Path=Strength}" />
    </StackPanel>

    <StackPanel Background="LightGreen" x:Name="SPH" Margin="0,148,0,99" Orientation="Horizontal">

        <TextBlock Text="{Binding Path=Name}" />
        <TextBlock Text="|"/>
        <TextBlock Text="{Binding Path=Bean}" />
        <TextBlock Text="|"/>
        <TextBlock Text="{Binding Path=CountryOfOrigin}" />
        <TextBlock Text="|"/>
        <TextBlock Text="{Binding Path=Strength}" />
    </StackPanel>

    <TextBlock Background="Coral" x:Name="TB" Margin="0,246,0,32" />
</Grid>
```

```
Fourth Coffee Quencher

Fourth Coffee Quencher
Arabica
Brazil
3

Rocket|Riffle|PNG|5

Rocket
```

Classes.cs

```csharp
public class Coffee
{
    public string Name { get; set; }
    public string Bean { get; set; }
    public string CountryOfOrigin { get; set; }
    public string Strength { get; set; }


}
```



Each item in the listbox will be a grid

Grid has 4 rows

This is the format for each item in the listbox

```
<Grid>

    <ListBox x:Name="cfList"  Margin="30,29,144,43">
        <ListBox.ItemTemplate>
            <DataTemplate>
                <Grid>
                    <Grid.RowDefinitions>
                        <RowDefinition Height="2*" />
                        <RowDefinition Height="*" />
                        <RowDefinition Height="*" />
                        <RowDefinition Height="*" />
                    </Grid.RowDefinitions>
                    <TextBlock Text="{Binding Path=Name}" Grid.Row="0" FontSize="22" Background="Black" Foreground="White" />
                    <TextBlock Text="{Binding Path=Bean}" Grid.Row="1" />
                    <TextBlock Text="{Binding Path=CountryOfOrigin}" Grid.Row="2" />
                    <TextBlock Text="{Binding Path=Strength}" Grid.Row="3" />
                </Grid>
            </DataTemplate>
        </ListBox.ItemTemplate>
    </ListBox>

