Contenido

Estilos en WPF

Los estilos en Windows Presentation Foundation (WPF) son una herramienta que nos permite establecer la apariencia y comportamiento de los elementos de la interfaz de usuario. Pueden ser aplicados a nivel de aplicación, ventana o incluso a elementos individuales o contenedores específicos. A continuación, se explica cómo funcionan los estilos en diferentes ámbitos y las diferencias entre aplicar estilos a nivel global, de ventana o de contenedor.

1. Conceptos Básicos

1.1 Definición de Estilos

En WPF, un estilo es un conjunto de propiedades que se pueden aplicar a un tipo específico de elemento o a un conjunto de elementos. Los estilos proporcionan coherencia visual y simplifican la gestión de la apariencia de la interfaz de usuario.

1.2 Estructura de un Estilo

Un estilo consta de:

- ✓ *Selector:* Define a qué tipo de elemento o conjunto de elementos se aplicará el estilo.
- ✓ *Propiedades:* Especifica las propiedades que se establecerán para los elementos seleccionados.
- ✓ *x:Key:* Un identificador único que se utiliza para referenciar el estilo y aplicarlo selectivamente a los elementos.

2. Aplicación de Estilos en WPF

2.1 Estilos Globales

En el archivo *App.xaml*, puedes definir estilos globales que se aplicarán a lo largo de toda la aplicación. Estos estilos afectan a todos los elementos del tipo especificado.

Para aplicar estilos globales a nuestro proyecto, lo hacemos entre las etiquetas ***<Application>*** del archivo *App.xaml*. Como vemos en la siguiente imagen, ponemos el atributo ***TargetType*** para establecer el componente al que se le aplicará el estilo. Mediante la etiqueta ***<Setter>*** establecemos las propiedades que queremos modificar utilizando ***Property*** y con la propiedad ***Value***, establecemos el valor a modificar.

```
<Application.Resources>
    <Style TargetType="Button">
        <Setter Property="Background" Value="AliceBlue"/>
        <Setter Property="FontSize" Value="14"/>
    </Style>
</Application.Resources>
```

De esta forma hemos aplicado estilos a todos los controles de tipo botón que tengamos en nuestra aplicación. Hemos aplicado un cambio del color de fondo y un tamaño de fuente de 14.

2.2 Estilos a Nivel de Ventana

Puedes definir estilos específicos para una ventana en particular en el archivo *Window.xaml* o en el código de la ventana.

Como vemos en la siguiente imagen, hemos aplicado los estilos a nivel de ventana. De esta manara estos estilos solo afectarán a la ventana donde estén definidos. En esta ocasión, he aplicado estilos utilizando el identificador ***x:Key="LocalLabelStyle"*** para aplicar este estilo únicamente a las etiquetas donde apliquemos este identificador.

```
<Window.Resources>
    <Style TargetType="Label" x:Key="LocalLabelStyle">
        <Setter Property="Background" Value="Red"/>
        <Setter Property="HorizontalAlignment" Value="Center"/>
        <Setter Property="VerticalAlignment" Value="Center"/>
    </Style>
</Window.Resources>
```

Para aplicar ese estilo a la etiqueta que deseemos bastaría con hacer esto:

```
<Label Content="Label 1" Style="{StaticResource LocalLabelStyle}"></Label>
```

De esta forma aplicamos ese estilo a una única etiqueta. Si tenemos varias ***label*** en nuestra ventana, de esta forma solo lo aplicaremos a las que nosotros deseemos.

2.3. Estilos a Nivel de Contenedor

Puedes aplicar estilos a un contenedor específico, como un ***Grid*** o ***StackPanel***. Los estilos definidos en el nivel del contenedor afectarán a los elementos secundarios contenidos en ese contenedor.

En la siguiente imagen vemos como aplicar estilos a nivel de contenedor:

```
<Grid.Resources>
    <Style TargetType="TextBlock">
        <Setter Property="Foreground" Value="Purple" />
        <Setter Property="FontSize" Value="34" />
    </Style>
</Grid.Resources>
```

En esta ocasión, he aplicado los estilos a los controles de tipo ***TextBlock*** sin utilizar el identificador ***x:Key*** por lo que se aplicará a todos los ***TextBlock*** que haya en el ***Grid***.

2.4. Aplicación de Estilos con x:Key

Aunque lo hemos visto en el apartado 2.2, lo explico en este apartado de nuevo. Podemos aplicar estilos a controles individuales mediante la utilización del identificador ***x:Key*** en la etiqueta ***<Style>***.

```
<Application.Resources>
    <Style TargetType="Button" x:Key="ButtonMain">
        <Setter Property="Background" Value="AliceBlue"/>
        <Setter Property="FontSize" Value="14"/>
    </Style>
</Application.Resources>
```

Después, podemos aplicar el estilo a nuestro control de la siguiente forma:

```
<Button Content="Botón 1" Style="{StaticResource ButtonMain}"/>
```

2.5. Aplicación de estilos mediante código C#

También podemos aplicar estilos si estamos utilizando *code-behind*. Es decir, si estamos creando controles mediante código en C#, podemos aplicar los estilos también. Por ejemplo, en este código vemos como aplico los estilos que he definido en *App.xaml* a un botón en código:

```
myButton.Style = (Style)FindResource("ButtonMain");
```

3. Consideraciones Finales

- ✓ La propiedad ***x:Key*** es esencial para identificar y aplicar estilos de manera selectiva.
- ✓ La jerarquía de estilos es importante. Si se definen estilos en varios niveles (global, ventana, contenedor, elemento), el estilo más específico tiene prioridad.
- ✓ Los estilos pueden contener desencadenadores (Triggers) que responden a eventos o cambios de estado.

Recuerda utilizar estilos para mejorar la consistencia visual y simplificar la gestión del diseño en tu aplicación WPF.