Transition Effects

Aka Animation

Introducing Animations

- In movies:
  - Animation is an illusion that is created by quickly cycling through a series of images, each slightly different from the last.
- On a computer - Prior to WPF:
  - To make a text grow:
    - The program creates a timer.
    - The program checks the timer at set intervals to see how much time has elapsed.
    - Each time the program checks the timer, it computes the current font size based on how much time has elapsed.
    - The program then updates the fontSize with the new value and WPF redraws it.

```
void timer_Tick(object sender, EventArgs e)
      {
         long elapsed = Environment.TickCount - _start;
         if (elapsed >= 5000)
         {
            _button1.FontSize = 18.0;
            ((DispatcherTimer)sender).IsEnabled = false;
            return;
         }
         _button1.FontSize = 9.0 + (9.0 / (5000.0 / elapsed));
      }
```

Animations in WPF

- WPF includes an efficient timing system that makes it easy to animate controls and other graphical objects.
- WPF handles all the behind-the-scenes work of managing a timing system and redrawing the screen efficiently.
- Animation can be defined as the modification of a value based on time.
- You animate objects by applying animation to their individual **properties**.
  - For example, to make a framework element grow, you animate its Width and Height properties.
  - To make an object fade from view, you animate its Opacity property.

Animations in WPF – C#

```
public Window1()
     {
        InitializeComponent();

        DoubleAnimation animation = new DoubleAnimation();
        //animation.From = 9.0; // You don't have to specify that
        animation.To = 24.0;
        animation.Duration = TimeSpan.FromMilliseconds(5000);
        _button1.BeginAnimation(Button.FontSizeProperty, animation);
     }
```

Animations in WPF – XAML

```
<Button.Triggers>
    <EventTrigger RoutedEvent="Button.MouseEnter">
        <EventTrigger.Actions>
            <BeginStoryboard>
                <Storyboard>
                    <DoubleAnimation To="18.0"
                                     Duration="0:0:5"
                                     Storyboard.TargetName="_button1"
                                     Storyboard.TargetProperty="FontSize"
                                     />
                 </Storyboard>
             </BeginStoryboard>
        </EventTrigger.Actions>
    </EventTrigger>
. . .
```

WPF Property Animation

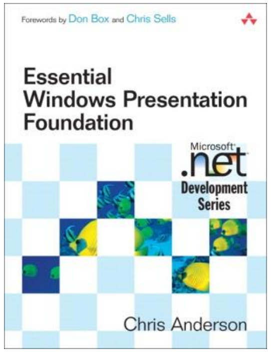
- For a property to have animation capabilities, it must meet the following three requirements:
  - It must be a **dependency property**.
  - It must belong to a class that inherits from Dependency Object and implements the **IAnimatable** interface.
  - There must be a compatible animation type available.
    - (If WPF does not provide one, you can create your own)

- You can use animations almost anywhere, which includes in styles and control templates.

References

- Pro WPF chapter 15 and 16

- Essential Windows Presentation Foundation
  By Chris Anderson
  Publisher: Addison-Wesley Professional
  ISBN-13:9780321374479



Forewords by Don Box and Chris Sells

Essential Windows Presentation Foundation

Microsoft .net Development Series

Chris Anderson

- Animation Overview
  http://msdn.microsoft.com/en-us/library/ms752312.aspx