Windows Presentation Foundation - WPF

UT5. GENERACIÓN DE INTERFACES A TRAVES DE XML

Características

Se introdujo con Windows Vista.

Toma características de las aplicaciones de escritorio como de las aplicaciones web. => Sí, se puede visualizar en un navegador.

Amplia las posibilidades de WinForms, soporte para multimedia, documentos enriquecidos, animaciones, etc…

Separa el código del Frontend (XAML) del Backend (.NET).

Favorece un modelo desarrollo MVC. => Mucha bibliografía emplea MVVC.

Emplea Direct3D (GPU) para la presentación descargando a la CPU.

Gráficos vectoriales, escalado sin perdida.

XAML

Acrónimo de “eXtensible Application Markup Language”.

Lenguaje declarativo basado en XML.

Permite definir estilos.

Aunque se parezca al HTML en este caso se diferencian MAYUSCULAS de minúsculas.

Al compilar se generan fichero binarios “.baml” y se incrustan en el ensamblado.

XAML

Las etiquetas se cierran con “/”. En el caso de incluir contenido (otras etiquetas) tendremos una etiqueta de apertura y otra de cierre.

```
<Button FontWeight=”Bold” Content=”Texto del botón” />
```

Es equivalente a

```
<Button>
    <Button.FontWeight>Bold</Button.FontWeight>
    <Button.Content>Texto del botón</Button.Content>
</Button>
```

XAML

Los controles son ahora etiquetas con atributos existiendo una relación directa con las clases y propiedades de la API de Windows.

```
<Button>Pulsar</Button>
```

Es equivalente a

```
Button btn = new Button();

btn.Content = "Pulsar";

this.pnlPrincipal.Children.Add(btn);
```

XAML

Clases base

- FrameworkElement
  - Base para todo los controles del layout.
  - Añade funcionalidad para el árbol lógico, estilos, enlace a datos, eventos.
  - Hereda de UIElement.
- ContentControl
  - Define un control con un único contenido. => Es decir, todos los controles que usemos para representar un datos tienen esta estructura.
  - Hereda de FrameworkElement.
- UIElement
  - Clase básica de WPF.

Estructura de una aplicación WPF

Al crea un nuevo proyecto de aplicación WPF se crean 3 ficheros:

- App.config
  - Incluye parámetros de configuración de la aplicación. Versión del framework, cadenas de conexión, módulos de terceros, parámetros de definición propia, etc…
- App.xaml
  - Permite definir el punto de arranque de la aplicación. OJO: Ventana o código.
  - Lugar en el que definir los estilos/recursos globales. “Application.Resources”.
  - Gestión última de las excepciones no controladas.
- MainWindow.xaml
  - Ventana inicial de la aplicación.

Cada fichero .xaml tiene un fichero con el código de respaldo .xaml.cs

Control Window

Esta clase es el equivalente a la clase Form de WinForms.

Hereda de **ContentControl** por lo que tendremos un único control hijo que será un panel. Por defecto Grid.

x:Class define el fichero de código de respaldo.

xmlns:VARIABLE define espacios de nombres, habitualmente trabajaremos con:

- x: espacio de nombres de WPF.
- local: espacio de nombres propio de la ventana, permite referenciar al código del proyecto.
- d: espacio de nombres para las herramientas de diseño, solo aplica en tiempo de diseño.

Control Window - XAML

```
<Window x:Class="WpfHolaMundo.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:WpfHolaMundo"
        mc:Ignorable="d"
        Title="MainWindow" Height="450" Width="800">
    <Grid>

    </Grid>
</Window>
```

Estructura Windows – Código de respaldo

Comportamiento similar al de WinForms, aquí tendré el código de respaldo, la gestión de eventos, etc…

Para poder referenciar a una etiqueta XAML está tiene que tener un nombre. Esto aplica también a la ventana. => Propiedad “Name”

```
public partial class MainWindow : Window
{
        public MainWindow()
        {
                InitializeComponent();
        }
}
```

Consideraciones

La caja de propiedades ya no es tan útil como en WinForms, la forma rápida de trabajar es la edición manual del XAML.

- Al contener etiquetas unas dentro de otras, y al existir referencias a la etiqueta padre, la caja de propiedades ya no es capaz de gestionar todos los aspectos del control.

El editor XAML ayuda con los nombre de las etiquetas, úsalo.

Algunos errores en el editor no desaparecen hasta que se recompila el proyecto.