



XUNTA DE GALICIA

CONSELLERÍA DE CULTURA, EDUCACIÓN  
E ORDENACIÓN UNIVERSITARIA



UNIÓN EUROPEA

Fondo Social Europeo  
*O FSE inviste no teu futuro*



Centros  
Integrados  
Formación Profesional

# Unidad 2

## Generación de interfaces gráficas de usuario utilizando el lenguaje XML

Módulo Desarrollo de Interfaces

Ciclo Superior Desarrollo de aplicaciones multiplataforma

Curso 2015-2016



# Contenido

1. Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación
2. Elementos, etiquetas, atributos y valores
3. Herramientas libres y propietarias para la creación de interfaces de usuario multiplataforma
4. Controles: propiedades
5. Eventos: controladores
6. Edición y depuración del documento XML
7. Generación de código para diferentes plataformas

# **2.1. Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación**

Unidad 2: Generación de interfaces gráficas de usuario utilizando el lenguaje XML

3

## 2.1. Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación



- **XML** (*Extensible Markup Language*) es un subconjunto del **SGML** (Standard Generalized Markup Language).
- XML es un metalenguaje con el que se pueden definir otros lenguajes de etiquetas.
- Los documentos XML tienen formato de texto.
- Desde febrero de 1998 es una recomendación del W3C (World Wide Web Consortium).

# Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación



- Un **lenguaje de marcado** o **lenguaje de marcas** es una forma de codificar un documento que, junto con el texto, incorpora etiquetas o marcas que contienen información adicional acerca de la estructura del texto o su presentación.
- A veces se confunde lenguaje de marcas con lenguaje de programación y es un error ya que el primero no contiene ni variables, expresiones aritméticas ni sentencias del tipo condicional o iterativo.

# Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación



- Entre los lenguajes de marcas o de descripción más importantes y de gran influencia en el campo que nos ocupa, la generación de interfaces gráficas, se encuentra el **XML**.
- XML no ha nacido sólo para su aplicación en Internet, sino que se propone como un **estándar para el intercambio de información estructurada** entre diferentes plataformas.
- Se puede usar en bases de datos, editores de texto, hojas de cálculo e interfaces gráficas, entre otras...

# Ejemplo de documento xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE Edit_Mensaje SYSTEM "Edit_Mensaje.dtd">

<Edit_Mensaje>
  <Mensaje>
    <Remitente>
      <Nombre>Nombre del remitente</Nombre>
      <Mail>Correo del remitente </Mail>
    </Remitente>
    <Destinatario>
      <Nombre>Nombre del destinatario</Nombre>
      <Mail>Correo del destinatario</Mail>
    </Destinatario>
    <Texto>
      <Asunto>
        Este es mi documento con una estructura muy sencilla
        no contiene atributos ni entidades...
      </Asunto>
      <Parrafo>
        Este es mi documento con una estructura muy sencilla
        no contiene atributos ni entidades...
      </Parrafo>
    </Texto>
  </Mensaje>
</Edit_Mensaje>
```




# Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación



- Entre los lenguajes de descripción de interfaces basados en XML más importantes (*no están todos ya que algunos aún se están desarrollando*) tenemos:
  - **UIML** (*User Interface Markup Language*)
  - **XIML** (*eXtensible Interface Markup Language*)
  - **MXML** (*Macromedia*)
  - **GladeXML-GtkBuilder**
  - **XAML** (*eXtensible Application Markup Language*)
  - **Xforms**
  - **XUL** (*XML-based User-interface Language*)
  - **XBL** (*eXtensible Bindings Language*)

