

Exploit WPF Graphics Without Wounding the Eyes

Brian Noyes
Chief Architect, IDesign Inc
(www.idesign.net)

Pre-requisites for this presentation:

- 1) Understand WPF Fundamentals
- 2) Experience with Windows Forms or ASP.NET UI design

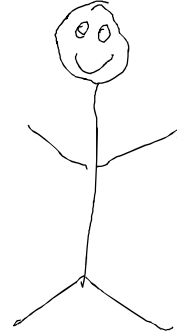
Level: Intermediate

About Brian

- | Chief Architect, IDesign Inc. (www.idesign.net)
- | Microsoft Regional Director / MVP
- | Publishing
 - **Developing Applications with Windows Workflow Foundation**, LiveLessons training DVD, June 2007.
 - **Smart Client Deployment with ClickOnce**, Addison Wesley, January 2007
 - **Data Binding in Windows Forms 2.0**, Addison Wesley, January 2006
 - MSDN Magazine, MSDN Online, CoDe Magazine, The Server Side .NET, asp.netPRO, Visual Studio Magazine
- | Speaking
 - Microsoft TechEd US, Europe, Malaysia, Visual Studio Connections, DevTeach, INETA Speakers Bureau, MSDN Webcasts
- | Participates in Microsoft Design Reviews
- | E-mail: brian.noyes@idesign.net
- | Blog: <http://briannoyes.net>

Agenda

- | UI Design Principles
- | Styles
- | Themes
- | Templates
- | Animations
- | Transparency



Know Your User

- | Your user is not you
- | Your user is not interested in using your software, they are interested in getting their job done
- | Your highest priority development tasks may not be the highest priority user tasks
- | Need user involvement in the development process
 - Observe users doing their job with current software / processes
 - Review designs with users or user proxy
 - Perform usability studies

UI Design Principles

- I Too much information in a screen is a bad thing
- I Too little information forces unnecessary navigation

UI Design Principles

- I Keep frequent tasks close to the top
- I Minimize major context switches – lessen the “cognitive distance” between user tasks
 - Pop ups
 - Pane swap outs
- I Provide navigation hints and links when taking the user away from where they were
- I Provide focus hints to draw the user's attention to where it needs to be

UI Design Principles

- | Following design conventions may not be a good thing
 - Whose convention?
 - Has the user been exposed to the convention?
 - Does branding /differentiation matter?
- | However, don't force the user to learn everything from scratch

UI Design Principles

- | Be forgiving of user mistakes
- | Don't distract the user
- | Support different types of users with separate UIs
 - Driven by user preferences

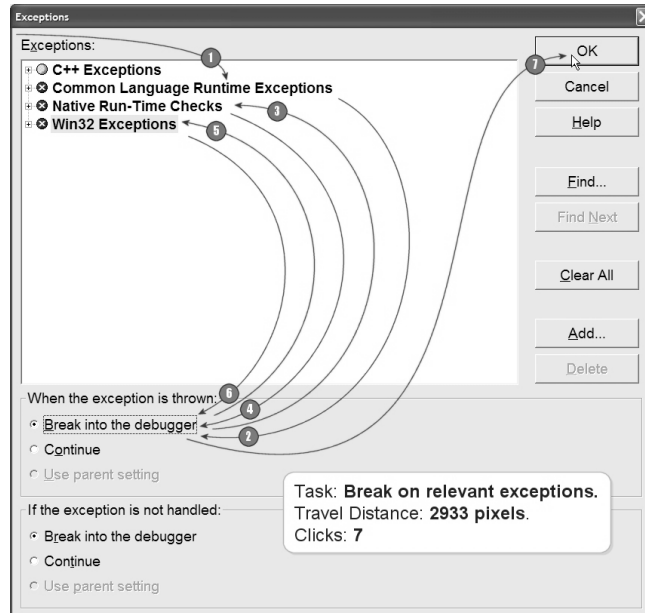
Differentiated UI/UX

- | A fuzzy term at best
- | Your application provides a unique experience for the user compared to other applications
 - Unique can be good or very very bad
- | Branding / theming – WPF Styles
- | Dynamic UI behaviors – WPF Animations
- | Specialized interaction models – WPF custom controls

Measuring a Good UI

- | Usability Testing
- | User/User Proxy feedback
- | Keystroke counting
- | Mouse travel distance
- | Shifts in eye focus

Example: Mouse Travel



Agenda

- | UI Design Principles
- | Styles
- | Themes
- | Templates
- | Animations
- | Transparency

WPF Styles

- | Allows you to alter the appearance and behavior of controls without deriving new control types or doing custom drawing
- | Similar in concept and capability to CSS
- | Can apply discretely to individual controls
- | Can apply to all controls of a given type in a given UI scope
 - Window, or any container control in the element hierarchy

WPF Styles

- | Define style as a resource
- | Add property setters
- | Can apply based on type and with name:

```
<Window x:Class="StyleDemo.Window1"
  xmlns="http://schemas.microsoft.com/wfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/wfx/2006/xaml"
  Title="StyleDemo" Height="300" Width="300"
  >
  <Window.Resources>
    <Style x:Key="BlueButton" TargetType="{x:Type Button}">
      <Setter Property="Background" Value="Blue"/>
      <Setter Property="FontSize" Value="16" />
    </Style>
  </Window.Resources>
  <Grid >
    <Button Style="{StaticResource BlueButton}" Height="23" Margin="82, 46, 135, 0"
      Name="button1" VerticalAlignment="Top">Button</Button>
  </Grid>
</Window>
```

