بررسی Resource ها در WPF و Silverlight

سیستم Resource در WPF راهی برای نگهداری و استفاده مجدد از یک مجموعه شی ، براش ها ،Template ها ،رنگ ها و... است.شما می توانید به راحتی Resource ها را در کد XAML مربوط به یک Window و یا به عنوان Resource کل برنامه تعریف کنید که با این عمل مقدار خیلی زیادی از کدنویسی کاهش می یابد همچنین امکان تغییر در ظاهر سیستم بسیار سهل می گردد.

پایه و اساس Resource :

این امکان را فراهم می کند که Resource ها را در کد و یا در قسمت های مختلف تعریف کنید به عنوان مثال Resource می تواند برای یک کنترل خاص و یا یک پنجره خاص و حتی برای یک برنامه خاص تعریف شود.

تا اینجا بیان نمودم که Resource می تواند برای هر کنترلی تعریف شود این به این دلیل است که هر FrameworkElement دارای پیاده سازی ضمنی از کلاس ResourceDictionary است. ResourceDictionary مجموعه ای از Resource ها را در خود نگه داری می کند که این Resourceها می توانند هر نوع شئ را در خود نگهداری کنند که بوسیله ویژگی Key که رشته است،ایندکس می شوند.

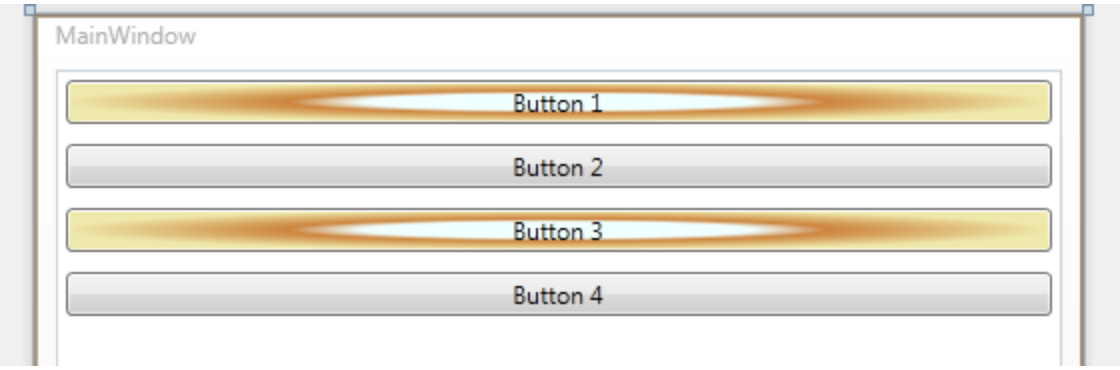
یک نمونه ساده تعریف و استفاده از Resource:

در این نمونه می خواهیم یک RadialGradientBrushتعریف کرده سپس آن را به خاصیت Backgroundچند دکمه نسبت دهیم.

کد:

```
<Window.Resources>
      <RadialGradientBrush  x:Key="btnGRBrush" >
         <GradientStop Color="Azure" Offset="0.4"/>
         <GradientStop Color="Peru" Offset="0.55"/>
         <GradientStop Color="PaleGoldenrod" Offset="1"/>
      </RadialGradientBrush>
   </Window.Resources>
   <StackPanel >
      <Button Background="{StaticResource btnGRBrush}" Margin="5">Button
1</Button>
      <Button  Margin="5">Button 2</Button>
      <Button Background="{StaticResource btnGRBrush}" Margin="5">Button
3</Button>
      <Button  Margin="5">Button 4</Button>
   </StackPanel>
```

دقت داشته باشید در کد XAML بالابرای دودکمه از RadialGradientBrush تعریف شده در Resource پنجره استفاده شده که خروجی نیز در تصویر زیر به نمایش در آمده است.



نحوه ی استفاده از Resource ها :

اگر دقت کرده باید در کد بالا وقتی براش RadialGradientBrush را به خاصیت Background نسبت می دادیم نوع اشاره ما به براش StaticResource بود، این به چه معنی است ؟

در هنگام اشاره به یک Object که در Resource تعریف شده است،دو نوع طریق اشاره وجود دارد:

- **StaticResource** :در این نوع **درهنگام ساخته شدن** پنجره ، Resource یک بار نسبت داده می شود.
- **DynamicResource** :در این نوع ،در **هنگام نیاز** شئ درون Resource نسبت داده می شود و در صورت تغییر خصوصیات کنترل های که به آن اشاره می کنند نیز تغییر می کنند .