```
<?xml version="1.0"?>
<quiz>
  <question>
    Who was the forty-second
    president of the U.S.A.?
  </question>
  <answer>
    William Jefferson Clinton
  </answer>
  <!-- Note: We need to add
    more questions later.-->
</quiz>
```





# Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación

## **UIML (*User Interface Markup Language*)**



- Es un sencillo lenguaje basado en XML que permite realizar una descripción declarativa de la interfaz de usuario de un modo independiente del dispositivo.
- *Para describir una interfaz de usuario en UIML se debe realizar, por un lado la definición de la interfaz genérica, y por otro un documento UIML que representa el estilo de presentación apropiado para el dispositivo en el cual la interfaz de usuario se va a ejecutar.*
- De este modo, una misma aplicación solamente necesitará un único documento UIML de especificación válido para cualquier dispositivo y un documento de estilo propio para cada dispositivo.

# Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación

## ***UIML (User Interface Markup Language)***



Algunas direcciones de consulta:

- <http://en.wikipedia.org/wiki/UIML>
- [https://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=uiml](https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=uiml)
- [http://catarina.udlap.mx/u\\_dl\\_a/tales/documentos/lis/aragon\\_ch/capitulo2.pdf](http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/aragon_ch/capitulo2.pdf)

# Lenguajes de descripción de interfaces basadas

## en XML: ámbito de aplicación

### ***XIML (eXtensible Interface Markup Language)***



- Es un lenguaje de especificación basado en XML.
- Se propone como lenguaje de especificación común e infraestructura de desarrollo para profesionales de la interfaz de usuario en todos los ámbitos, diseñadores, ingenieros de software o expertos en usabilidad.
- <http://www.ximl.org/>

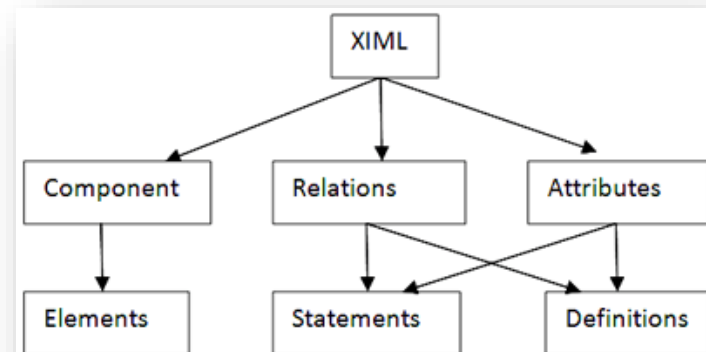


Figure. 1.0 Basic Structure of XIML

# Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación

## **MXML (*Macromedia eXtensible Markup Language*)**

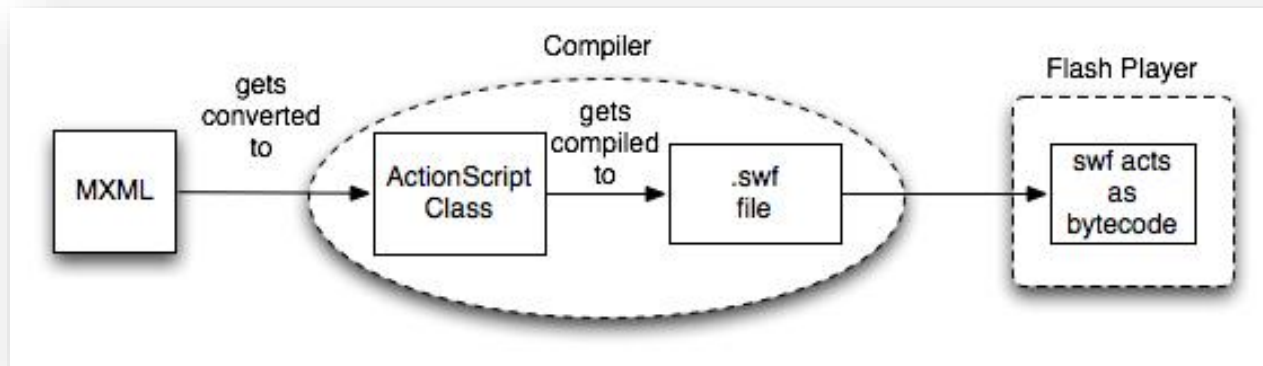


- Es un lenguaje descriptivo desarrollado inicialmente por Macromedia hasta el 2005 para la plataforma FLEX de Adobe.
- Es un lenguaje que describe interfaces de usuario, crea modelos de datos y tiene acceso a los recursos del servidor, del tipo RIA (Rich Internet Application).
- MXML tiene una mayor estructura en base a etiquetas, similar a HTML, pero con una sintaxis menos ambigua, proporciona una gran variedad e inclusive permite extender etiquetas y crear sus propios componentes.
- Una vez compilado genera ficheros .swf

# Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación **MXML (*Macromedia*)**



- <http://es.wikipedia.org/wiki/MXML>
- <https://learn.adobe.com/wiki/display/Flex/MXML>



# Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación **GladeXML-GtkBuilder**

