Estilos e Comportamentos

José Antônio da Cunha

IFRN

Estilos e Comportamentos

Estilos (Styles)

Estilo é um conjunto de valores de propriedade que você pode aplicar a um elemento em uma única etapa. No Silverlight, estilos permitem agilizar a marcação XAML simplificando a formatação de detalhes repetitivos.

O sistema de estilo Silverlight desempenha um papel semelhante ao da folha de estilo em cascata (CSS) padrão de marcação HTML. Como CSS, estilos Silverlight permitem definir um conjunto comum de características de formatação e aplicá-los em toda a sua aplicação, para assegurar a consistência.

Estilos e Comportamentos

Definindo Estilo

O primeiro passo é definir um objeto de estilo que envolve todas as propriedades que você deseja definir. Você vai armazenar esse objeto Style como um recurso, geralmente na coleção UserControl.Resources que mantém os recursos de toda a página:

```
<UserControl.Resources>
        <Style x:Key="BigButtonStyle" TargetType="Button" >
            <Setter Property="FontFamily" Value="Georgia" />
            <Setter Property="FontSize" Value="40" />
            <Setter Property="Foreground" Value="SlateGray" />
            <Setter Property="Padding" Value="20" />
            <Setter Property="Margin" Value="10" />
        </Style>
    </UserControl.Resources>
```

Estilos e Comportamentos

Definindo Estilo – outro exemplo

```
<UserControl.Resources>
      <Style x:Key="BigButtonStyle" TargetType="Button" >
         <Setter Property="FontFamily" Value="Georgia" />
         <Setter Property="FontSize" Value="40" />
         <Setter Property="Foreground" Value="SlateGray" />
         <Setter Property="Padding" Value="20" />
         <Setter Property="Margin" Value="10" />
      </Style>
      <Style x:Key="BackgroundButtonStyle2" TargetType="Button" >
         <Setter Property="Background">
            <Setter.Value>
               <LinearGradientBrush StartPoint="0,0" EndPoint="1,0">
                  <GradientStop Color="Blue"></GradientStop>
                  <GradientStop Color="Yellow" Offset="1"></GradientStop>
               </LinearGradientBrush>
            </Setter.Value>
         </Setter>
      </Style>
  </UserControl.Resources>
```

Estilos e Comportamentos

Aplicando Estilo

Cada elemento do Silverlight pode usar um único estilo. No entanto, para configurar um botão para usar o estilo que você criou anteriormente, aponte o botão para o recurso de estilo como este:

```
<Grid x:Name="LayoutRoot" Background="White">
        <Button x:Name="btnOK" Content="Confirmar" Height="90" Width="300"
                Style="{StaticResource BigButtonStyle}"></Button>
        <Button x:Name="btnNOK" Content="Cancelar" Height="90" Width="300"
                Style="{StaticResource BackgroundButtonStyle2}"
Margin="50,210,50,0"></Button>
    </Grid>
```

Estilos e Comportamentos

Aplicando Estilo

Geralmente vincula-se os estilos na sua marcação (XAML), no entanto, pode-se vinculá-los programaticamente.

O código a seguir, vincula o estilo BigButtonStyle da coleção recursos da página, e aplicá-o ao objeto botão btnOk.

```
btnOK.Style = (Style)this.Resources["BigButtonStyle"];
```

Estilos e Comportamentos

Dicionário (Dictionary)

Você pode até mesmo recuperar um estilo de um dicionário de recursos separado em um arquivo no seu projeto (ou um assembly referenciado). Primeiro, você precisa criar um objeto ResourceDictionary e fornecer a URI correta:

Veja o arquivo de ResourceDictionary “**AlternateBigButtonStyle**” no slide a seguir:

Estilos e Comportamentos

AlternateBigButtonStyle.XAML

```
<ResourceDictionary
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">

    <Style x:Key="BigButtonStyle" TargetType="Button" >
        <Setter Property="FontFamily" Value="Georgia" />
        <Setter Property="FontSize" Value="40" />
        <Setter Property="Foreground" Value="SlateGray" />
        <Setter Property="Padding" Value="20" />
        <Setter Property="Margin" Value="10" />
    </Style>
    <Style x:Key="BackgroundButtonStyle2" TargetType="Button" >
        <Setter Property="Background">
            <Setter.Value>
                <LinearGradientBrush StartPoint="0,0" EndPoint="1,0">
                    <GradientStop Color="Blue"></GradientStop>
                    <GradientStop Color="Yellow" Offset="1"></GradientStop>
                </LinearGradientBrush>
            </Setter.Value>
        </Setter>
    </Style>
</ResourceDictionary>
```

Estilos e Comportamentos

Mudando o estilo do Butão a partir do dicionário.

```
ResourceDictionary resourceDictionary = new ResourceDictionary();
        resourceDictionary.Source = new Uri("/Styles/AlatenateBigButtonStyle.xaml",
            UriKind.Relative);

        Style newStyle = (Style)resourceDictionary["BigButtonStyle"];
        btnOK.Style = newStyle;
```

Estilos e Comportamentos

Herança de Estilos (Style Inheritance)

Em algumas situações, você pode querer usar as propriedades de um estilo como o básico para um outro, estilo mais especializados.

```
<UserControl.Resources>
        <Style x:Key="BigButtonStyle" TargetType="Button" >
            <Setter Property="FontFamily" Value="Georgia" />
            <Setter Property="FontSize" Value="40" />
            <Setter Property="Foreground" Value="SlateGray" />
            <Setter Property="Padding" Value="20" />
            <Setter Property="Margin" Value="10" />
        </Style>
        <Style x:Key="EmphasizeBigButtonStyle" TargetType="Button"
                BasedOn="{StaticResource BigButtonStyle}">
            <Setter Property="BorderBrush" Value="Black" />
            <Setter Property="BorderThickness" Value="5" />
        </Style>
    </UserControl.Resources>
```

Estilos e Comportamentos

Usando a herança de Estilos

```
<Button x:Name="btnNOK" Content="Cancelar" Height="90" Width="300"
            Style="{StaticResource EmphasizeBigButtonStyle}"
            Margin="50,210,50,0">
</Button>
```

Estilos e Comportamentos

Não é necessário que você defina o estilo de um elemento, com um recurso. Você pode definir as características de um elemento no próprio elemento. Veja o exemplo a seguir:

```
<Button Content="Um botão customizado" Height="70" Width="300">
        <Button.Style>
            <Style TargetType="Button" >
                <Setter Property="FontFamily" Value="Georgia" />
                <Setter Property="FontSize" Value="30" />
                <Setter Property="Foreground" Value="White" />
            </Style>
        </Button.Style>
</Button>
```