بگذارید این نوع اشاره ها را با یک مثال بررسی کنیم.در این مثال نوع اشاره دکمه ۱ را به DynamicResourceتغییر می دهیم و با کلیک بر روی دکمه ۱ رنگ براش RadialGradientBrush تعریف شده در Resource را تغییر می دهیم تا نتیجه را مشاهده کنیم :

کد:

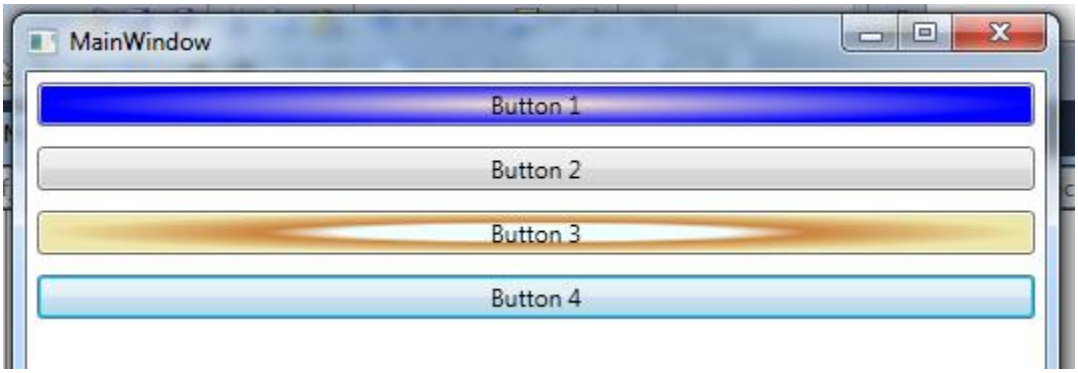
```
<Window.Resources>
        <RadialGradientBrush  x:Key="btnGRBrush" >
            <GradientStop Color="Azure" Offset="0.4"/>
            <GradientStop Color="Peru" Offset="0.55"/>
            <GradientStop Color="PaleGoldenrod" Offset="1"/>
        </RadialGradientBrush>
</Window.Resources>
    <StackPanel >
        <Button Background="{DynamicResource btnGRBrush}" Margin="5"
Click="Button_Click">Button 1</Button>
        <Button  Margin="5">Button 2</Button>
        <Button Background="{StaticResource btnGRBrush}" Margin="5">Button
3</Button>
        <Button  Margin="5">Button 4</Button>
    </StackPanel>
```

Code Behindبه صورت زیر :

کد:

```
private void Button_Click(object sender, RoutedEventArgs e)
        {
            this.Resources["btnGRBrush"] = new
RadialGradientBrush(Colors.BlanchedAlmond ,Colors.Blue);
        }
```

با اینکار خروجی به صورت زیر تغییر میکند که تفاوت بین DynamicResource و StaticResource به نمایش در می آید.



نکته مهم :توجه داشته باشید که در هنگام استفاده از این نوع اشاره ها دقت کنید چون StaticResourceها در هنگام ساخته شدن فرم ساخته شده و نسبت داده شده می شود ولی DynamicResourceتا هنگام نیاز ساخته نمی شوند.سعی کنید از StaticResource هنگامی استفاده کنید که هیچ تغییری در آن ها بوجود نمی آید و از DynamicResource هنگامی که در هنگام اجرا تغییر می کنند (به عنوان مثال از دینامیک برای تنظیمات برنامه ای که با تغییر خصوصیات در همان لحظه تغییر ها اعمال می شوند و از استاتیک ها زمانی که با تغییر تنظیمات برنامه نیاز به ریستارت برنامه برای اعمال تغییر می باشد.) .

خاصیت Shared :

وقتی که شما از چندین کنترل به یک Resource اشاره می کنید در واقع شما به یک نمونه از Resource اشاره می کنید که دلیل آن true بودن خاصیت Shared به صورت پیش فرض است ولی اگر بنا به دلایلی می خواهید که در هنگام اشاره هر کنترل به resource یک نمونه جدید به آن نسبت داده شود این خاصیت را با False مقداردهی کنید

کد:

```
<Window.Resources>
    <RadialGradientBrush  x:Key="btnGRBrush" x:Shared="False" >
      <GradientStop Color="Azure" Offset="0.4"/>
      <GradientStop Color="Peru" Offset="0.55"/>
      <GradientStop Color="PaleGoldenrod" Offset="1"/>
    </RadialGradientBrush>
  </Window.Resources>
```

نکته : با این کار در عمل مزایای DynamicResource از بین می رود.

