Today’s Lecture

- Clip – C6 @ Iowa State
  - Alternative displays
- Primer – WPF
  - Properties – Simple vs. Complex
    - Backgrounds, etc.
  - Controls
    - Grid, Label, CheckBox

Reminders
- Blog Post (Week)
- Project 1
- Homework 3

Small Group Exercise

- What are your thoughts on 3-D and “virtual reality”?
- What is the coolest VR experience that you have had?

Split into groups of 2-4 students

Simple vs. Complex Properties

- Previously
  - Focused on simple properties
  - <Button Content="Connect" />
- Complex properties
  - Property-Element Syntax
  - Complex property vs. nested object in the container
    - <Grid Name="grid1">
    - <Grid.Background>
    - ...
    - </Grid>
    - </Grid>
Haven’t we seen this before?

```xml
<Grid>
    <Grid.RowDefinitions>
        <RowDefinition />
        <RowDefinition />
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition />
        <ColumnDefinition />
    </Grid.ColumnDefinitions>
</Grid>
```

To Recap

```xml
<Grid Name="grid1">
    <Button>
        ...
    </Button>
</Grid>
```

Makes a Button in the Grid container

```xml
<Grid Name="grid2">
    <Grid.Background>
        ...
    </Grid.Background>
</Grid>
```

Modifies the background for the grid container

Combining = OK

```xml
<Grid Name="grid1">
    <Grid.RowDefinitions>
        <RowDefinition />
        <RowDefinition />
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition />
        <ColumnDefinition />
    </Grid.ColumnDefinitions>
    <Button>
        ...
    </Button>
</Grid>
```

Fancier Example - Gradient

```xml
<Grid>
    <Grid.Background>
        <LinearGradientBrush>
            <LinearGradientBrush.GradientStops>
                <GradientStop Offset="0.00" Color="Red"/>
                <GradientStop Offset="0.50" Color="Indigo"/>
                <GradientStop Offset="1.00" Color="Violet"/>
            </LinearGradientBrush.GradientStops>
        </LinearGradientBrush>
        <Grid.Background>
            ...
        </Grid.Background>
    </Grid.Background>
</Grid>
```

Two Property-Element examples

- Grid.Background
- LinearGradientBrush.GradientStops

Voila!

```xml
<Button>
    <Button.Foreground>
        <x:Static Member="SystemColors.ActiveCaptionBrush"/>
    </Button.Foreground>
</Button>
```

Markup Extensions

```xml
<Button>
    <Button.Foreground>
        <x:Static Member="SystemColors.ActiveCaptionBrush"/>
    </Button.Foreground>
</Button>
```

Markup allows us to set the property value dynamically from another control

- Static -> static property in another class
Attached Properties

```
<Grid Name="grid1">
  <Button Grid.Row="1">
  ...
  </Button>
</Grid>
```

Set the a property of this object as from its attached outer container

---

Miscellaneous Items

- How do I get XAML specific chars?
  - `, >, &, "
  - &lt;
  - &gt;
  - &amp;
  - &quot;

---

Events

- Already hinted at this
  - Will see a lightning bolt in Visual Studio

```
<Button Name="btnConnect" Grid.Row="1" Grid.Column="0"
  Click="btnConnect_Click">Connect</Button>
```

Call btnConnect_Click whenever the button is pressed

---

React to an event

```
<Button Name="btnConnect"
  Grid.Row="1" Grid.Column="0"
  Click="btnConnect_Click">Connect</Button>
```

```
private void btnConnect_Click(object sender,
  RoutedEventArgs e)
```

---

Using your classes

```
xmns:local="clr-namespace:WpfWiiSkeleton"
```

```
<local:MyObject> </local:MyObject>
```

---

Specific Controls

- CheckBox
  - Check -> yea / nay
  - Value -> Set or not
- Label
  - Text without a button
  - Normal properties for font, size, etc.
Example - Screenshot

CheckBox

Label

---

Ideal WPF Window

- Elements are not explicitly sized
  - Grow to fill content
  - Limit via min or max size
- Elements do not indicate their position
  - Arranged by the container
  - Can add whitespace via Margin
- Layout containers share space among children
  - May specify preferred size
- Layout containers can be nested
  - Start with grid, work inwards

---

GUI Positioning

- Think of a window as a set of containers
  - Nest containers in containers in containers in ...
  - Different broad containers
    - Grid
    - StackLayout
    - DockLayout
    - Canvas

---

Layout Process

- Two stages
  - Measure
    - Foreach child in container
  - Arrange
    - What's your preferred size?
    - This is your size

---

Core Layout Panels

- Types
  - StackPanel
    - Vertical or horizontal stack
  - WrapPanel
    - Tries to lay out in lines or columns as specified, think like flowing water
  - DockPanel
    - Align versus edge
  - Grid
    - Arrange according to invisible table
    - More commonly used containers

---

Core Layout Panels

- UniformGrid
  - More restrictive grid
- Canvas
  - Absolute positioning
  - Bad choice for WPF without work
  - Unless
    - ScatterView
    - InkCanvas
StackPanel

<StackPanel Grid.Row="1" Grid.Column="2" Grid.RowSpan="2">
    <Label>Button Stack</Label>
    <Button>Button 1</Button>
    <Button>Button 2</Button>
    <Button>Button 3</Button>
    <Button>Button 4</Button>
    <Button>Button 5</Button>
    <CheckBox>Haz Cheeseburger</CheckBox>
</StackPanel>

Stack Panel - Screenshot

StackPanel

<StackPanel Orientation="Horizontal" Grid.Row="2" Grid.Column="0" Grid.ColumnSpan="3">
    <Label>Button Stack</Label>
    <Button>Button 1</Button>
    <Button>Button 2</Button>
    <Button>Button 3</Button>
    <Button>Button 4</Button>
    <Button>Button 5</Button>
    <CheckBox>Haz Cheeseburger</CheckBox>
</StackPanel>

Horizontal Stack

Questions?

- Weekly Blog
- Homework 3
- Project 1

Project 2 will be coming soon as well – Wii Fit