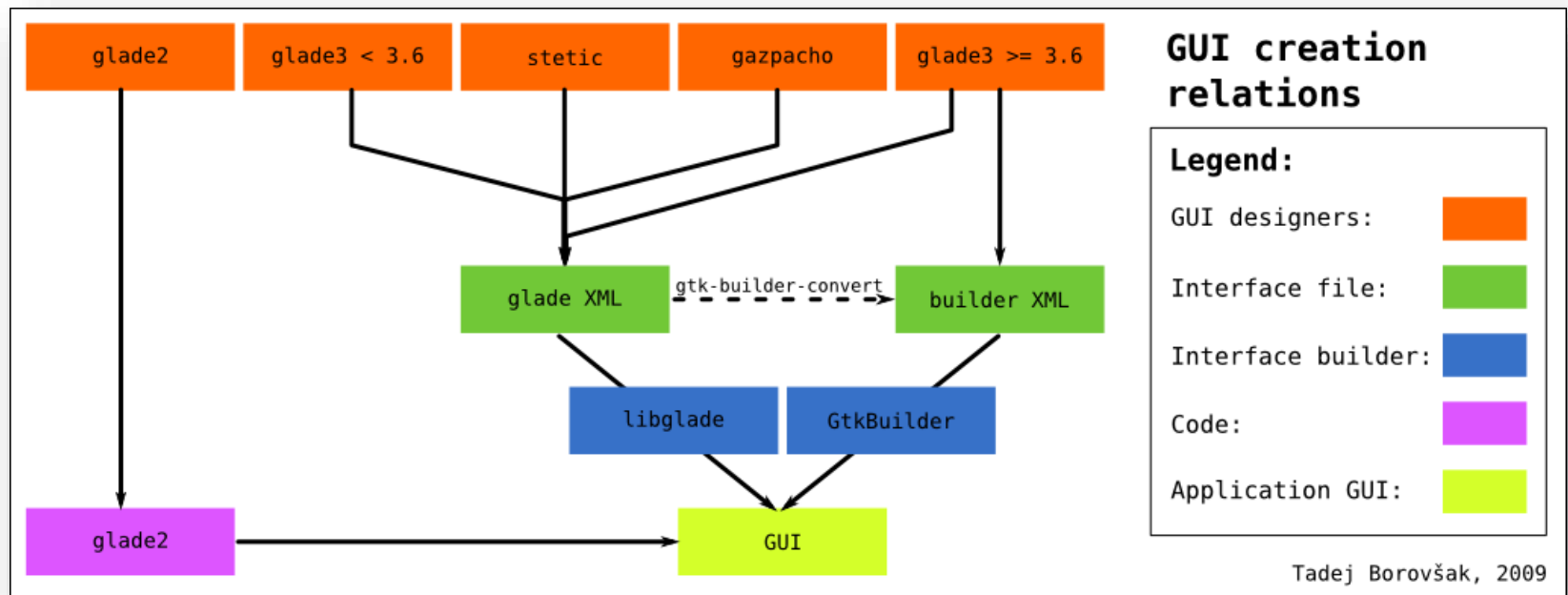


- Es una herramienta de desarrollo visual de interfaces gráficas mediante GTK/GNOME.
- Es independiente del lenguaje de programación y no genera código fuente sino un archivo **XML**.
- El **IDE Anjuta** lo lleva integrado para el desarrollo de interfaces pero puede trabajar independientemente.
- Actualmente está en la versión Glade3 que fue rescrita totalmente aumentando el número de widgets y haciéndolo más ligero.
- **GtkBuilder** es un formato XML que Glade usa para almacenar los elementos de las interfaces diseñadas.
- Estos archivos pueden emplearse para construirlas en tiempo de ejecución mediante el objeto GtkBuilder de GTK+.
- **GladeXML** era el formato que se usaba en conjunto con la biblioteca *libglade* (ambos obsoletos en favor de GtkBuilder).

# Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación **GladeXML-GtkBuilder**



- <http://developer.gnome.org/libglade/unstable/GladeXML.html>
- <http://developer.gnome.org/gtk/2.24/GtkBuilder.html>



Tadej Borovšak, 2009



# Lenguajes de descripción de interfaces basadas en XML:

## ámbito de aplicación

### **XAML** (*eXtensible Application MarkupLanguage*)



- Es el lenguaje de formato para la interfaz de usuario para la **Base de Presentación de Windows y Silverlight**, el cual es uno de los "pilares" de la interfaz de programación de aplicaciones .NET en su versión 3.0 (conocida con anterioridad con el nombre clave WinFX).
- XAML es un lenguaje declarativo basado en XML, optimizado para describir gráficamente interfaces de usuarios visuales ricas desde el punto de vista gráfico.
- En su uso típico, los archivos tipo XAML serían producidos por una herramienta de diseño visual, como Microsoft Visual Studio.

# Lenguajes de descripción de interfaces basadas en XML:

## ámbito de aplicación

### **XAML** (*eXtensible Application MarkupLanguage*)



- El XML resultante es interpretado en forma instantánea por un sub-sistema de despliegue de los sistemas Windows a partir del Vista que remplaza al GDI de las versiones anteriores de Windows.
- Los elementos de XAML se interconectan con objetos del Entorno Común de Ejecución.
- Los atributos se conectan con propiedades o eventos de esos objetos.
- XAML fue diseñado para soportar las clases y métodos de la plataforma .NET que tienen relación con la interacción con el usuario, en especial el despliegue en pantalla.

# Lenguajes de descripción de interfaces basadas en XML:

## ámbito de aplicación

### **XAML** (*eXtensible Application MarkupLanguage*)



- Con **Windows Presentation Foundation**, XAML es usado para **describir interfaces visuales para usuarios**.
- WPF permite la definición de objetos en 2D y 3D, rotaciones, animaciones y otra variedad de características y efectos.
- <http://es.wikipedia.org/wiki/XAML>
- <http://msdn.microsoft.com/es-es/library/cc295302%28v=expression.40%29.aspx>



# Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación



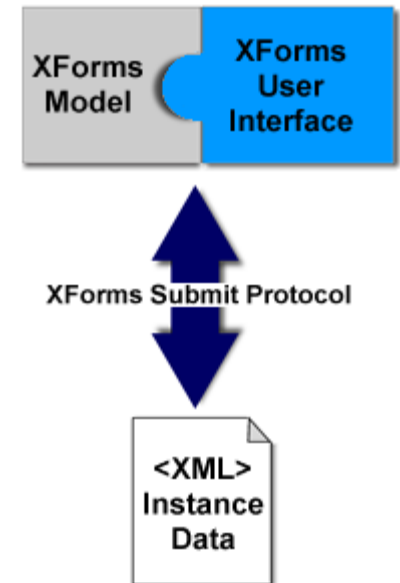
## Xforms

- Es un formato XML diseñado por el W3C para poder definir interfaces de usuario, **principalmente formularios web**.
- XForms ha sido diseñado para ser la nueva generación de formularios HTML/XHTML, pero es lo suficientemente genérico como para que pueda ser usado, de una manera independiente, para describir cualquier interfaz de usuario e incluso para realizar tareas simples y comunes de manipulación de datos.
- Recomendación oficial de W3C: XForms 1.1.
- Actualmente sólo el navegador Opera soporta XForms nativamente.

# Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación **Xforms**



- Existen varios plugins y extensiones que le dan soporte a otros navegadores.
- XForms también puede ser usado a través de varias tecnologías de servidor que convierten el código de XForms a formularios de HTML en tiempo de ejecución y de manera transparente.
- <http://es.wikipedia.org/wiki/XForms>
- <http://www.w3c.es/Divulgacion/GuiasBreves/XForms>



# Lenguajes de descripción de interfaces basadas en XML:

## ámbito de aplicación

### ***XUL (XML-based User-interface Language)***



- Es la aplicación de XML a la descripción de la interfaz de usuario en el navegador Mozilla.
- No es un estándar.
- Destaca por su portabilidad y su independencia de dispositivo.
- Provee un gran conjunto herramientas para crear menús, paneles, barras de herramientas, asistentes, entre otras.
- Muchas de las interfaces desarrolladas para las extensiones de Mozilla Firefox son basadas en XUL.
- [http://es.wikipedia.org/wiki/XML-based\\_User-interface\\_Language](http://es.wikipedia.org/wiki/XML-based_User-interface_Language)
- <https://developer.mozilla.org/es/docs/XUL>

# Lenguajes de descripción de interfaces basadas en XML: ámbito de aplicación

## **XBL (*eXtensible Bindings Language*)**



- Es un lenguaje de marcas que se emplea para definir el comportamiento y la apariencia de aplicaciones XUL y elementos XML.
- El lenguaje XUL define la disposición de la interfaz de usuario de una aplicación, que puede adoptar diferentes aspectos dependiendo del estilo definido.
- Sin embargo resulta imposible definir cómo funciona cada elemento, como por ejemplo, la forma en que funcionan una barra de progreso.
- Es aquí donde entra en juego el **lenguaje XBL**.



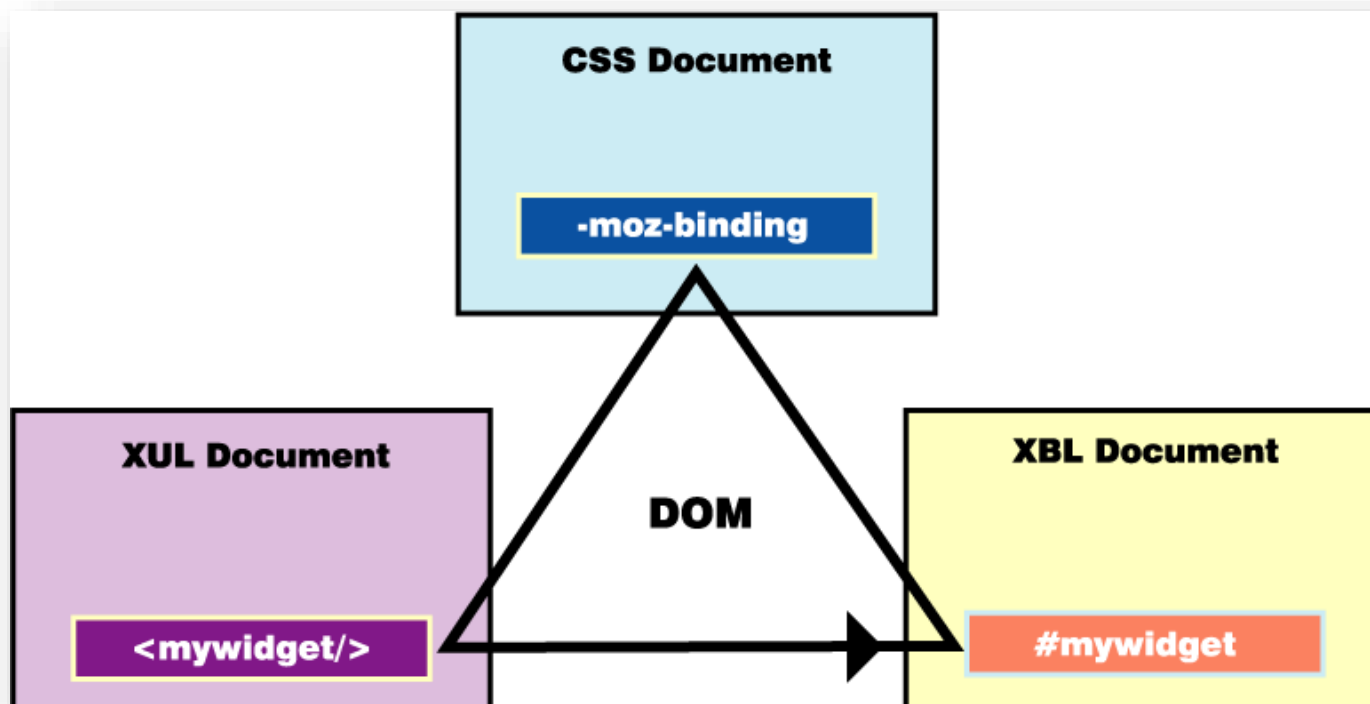
# Lenguajes de descripción de interfaces basadas en XML:

## ámbito de aplicación

### **XBL** (*eXtensible Bindings Language*)



- <https://developer.mozilla.org/es/docs/XBL>
- <http://www.w3.org/TR/xbl/>



## **2.2. Elementos, etiquetas, atributos y valores**

**Unidad 2: Generación de interfaces gráficas de usuario  
utilizando el lenguaje XML**

**24**

# Elementos, etiquetas, atributos y valores

- Existen tres términos comúnmente usados para describir las partes de un documento XML:

- *Etiquetas*
- *Elementos*
- *Atributos*