WPF Styles

- I Can apply based on name alone to multiple types:

```
<Window x:Class="StylesDemo.Window1"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  Title="StylesDemo" Height="300" Width="300"
  >
<Window.Resources>
  <Style x:Key="BlueButton" >
    <Setter Property="Control.Background" Value="Blue"/>
    <Setter Property="Control.FontSize" Value="16" />
  </Style>
</Window.Resources>
<Grid >
  <Button Style="{StaticResource BlueButton}" Height="23" Margin="82, 46, 135, 0"
    Name="button1" VerticalAlignment="Top">Button</Button>
  <TextBox Style="{StaticResource BlueButton}" Height="26" Margin="77, 0, 115, 61"
    Name="textBox1" VerticalAlignment="Bottom"></TextBox>
</Grid>
</Window>
```

WPF Styles

- I Can apply to all controls of a given type:

```
<Window x:Class="StylesDemo.Window1"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  Title="StylesDemo" Height="300" Width="300"
  >
<Window.Resources>
  <Style TargetType="{x:Type Button}" >
    <Setter Property="Control.Background" Value="Blue"/>
    <Setter Property="Control.FontSize" Value="16" />
  </Style>
</Window.Resources>
<Grid >
  <Button Height="23" Margin="82, 46, 135, 0" Name="button1"
    VerticalAlignment="Top">Button</Button>
  <Button Height="23" Margin="81, 83, 136, 0" Name="button2"
    VerticalAlignment="Top">Button</Button>
</Grid>
</Window>
```

Agenda

- | UI Design Principles
- | Styles
- | Themes
- | Templates
- | Animations
- | Transparency

WPF Themes

- | Collections of styles and templates
- | Applied to create a consistent user interface throughout the application
- | Defined through resource dictionaries
- | Can be internal (in the assembly) or external (another assembly)
- | Several themes built-in
 - Default selected based on operating system by WPF
- | Can create custom themes

WPF Built in Themes

OS Theme Source	Source
Windows Vista	Aero.NormalColor.xaml
Windows XP Default	Luna.NormalColor.xaml
Windows XP Silver	Luna.Metallic.xaml
Windows XP Olive	Luna.Homestead.xaml
Windows Media Center	Royale.NormalColor.xaml
Windows XP Zune	Zune.NormalColor.xaml
Windows Classic	Classic.xaml

Agenda

- | UI Design Principles
- | Styles
- | Themes
- | Templates
- | Animations
- | Transparency

WPF Control Templates

- I Define new visual tree through resource
- I Apply with Template property on control

```
<Window x:Class="ControlTemplates.Window1"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  Title="Control Templates" Height="300" Width="300"
  >
<Window.Resources>
  <ControlTemplate x:Key="mybutton" >
    <Ellipse Width="100" Height="50" Fill="Blue" />
  </ControlTemplate>
</Window.Resources>
<Grid>
  <Button Click="OnButtonClicked" Template="{StaticResource mybutton}"
    Name="button1" VerticalAlignment="Top">Button</Button>
</Grid>
</Window>
```

WPF Data Templates

- I Define the UI for a data item
- I Chunk of XAML that can be applied to a data item or collection of data items
- I Typically used with list-oriented controls (ItemsControl derived)

```
<DataTemplate x:Key="ImageCellTemplate">
  <Image Source="{Binding AlbumCover}" />
</DataTemplate>
```

```
<GridViewColumn Header="Cover"
  CellTemplate="{StaticResource ImageCellTemplate}" />
```

Agenda

- | UI Design Principles
- | Styles
- | Themes
- | Templates
- | Animations
- | Transparency

WPF Animation

- | Can be used for good or evil
- | Use for view transitions
 - Lessens “cognitive distance” between views
- | Use for grabbing users attention to relevant tasks
 - Draw them into the current task or focus
 - Don’t distract
 - If you must draw their focus to something that is “far” away from where their eyes are, use animation to pull their attention to the new location
- | Can be time-based or reactive to user events (or both)

WPF Animations

- | Composed of:
 - Storyboards
 - Timelines
 - Triggers
 - Paths
- | Can only animate DependencyProperties
- | Really just fancy property setters

Agenda

- | UI Design Principles
- | Styles
- | Themes
- | Templates
- | Animations
- | Transparency

Transparency / Opacity

- | Use to lessen “cognitive distance”
 - Let hidden UI show through
- | Use in combination with animation to ease transitions
- | Use to avoid claiming screen real estate you don't need
 - Non-rectangular windows

Resources

- | *Family.Show* - <http://www.codeplex.com/familyshow>
- | *WTF – WPF Transition Framework* - <http://blog.nukeation.com/post/WTF---WPF-Transformation-Framework.aspx>
- | *Visual Explanations: Images and Quantities, Evidence and Narrative*, Edward Tufte, ISBN 0961392126.
- | *Designing Interfaces*, Jenifer Tidwell, O'Reilly & Associates, ISBN 0596080031.
- | *WPF Unleashed*, Adam Nathan, Sams Publishing, ISBN 0672328917.
- | *Programming WPF*, Chris Sells & Ian Griffiths, O'Reilly & Associates, ISBN 0596510373.
- | *Essential WPF*, Chris Anderson, Addison Wesley, ISBN 0321374479.
- | E-mail: brian.noyes@idesign.net
- | Blog: <http://briannoyes.net>