# WPF之对象资源

在cs文件中，我们想使用.NET对象，直接new即可。在XAML中WPF元素可以很灵活创建，但是其他对象就显得很掣肘。

WPF元素都有一个资源字典，即Resources属性(类型是ResourceDictionary)，字典中可以创建和存放任何.NET对象。可以利用这一点，我们在XAML中存放我们想要使用的.NET对象，然后在XAML其他部分通过字符串索引使用对象。

x:key用于生成资源字典中的索引。

可在资源字典中实例化任何.NET对象， 只要该类是XAML友好的即可。这意味着该类要具有一些基本特性，如公有的无参构造函数和可写的属性。

<Window x:Class="ObjectResources.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:ObjectResources"

xmlns:sys="clr-namespace:System;assembly=mscorlib"

mc:Ignorable="d"

Title="MainWindow" Height="450" Width="800">

<Window.Resources>

<sys:String x:Key="str">我是资源字符串</sys:String>

<sys:Double x:Key="num">3.14159</sys:Double>

<SolidColorBrush x:Key="brush" Color="Black"/>

<Style TargetType="{x:Type Button}">

<Setter Property="Background" Value="Red"/>

<Setter Property="RenderTransform">

<Setter.Value>

<RotateTransform Angle="45"/>

</Setter.Value>

</Setter>

</Style>

</Window.Resources>

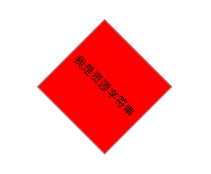
<Grid>

<Button Height="100" Width="100" Content="{StaticResource str}"/>

</Grid>

</Window>

效果图



## 使用资源

使用StaticResource或DynamicResource，静态资源和动态资源的主要区别是：静态资源第一次被使用时，从资源字典中找到对象资源，此后便便一直使用，后续不会再从资源字典中查找。动态资源每次被使用时，都会去资源字典中临时查找对象资源。

那么，当我们改变资源字典中某个key对应的资源时，动态资源会更新，而静态资源仍旧使用之前的未被替换前的对象资源。

<Window.Resources>

<sys:String x:Key="str">我是资源字符串</sys:String>

<sys:Double x:Key="num">3.14159</sys:Double>

<SolidColorBrush x:Key="brush" Color="Black"/>

<Style TargetType="{x:Type Button}">

<Setter Property="Background" Value="Red"/>

<Setter Property="RenderTransform">

<Setter.Value>

<RotateTransform Angle="45"/>

</Setter.Value>

</Setter>

</Style>

</Window.Resources>

<StackPanel>

<Button Height="100" Width="100" Content="{DynamicResource str}"/>

<Button Style="{x:Null}" Margin="50" Height="50" Content="点击我更新字典资源中的str" Click="Button\_Click"/>

</StackPanel>

</Window>

private void Button\_Click(object sender, RoutedEventArgs e)

{

this.Resources["str"] = "我被替换了";

}



gif动图

如果我们将XAML中的DynamicResource改成StaticResource红色Button上的文字不会发生变化。

## 误区

class Person

{

public string Name { get; set; }

public int Age { get; set; }

public override string ToString()

{

return Name + Age.ToString();

}

}

<Window.Resources>

<local:Person x:Key="person" Name="杨宗纬" Age="40" />

<sys:String x:Key="str">字符串</sys:String>

<SolidColorBrush x:Key="brush" Color="Red"/>

</Window.Resources>

<StackPanel>

<Button Content="{StaticResource person}" Background="{StaticResource brush}" Height="100"/>

<Button Content="{DynamicResource person}" Background="{DynamicResource brush}" Height="100"/>

<Button Content="点击我修改资源" Click="Button\_Click"/>

<Button Content="点击我替换资源" Click="Button\_Click\_1"/>

</StackPanel>

private void Button\_Click(object sender, RoutedEventArgs e)

{

(this.FindResource("person") as Person).Name = "张无忌";

(this.FindResource("brush") as SolidColorBrush).Color = Colors.Gray;

}

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

this.Resources["person"] = new Person { Name = "张无忌", Age = 45 };

this.Resources["brush"] = new SolidColorBrush(Colors.Gray);

}

修改资源效果图

字符串静态资源未变化，画刷静态资源变化

字符串动态资源未变化，画刷动态资源变化



替换资源效果图

字符串静态资源未变化，画刷静态资源未变化

字符串动态资源变化，画刷动态资源变化



根据现象总结：

替换资源，静态资源不变化，动态资源一定变化

修改资源，静态和动态资源是否变化，取决于资源对象有没有继承Freezable类。画刷继承，而自定义的Person未继承。

## 资源字典

WPF的Resources属性类型是ResourceDictionary，支持如下几种赋值方式。

<Window.Resources>

<ResourceDictionary>

<sys:Double x:Key="num">3.14159</sys:Double>

<SolidColorBrush x:Key="brush" Color="Black"/>

</ResourceDictionary>

</Window.Resources>

MergedDictionaries是个ResourceDictionary集合，会提取所有字典元素的key到自己的字典中。

<Window.Resources>

<ResourceDictionary>

<ResourceDictionary.MergedDictionaries>

<ResourceDictionary/>

<ResourceDictionary/>

</ResourceDictionary.MergedDictionaries>

</ResourceDictionary>

</Window.Resources>

xaml是资源，build action设置成page，xaml会被编译成BAML,性能会好点，当然也可以设置成Resource。

写法遵从URI。

<Window.Resources>

<ResourceDictionary Source="Dictionary1.xaml"/>

</Window.Resources>

## 查找资源的顺序

从自己的资源开始查找，查不到就查逻辑树上级元素的资源，一直查到Application.Resources，找到资源后便不会继续找到，找到Application.Resources还找不到，就会抛出异常。

同一个资源字典，允许添加多个key相同的资源，只是查找资源时，只能查到到最后一个被添加的资源。

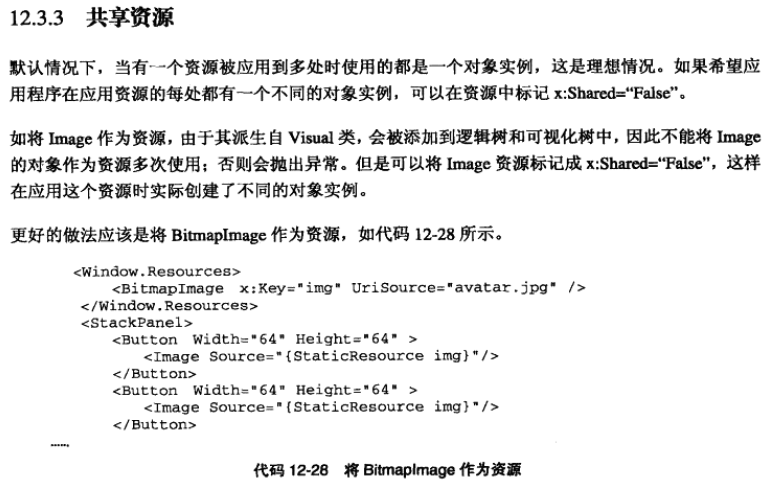
## 后台操作资源字典的方法

string text = (string)this.FinadResources(“str”);

从当前控件查找，逐级向上查找。

string text = this.Resources[“str”];

在指定的控件的的资源字典查找资源。



系统资源

替换资源

资源字典管理资源

# 本地化

使用卫星程序集X:static resource.getstring()