```
<direccion><datos>  
<titulo>Mrs.</titulo>  
<nombre> Mary </nombre>  
<apellidos> McGoon </apellidos> </datos>  
<calle> 1401 Main Street </calle>  
<ciudad estado="NC">Anytown</ciudad>  
<codigo-postal> 34829  
</codigo-postal>  
</direccion>
```

- Aquí está un documento de ejemplo que ilustra estos términos:

# Elementos, etiquetas, atributos y valores



- Una **etiqueta** es un texto entre el símbolo menor que (<) y el símbolo mayor que (>).
  - Existen etiquetas de inicio (como <nombre>) y etiquetas de fin (como </nombre>).
- Un **elemento** consta de la etiqueta de inicio, la etiqueta de fin y de todo aquello que esté entre ambas.
  - *En el ejemplo anterior, el elemento <datos> contiene tres elementos hijos: <titulo>, <nombre>, y <apellidos>.*

# Elementos, etiquetas, atributos y valores

- Un **atributo** es un par **nombre=valor** dentro de la etiqueta de inicio de un elemento.
  - *En este ejemplo, estado es un atributo del elemento <ciudad>:*

```
<direccion><datos>  
<titulo>Mrs.</titulo>  
<nombre> Mary </nombre>  
<apellidos> McGoon </apellidos>  
</datos>  
<calle> 1401 Main Street </calle>  
<ciudad estado="NC">Anytown</ciudad>  
<codigo-postal> 34829  
</codigo-postal>  
</direccion>
```

## **2.3. Herramientas libres y propietarias para la creación de interfaces de usuario multiplataforma**

Unidad 2: Generación de interfaces gráficas de usuario utilizando el lenguaje XML

# Herramientas libres y propietarias para la creación de interfaces de usuario multiplataforma



- Las herramientas de creación de interfaces de usuario, (*Sistema de Desarrollo de Interfaz de Usuario - SDIU*) nos permitirán:
  - Validar las entradas del usuario
  - Manipular errores y visualizar mensajes de error
  - Proporcionar respuestas y ayudas
  - Manipular ventanas y campos
  - Conectar el software de la aplicación y la interfaz
  - Aislar la aplicación de las funciones de gestión de la interfaz
  - Que el usuario personalice su interfaz.
  - Gestionar los dispositivos de salida (ratón, teclado, etc.)



# Herramientas libres y propietarias para la creación de interfaces de usuario multiplataforma



- Las herramientas de diseño de interfaces gráficas se encuadran dentro de las denominadas **RAD** (*Rapid Application Development*).
- *Clasificación:*
  - *Libres*
  - *Propietarias*

# Herramientas libres y propietarias para la creación de interfaces de usuario multiplataforma



## ■ *Libres*

- **Glade** (se considera una herramienta de diseño de interfaces pura ya que ésta es su única función). Luego podemos enlazarla con aquellos lenguajes de programación que deseemos.
- Otras como **Gambas**, **Mono**, **Kdevelop**, **Lazarus** y **Anjuta**, también de software libre, incluyen además las herramientas necesarias para elaborar el código.
- En Java podemos citar a **Eclipse** y **NetBeans** ambas de software libre aunque estos son compatibles con otros lenguajes.

# Herramientas libres y propietarias para la creación de interfaces de usuario multiplataforma



- *Propietarias* (software privativo )
  - Microsoft Visual Studio
  - Microsoft Expression Blend
  - Velneo
  - Jcreator
  - Delphi
  - Oracle
  - *Serían algunas de las más destacadas y todas ellas son **IDE** que incluyen el software de desarrollo de código además del de la interfaz de usuario.*

# **2.4. Controles. Propiedades**

**Unidad 2: Generación de interfaces gráficas de usuario  
utilizando el lenguaje XML**

**33**

# Controles. Propiedades

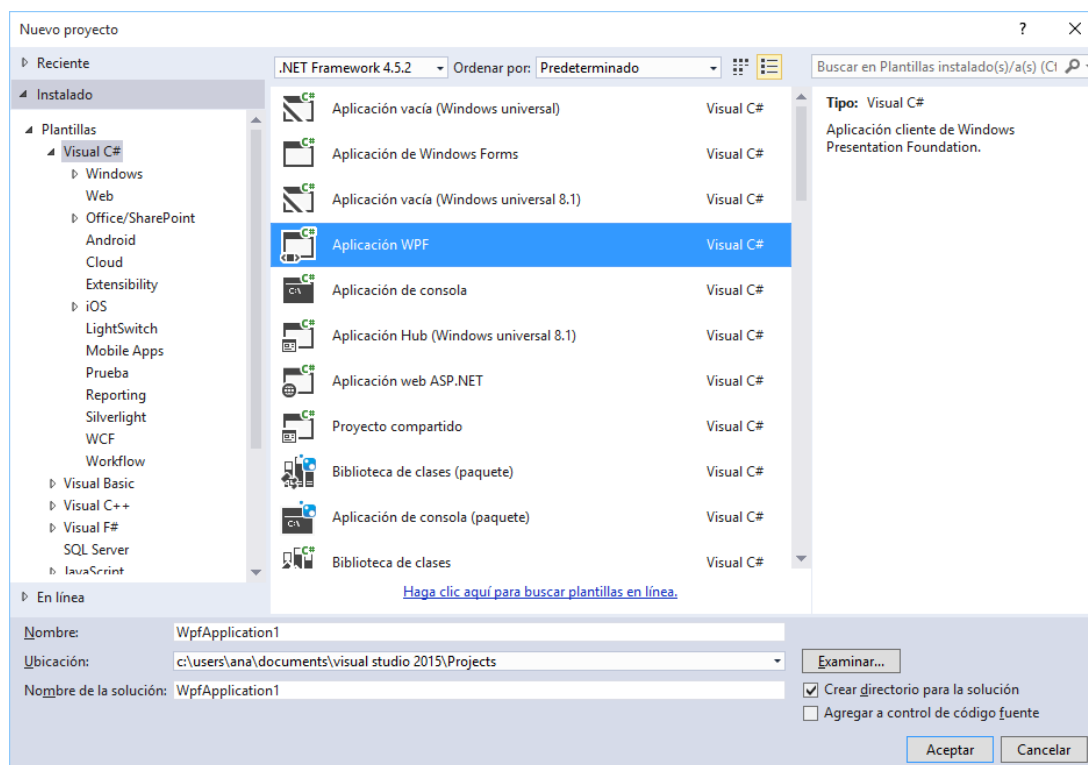
- Todos los controles de los que disponen los diferentes **Entornos de Diseño de Interfaces** disponen de una serie de propiedades las cuales podemos cambiar al incluirlos en las aplicaciones.
- Ejemplos de propiedades son el color, el tipo de letra, el nombre, el texto, etc...
- *En las practicas de clase veremos cómo utilizarlas.*
- A continuación mencionaremos los controles más generales y usuales que aparecen en los interfaces de usuario.

# Controles.

## Propiedades en XAML



- El tipo de aplicación que crearemos ahora será **Aplicación WPF** en Visual Studio 2015.

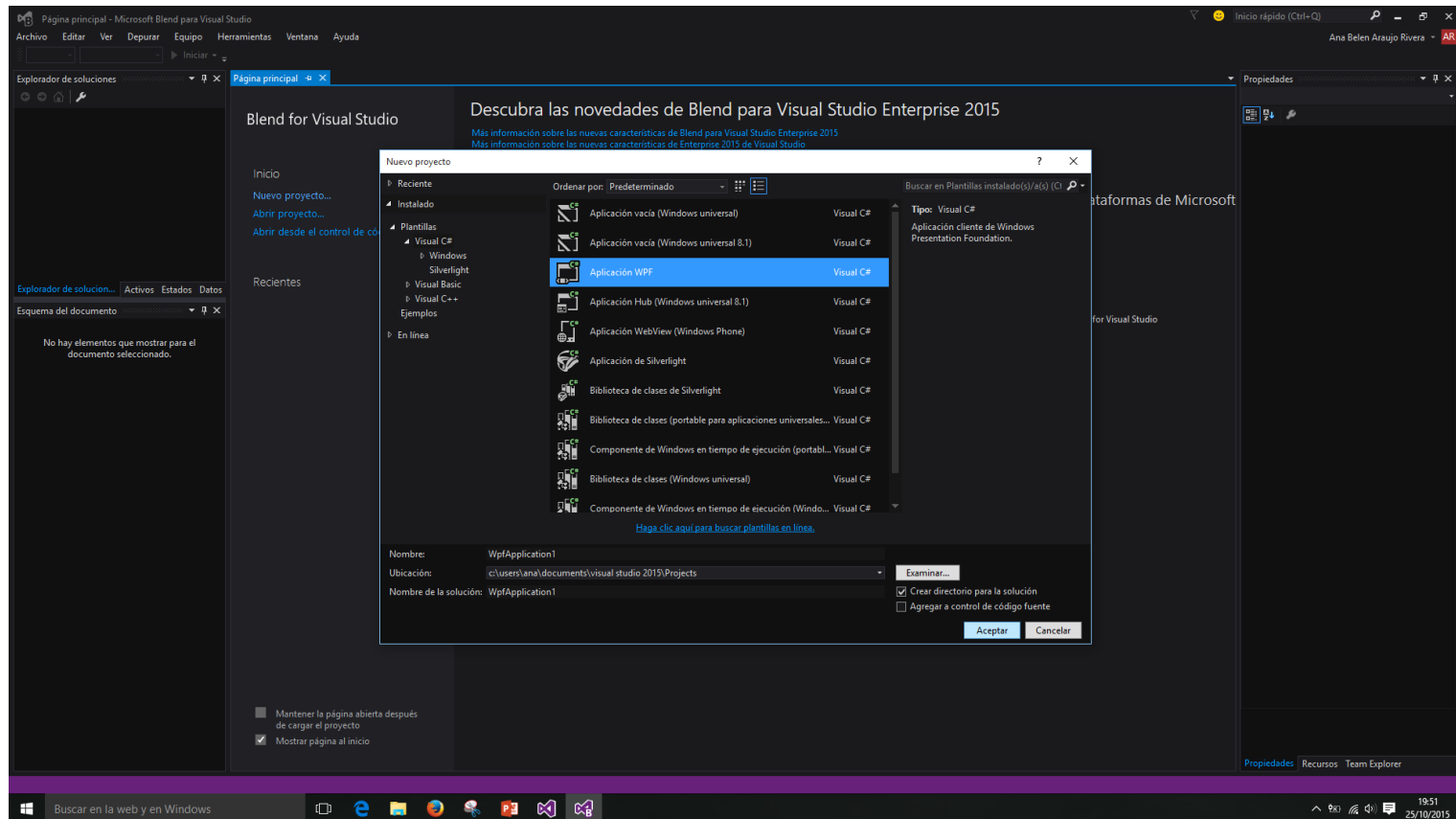




# Controles.

## Propiedades en XAML

- También podemos utilizar Microsoft Blend, totalmente compatible con Visual Studio.







# Controles.

## Propiedades en XAML

- Una aplicación WPF contiene dos ficheros .xaml por defecto: **App.xaml** y **MainWindow.xaml**, que inicialmente se muestran así:

```
<Application x:Class="WpfApplication3.App"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    StartupUri="MainWindow.xaml">
    <Application.Resources>
    </Application.Resources>
</Application>
```

App.xaml

```
<Window x:Class="WpfApplication3.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="MainWindow" Height="350" Width="525">
    <Grid>
    </Grid>
</Window>
```

MainWindow.xaml



# Controles.

## Propiedades en XAML

### ■ Elementos raíz

- Al igual que cualquier archivo XML estándar, los archivos XAML contienen un **elemento raíz**.
- Basándose en el tipo de aplicación, el archivo XAML contiene un elemento raíz diferente.
- Sin embargo, cada elemento raíz *actúa como un **contenedor principal***, que no tiene ninguna representación visual pero alberga diferentes controles o elementos de una forma estructurada para crear la interfaz de usuario.
- En App.xaml es:

```
<Application ...> ... </Application>
```

- En MainWindow.xaml es:

```
<Window ...> ... </Window>
```



# Controles.

## Propiedades en XAML

- Elemento por defecto en la interface de usuario
  - El elemento **Grid** es un control de diseño, y es el elemento de la interfaz de usuario predeterminada que aparece dentro del elemento raíz del archivo **MainWindow.xaml**

```
<Window x:Class="WpfApplication3.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="525">
    <Grid>