</Grid>
Window>
```

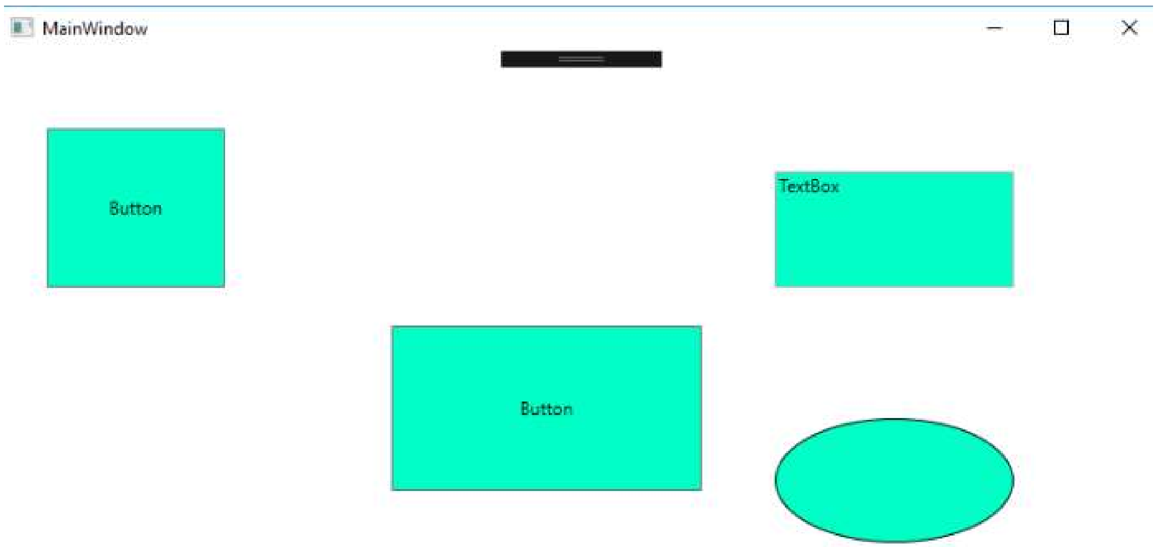
MainWindow

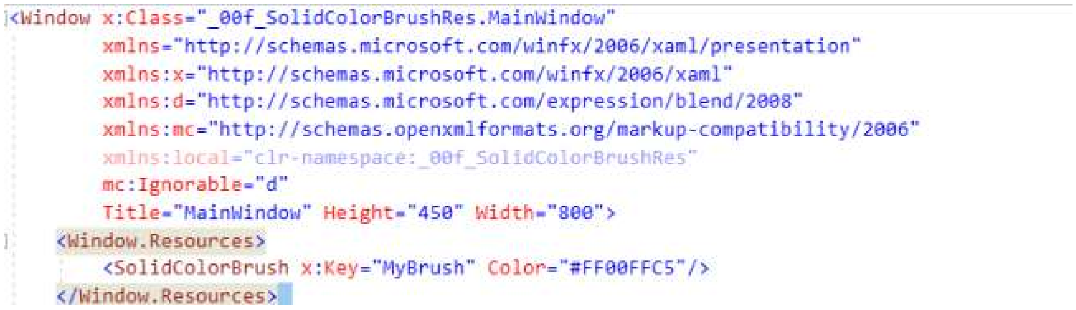
```
Rocket
VotingBox
PNG
5
Fanatic
Aribica
Kenya
4
Espresso
Aribica
Brazil
3
```

One listbox Item

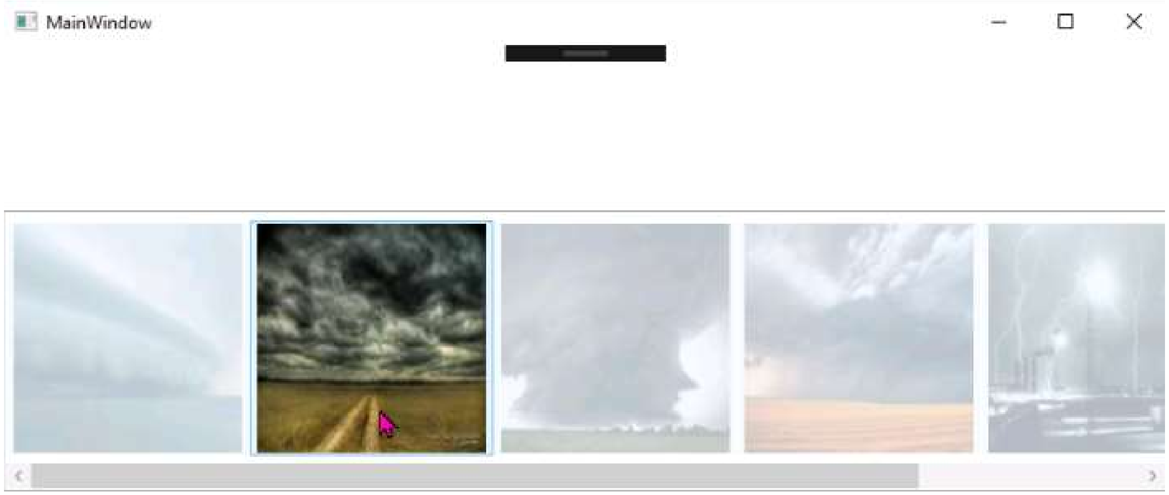
Lesson 3: Styling a UI

- Creating Reusable Resources in XAML
- Defining Styles as Resources
- Using Property Triggers
- Creating Dynamic Transformations





```
<Window x:Class="_00f_SolidColorBrushRes.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:local="clr-namespace:_00f_SolidColorBrushRes"
        mc:Ignorable="d"
        Title="MainWindow" Height="450" Width="800">
    <Window.Resources>
        <SolidColorBrush x:Key="MyBrush" Color="#FF00FFC5"/>
    </Window.Resources>
```


MainWindow

Course Outline

- Module 1: Review of Visual C# Syntax
- Module 2: Creating Methods, Handling Exceptions, and Monitoring Applications
- Module 3: Basic Types and Constructs of Visual C#
- Module 4: Creating Classes and Implementing Type-Safe Collections
- Module 5: Creating a Class Hierarchy by Using Inheritance
- Module 6: Reading and Writing Local Data
- Module 7: Accessing a Database
- Module 8: Accessing Remote Data (I’m replacing this with a better module)
- Module 9: Designing the User Interface for a Graphical Application
- Module 10: Improving Application Performance and Responsiveness
- Module 11: Integrating with Unmanaged Code
- Module 12: Creating Reusable Types and Assemblies
- Module 13: Encrypting and Decrypting Data