نحوه ی تعریف و استفاده از Application Resource:

تعریف Resource ها در هر فرم کار عاقلانه ای نیست برای همین در اکثریت مواقع Resource ها را در بالاترین سطح برنامه یعنی Application تعریف می کنیم.هنگامی که به یک Resource اشاره می کنیم اگر آنResource در کنترل ،Window ویا Page جاری وجود نداشته باشد،Resource های Application جستجو می شود که بهترین قسمت برای Resourceهای است که در کل برنامه مورد استفاده قرار می گیرند.

برای این کار Resource را در فایل App.xaml تعریف می کنیم به صورت زیر

کد:

```
<Application x:Class="wpf_resources.App"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        StartupUri="MainWindow.xaml">
```

```
    <Application.Resources>
        <!-- define resource Here-->
    </Application.Resources>
</Application>
```

: Resources System

همان تنظیمات سیستم است که ما از آنها استفاده می کنیم در قبل گفتیم که وقتی Resource ی پیدا نشود به Application Resource مراجعه می گردد و در صورت نبود به System Resource مراجعه می شود.این Resourceها در قالب سه کلاس هستند که در فضای نام System.Windows قرار دارند.

- SystemColors ;:دستیابی به تنظیمات رنگ سیستم را فراهم می کند.
- SystemFonts :دستیابی به تنظیمات قلم سیستم را فراهم می کند.
- SystemParameters :دستیابی به طیف عظیمی از تنظیمات سیستم را فراهم می کند از جمله تنظیمات پنجره ها و سایز آن ،موس و کیبورد و...گرفته تا جلوه های نمایش سیستم.

نکته :خصوصیات همه این کلاس ها از نوع ایستا هستند و در هنگام اشاره به آن می بایست به آن از نوع ایستا اشاره کنیم.

در مثال زیر از هر سه کلاس استفاده شده است

کد:

```
<!-- SystemFonts -->
```

```
        <Button  Margin="5"
                FontSize="{x:Static SystemFonts.IconFontSize}"
                FontWeight="{x:Static SystemFonts.MessageFontWeight}"
                FontFamily="{x:Static SystemFonts.CaptionFontFamily}">Button
2</Button>

        <!-- SystemColors-->
        <Button  Margin="5"
                BorderBrush="{x:Static SystemColors.ActiveBorderBrush}"
                Background="{x:Static SystemColors.HotTrackBrush}"
                Foreground="{x:Static SystemColors.HighlightTextBrush}"
                >Button 4</Button>

        <!-- SystemParameters -->
        <Button  Margin="5"
                Width="{x:Static SystemParameters.MenuWidth}"
                Height="{x:Static SystemParameters.IconGridWidth}">

        </Button>
```

: Resource Dictionaries

وقتی بخواهید که Resource های خود در پروژه های دیگر استفاده کنید و یا حتی آنها را از هم جدا کنید تا در صورت نیاز به راحتی از آنها استفاده کنید برای این منظور از Resource Dictionary استفاده می کنیم.Resource Dictionary یک فایل XAML ساده است که کاری

انجام نمی دهد بلکه Resource ها را در خود نگهداری می کند.

ساخت یک Resource Dictionary ساده :

برای این منظور از منوی Project سپس Add New item کلیک کرده سپس از پنجره باز شده " Dictionary Reource " را انتخاب می کنیم.فایل XAML جدیدی با محتوای زیر باز می شود.

کد:

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



</ResourceDictionary>
```

سپس Resource های خود را در آن می نویسیم و برای استفاده از آن بسته به نیاز خود در کنترل ،پنجره و یا برنامه از آن به صورت زیر استفاده می کنیم.

در اینجا از Resource Dictionary در Application Resource استفاده شده است:

کد:

```
<Application x:Class="wpf_resources.App"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    StartupUri="MainWindow.xaml">
```

```
    <Application.Resources>
        <ResourceDictionary Source="WhistlerBlue.xaml"/>
    </Application.Resources>
</Application>
```

نکته : توجه داشته باشید خاصیت Build Action فایل Dictionary حتما با Page تنظیم شده باشد Performance بالای داشته باشیم

نکته : اگر در برنامه خورد چندین Resource Dictionary داشته باشیم که حتما همینطور است از روش زیر برای استفاده از آنها استفاده می کنیم

کد:

```
<Application x:Class="wpf_resources.App"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             StartupUri="MainWindow.xaml">
    <Application.Resources>
        <ResourceDictionary>
            <ResourceDictionary.MergedDictionaries>
                <ResourceDictionary Source="WhistlerBlue.xaml"/>
            </ResourceDictionary.MergedDictionaries>
        </ResourceDictionary>
    </Application.Resources>
</Application>
```