    </Grid>
</Window>
```

- Se puede utilizar como un control de diseño de base para la construcción de interfaces de usuario más complejas.
- *Puede llevar dentro otros controles Grid (Cuadrícula)*



# Controles.

## Propiedades en XAML

- **Controles integrados de WPF por función :**
  - Botones: [Button](#).
  - Entrada: [TextBox](#), [RichTextBox](#) y [PasswordBox](#).
  - Selección: [CheckBox](#), [ComboBox](#), [ListBox](#), [RadioButton](#) y [Slider](#).
  - Diseño: [Border](#), [Canvas](#), [DockPanel](#), [Expander](#), [Grid](#), [GridSplitter](#), [GroupBox](#), [Separator](#), [ScrollViewer](#), [StackPanel](#), [Viewbox](#) y [WrapPanel](#).
  - Presentación y selección de fechas: [Calendar](#) y [DatePicker](#).
  - Información para el usuario: [AccessText](#), [Label](#), [Popup](#), [ProgressBar](#), [StatusBar](#), [TextBlock](#) y [ToolTip](#).
  - Multimedia: [Image](#) y [MediaElement](#).
  - Navegación: [Frame](#) y [TabControl](#).
  - Menús: [Menu](#) , [ToolBar](#) y [ContextMenu](#).
  - Documentos: [DocumentViewer](#), [FlowDocumentPageViewer](#), [FlowDocumentReader](#), [FlowDocumentScrollViewer](#) y [StickyNoteControl](#).
  - Entradas de lápiz digitales: [InkCanvas](#) y [InkPresenter](#).
  - Presentación de datos: [DataGrid](#), [ListView](#) y [TreeView](#).



# Controles.

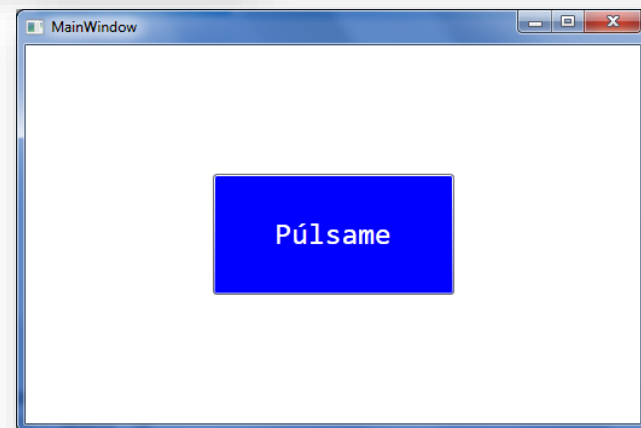
## Propiedades en XAML

- **Botones: Button**

- Representa un control de botón de Windows

```
<Button Width="200" Height="100" FontFamily="Consolas"
        FontSize="25" Background="Blue" Foreground="White">
    Púlsame
</Button>
```

- Propiedades, métodos y eventos





# Controles.

## Propiedades en XAML

- ***XAML y el code-behind (código trasero)***
  - *Si hacemos doble clic en un control podremos acceder a la parte de código del manejador de eventos por defecto.*
  - *Esta acción estará definida en un lenguaje procedural de .NET, por ejemplo C#, esto no se puede definir mediante XAML, ya que éste es un lenguaje de marcas declarativo.*
  - *Este código se almacena en un fichero **MainWindow.xaml.cs***

```
private void Button_Click_1(object sender, RoutedEventArgs e)
{
}
}
```



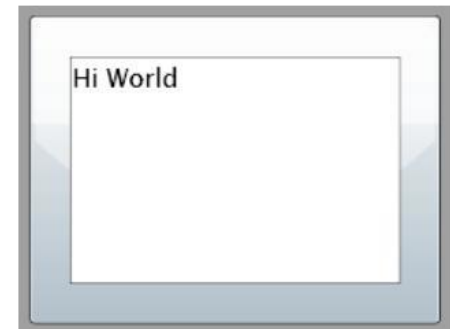
# Controles.

## Propiedades en XAML

### ■ Controles de Entrada: TextBox

- Representa un control que se puede utilizar para mostrar o modificar texto sin formato.
- Un uso común de TextBox es editar el texto sin formato de un formulario.
- *Por ejemplo, un formulario que pide el nombre del usuario, el número de teléfono, etc., utilizará controles TextBox para la entrada de texto.*
- Propiedades, métodos y eventos

```
<Grid x:Name="LayoutRoot">  
    <Button>  
        <TextBox  
            Text="Hi World"  
            Margin="35"  
            FontSize="25"  
        />  
    </Button>  
</Grid>
```







# Controles.

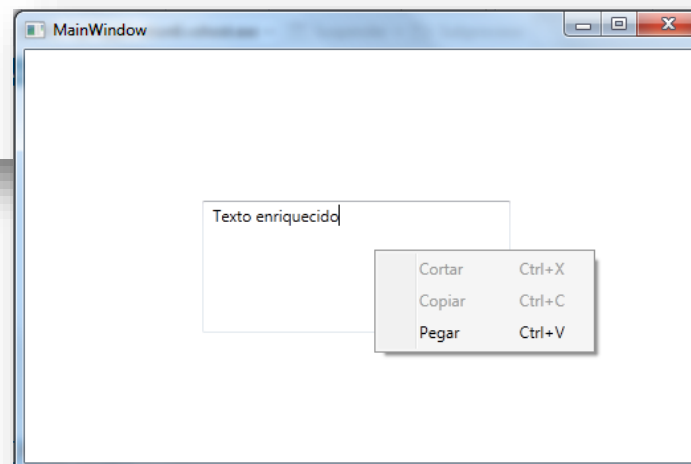
## Propiedades en XAML

### ■ Controles de Entrada: RichTextBox

- Representa un control de cuadro de texto enriquecido de Windows.
- Funciona sobre objetos FlowDocument.
- Propiedades, métodos y eventos

```
<RichTextBox HorizontalAlignment="Left" Height="100" Margin="111,102,0,0" VerticalAlignment="Top" Width="299">  
    <FlowDocument>  
        <Paragraph>  
            <Run Text="Texto enriquecido"/>  
        </Paragraph>  
    </FlowDocument>  
</RichTextBox>
```

Los FlowDocument son documentos que pueden ser creados y mostrados en una aplicación WPF. Consisten primordialmente en textos, con figuras y otros elementos incluidos en el entorno.





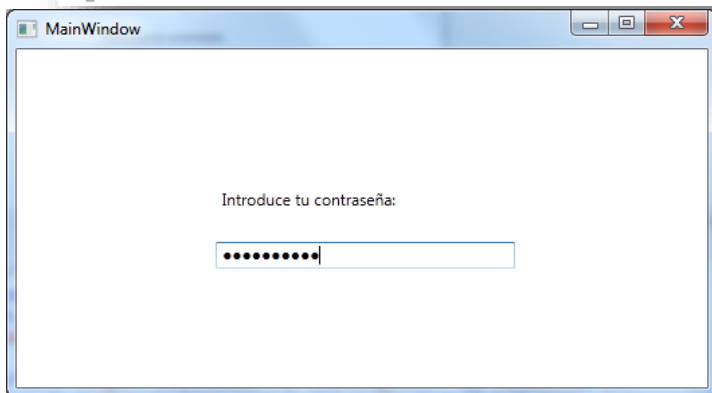
# Controles.

## Propiedades en XAML

### ■ Controles de Entrada: PasswordBox

- Representa un control diseñado para proteger y administrar contraseñas.
- Propiedades, métodos y eventos

