Introducción a la programación con C#

Interfaz gráfica de usuario con WPF

Un poco de repaso...

¿Qué es la herencia? Y ¿cuáles son los beneficios de un correcto uso?

Un poco de repaso...

¿Cuál es la diferencia entre **herencia extensión** y **herencia por implementación**?

¿Qué es una interfaz?

Según WordReference:

- Zona de comunicación o acción de un sistema sobre otro.
- Dispositivo que conecta dos aparatos o circuitos.
- Dispositivo capaz de transformar las señales emitidas por un aparato en señales comprensibles por otro.

¿Y una interfaz gráfica de usuario?

La interfaz de usuario **(UI)** es el punto de interacción y comunicación entre una persona y una computadora.

También es la forma en que un usuario interactúa con una aplicación o un sitio web.

e**X**tensible **M**arkup **L**anguage

Lenguaje de Marcas Extensible.

Es un **metalenguaje** que permite definir lenguajes de marcas utilizado para almacenar datos en forma legible.

Permite definir la gramática de lenguajes específicos para estructurar documentos grandes. También da soporte a bases de datos, siendo útil cuando varias aplicaciones deben comunicarse entre sí o integrar información.

e**X**tensible **A**pplication **M**arkup **L**anguage

Es un lenguaje declarativo basado en XML, optimizado para describir gráficamente interfaces de usuario visuales ricas desde el punto de vista gráfico.

```
<Window
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    Title="Window with Button"
    Width="250" Height="100">

  <!-- Add button to window -->
  <Button Name="button">Click Me!</Button>

</Window>
```

Eventos

Los Eventos son las acciones sobre el programa.

Como por ejemplo:

- Clic sobre un botón.
- Doble clic sobre el nombre de un fichero para abrirlo.
- Pulsar una tecla o una combinación de teclas.

Windows Presentation Foundation

Es un marco de **interfaz de usuario** que crea aplicaciones cliente de **escritorio**.

Admite un conjunto de características de desarrollo de aplicaciones, incluido un modelo de aplicación, recursos, controles, gráficos, diseños, enlace de datos, documentos y seguridad.

Utiliza el lenguaje **XAML** para proporcionar un modelo declarativo para la programación de aplicaciones.

Documentación oficial

Clase de código subyacente

El comportamiento principal de una aplicación es implementar la funcionalidad que responde a las interacciones del usuario.

Por ejemplo, hacer clic en un un botón y llamar a la lógica de negocios y la lógica de acceso a datos como respuesta.

En WPF, este comportamiento se implementa en el código que está asociado al marcado. Este tipo de código se conoce como **código subyacente**.

Uso de **partial** class

```xaml
<Window
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    x:Class="SDKSample.AWindow"
    Title="Window with Button"
    Width="250" Height="100">

  <!-- Add button to window -->
  <Button Name="button" Click="button_Click">Click Me!</

</Window>
```

```xaml.cs
using System.Windows;

namespace SDKSample
{
    public partial class AWindow : Window
    {
        public AWindow()
        {
            // InitializeComponent call is required to merge the UI
            // that is defined in markup with this class, including
            // setting properties and registering event handlers
            InitializeComponent();
        }

        void button_Click(object sender, RoutedEventArgs e)
        {
            // Show message box when button is clicked.
            MessageBox.Show("Hello, Windows Presentation Foundation!");
        }
    }
}
```

Controles **WPF** por función

- Botones: Button y RepeatButton.
- Presentación de datos: DataGrid, ListView y TreeView.
- Presentación y selección de fechas: Calendar y DatePicker.
- Cuadros de diálogo: OpenFileDialog, PrintDialogy SaveFileDialog.
- Entrada de lápiz digital: InkCanvas y InkPresenter.
- Documentos: DocumentViewer, FlowDocumentPageViewer, FlowDocumentReader, FlowDocumentScrollViewery StickyNoteControl.
- Entrada: TextBox, RichTextBoxy PasswordBox.

Controles **WPF** por función

- Diseño: Border, BulletDecorator, Canvas, DockPanel, Expander, Grid, GridView, GridSplitter, GroupBox, Panel, ResizeGrip, Separator, ScrollBar, ScrollViewer, StackPanel, Thumb, Viewbox, VirtualizingStackPanel, Windowy WrapPanel.
- Multimedia: Image, MediaElementy SoundPlayerAction.
- Menús: ContextMenu, Menuy ToolBar.
- Navegación: Frame, Hyperlink, Page, NavigationWindowy TabControl.
- Selección: CheckBox, ComboBox, ListBox, RadioButtony Slider.
- Información del usuario: AccessText, Label, Popup, ProgressBar, StatusBar, TextBlocky ToolTip.

Documentación oficial

Controles de diseño

Al crear crear una interfaz de usuario, se organizan los controles según su ubicación y tamaño para crear un diseño.

Un requisito fundamental de cualquier diseño es adaptarse a los cambios de tamaño de la ventana y de configuración de pantalla.

WPF proporciona un sistema de diseño extensible para darle solución.

Controles de diseño

- **Canvas**: los controles secundarios proporcionan su propio diseño.
- **DockPanel**: los controles secundarios se alinean con los bordes del panel.
- **Grid**: los controles secundarios se colocan por filas y columnas.
- **StackPanel**: los controles secundarios se apilan vertical u horizontalmente.
- **VirtualizingStackPanel**: los controles secundarios se virtualizan y se organizan en una sola línea en sentido horizontal o verticalmente.
- **WrapPanel**: los controles secundarios se sitúan en orden de izquierda a derecha y se ajustan a la línea siguiente cuando no hay suficiente espacio en la línea actual.

¡¡A trabajar!!