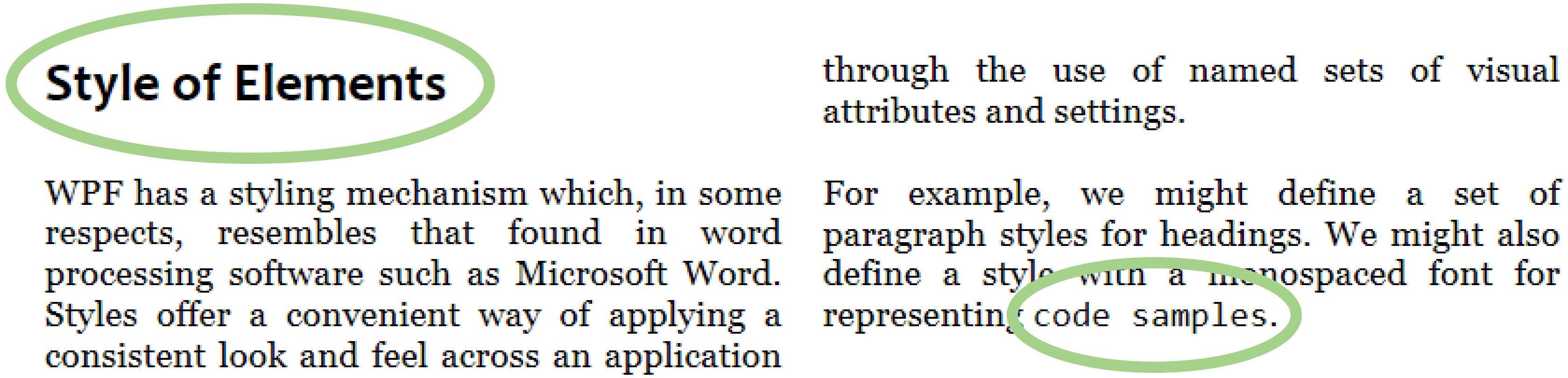
Styles

Outline

- Styles and Properties
- Resources
- Triggers
- Styling, templates, & controls

Styles



Style of Elements

WPF has a styling mechanism which, in some respects, resembles that found in word processing software such as Microsoft Word. Styles offer a convenient way of applying a consistent look and feel across an application through the use of named sets of visual attributes and settings.

For example, we might define a set of paragraph styles for headings. We might also define a style with a monospaced font for representing `code samples`.

```
<Style x:Key="heading">
  <Setter Property="TextElement.FontFamily" Value="Candara" />
  <Setter Property="TextElement.FontSize" Value="24" />
  <Setter Property="Paragraph.TextAlignment" Value="Left" />
  <Setter Property="Paragraph.Margin" Value="0,25" />
  <Setter Property="TextElement.FontWeight" Value="Bold" />
</Style>

<Style x:Key="code">
  <Setter Property="TextElement.FontFamily" Value="Consolas" />
</Style>
```

Styles and Properties

- **Styles set properties**

```
<Style>
  <Setter Property="Control.FontFamily" Value="Segoe Script" />
  <Setter Property="Control.Foreground" Value="DarkRed" />
</Style>
```

Resources

```
...
<Window.Resources>
  <Style x:Key="myTextStyle">
    <Setter Property="Control.FontFamily" Value="Segoe Script" />
    <Setter Property="Control.Foreground" Value="DarkRed" />
  </Style>
</Window.Resources>
...

  <Button Style="{StaticResource myTextStyle}">
  ...
```

```
...
<Application.Resources>
  <Style TargetType="{x:Type Button}">
    <Setter Property="FontFamily" Value="Segoe Script" />
    <Setter Property="Foreground" Value="DarkRed" />
  </Style>
</Application.Resources>
...
```

Extending Styles

```
<Style x:Key="myExtendedTextStyle"
       BasedOn="{StaticResource myTextStyle}">

  <Setter Property="Control.FontStyle" Value="Italic" />
</Style>

<Style x:Key="myExtendedButtonStyle"
       BasedOn="{StaticResource {x:Type Button}}">

  <Setter Property="Control.FontStyle" Value="Italic" />
</Style>
```

Styles vs Local Properties

- **Properties supplied by style are inherited**

- **Can mix of setter target types**
  - Inapplicable properties ignored

```
<Style x:Key="mixed">
  <Setter Property="Control.Background" Value="Red" />
  <Setter Property="TextElement.FontFamily" Value="Palatino Linotype" />
  <Setter Property="DockPanel.Dock" Value="Top" />
  <Setter Property="Paragraph.TextAlignment" Value="Justify" />
</Style>
```

Triggers

```
<Style x:Key="mixed">
  <Setter Property="TextElement.FontFamily" Value="Palatino Linotype" />

  <Style.Triggers>
    <Trigger Property="UIElement.IsMouseOver" Value="True">
      <Setter Property="TextElement.FontStyle" Value="Italic" />
    </Trigger>
  </Style.Triggers>

</Style>
```

Animation Triggers (Properties)

```
<Style x:Key="mixed">
  <Setter Property="TextElement.FontFamily" Value="Palatino Linotype" />

  <Style.Triggers>
    <Trigger Property="UIElement.IsMouseOver" Value="True">
      <Trigger.EnterActions>
        <BeginStoryboard>
          <Storyboard>
            <DoubleAnimation Duration="0:0:2" To="75"
                Storyboard.TargetProperty="(TextElement.FontSize)" />
          </Storyboard>
        </BeginStoryboard>
      </Trigger.EnterActions>

      <Trigger.ExitActions>
        <BeginStoryboard>
          <Storyboard>
            <DoubleAnimation
                Storyboard.TargetProperty="(TextElement.FontSize)" />
          </Storyboard>
        </BeginStoryboard>
      </Trigger.ExitActions>
    </Trigger>
  </Style.Triggers>
</Style>
```

Animation Triggers (Events)

```
<Style x:Key="mixed">
  <Setter Property="TextElement.FontFamily" Value="Palatino Linotype" />

  <Style.Triggers>
    <EventTrigger RoutedEvent="UIElement.MouseEnter">
      <BeginStoryboard>
        <Storyboard>
          <DoubleAnimation Duration="0:0:2" To="75"
              Storyboard.TargetProperty="(TextElement.FontSize)" />
        </Storyboard>
      </BeginStoryboard>
    </EventTrigger>
  </Style.Triggers>
</Style>
```

Items Container Style

- **ItemsControls generate container**
  - ListBox → ListBoxItem
  - TreeView → TreeViewItem
  - etc.

- **ItemContainerStyle applied to container**

Styles, Templates, and Controls

- **Theming typically uses styles**
  - Setter for Template
- **Templates reflect properties**

Summary

- Styles and Properties
- Resources
- Triggers
- Styling, templates, & controls