```
<Window x:Class="WpfApplication6.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="525">
    <Grid>
        <PasswordBox HorizontalAlignment="Left" Margin="150,129,0,0" VerticalAlignment="Top" Width="194"/>
    </Grid>
</Window>
```



### Control de texto: Label

Representa la etiqueta de texto de un control y proporciona compatibilidad para teclas de acceso.

Propiedades, métodos y eventos



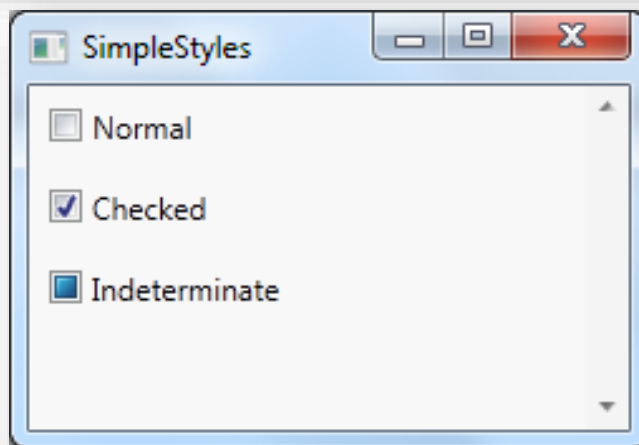
# Controles.

## Propiedades en XAML

### ■ Controles de Selección: CheckBox

- Representa un control que un usuario puede seleccionar y borrar.
- Propiedades, métodos y eventos

```
<CheckBox Margin="8">Normal</CheckBox>  
<CheckBox Margin="8" IsChecked="true">Checked</CheckBox>  
<CheckBox Margin="8" IsThreeState="true" IsChecked="{x:Null}">Indeterminate</CheckBox>
```





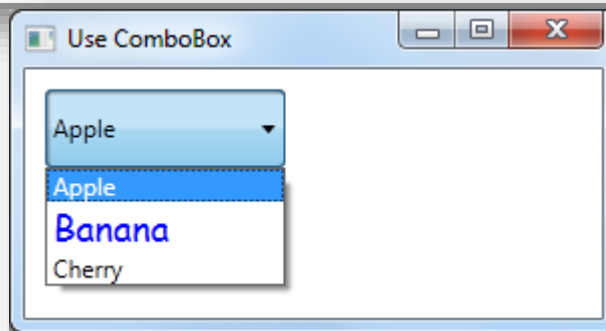
# Controles.

## Propiedades en XAML

### ■ Controles de Selección: ComboBox

- Representa un control de selección con una lista desplegable que se muestra o se oculta haciendo clic en la flecha en el control.
- Cada elemento se añade a la lista mediante **<ComboBoxItem>**

```
<ComboBox Height="39" HorizontalAlignment="Left" Margin="10,10,0,0" Name="ComboBox1"
           VerticalAlignment="Top" Width="120" SelectedIndex="0" VerticalContentAlignment="Center">
    <ComboBoxItem>Apple</ComboBoxItem>
    <ComboBoxItem FontFamily="Comic Sans MS" FontSize="18" Foreground="Blue">Banana</ComboBoxItem>
    <ComboBoxItem>Cherry</ComboBoxItem>
</ComboBox>
```



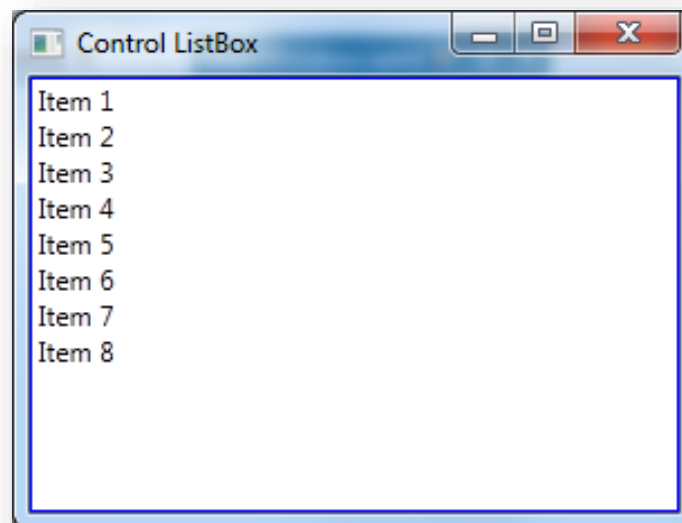


# Controles.

## Propiedades en XAML

- **Controles de Selección: ListBox**
  - Contiene una lista de elementos seleccionables.
  - Cada elemento se añade a la lista mediante **<ListBoxItem>**
  - Propiedades, métodos y eventos

```
<ListBox BorderThickness="1"
  VerticalAlignment="Stretch"
  HorizontalAlignment="Stretch"
  Grid.Column="0" Grid.ColumnSpan="1"
  Grid.Row="0" Grid.RowSpan="1"
  Name="listBox1" BorderBrush="Blue"
  SelectionMode="Single">
  <ListBoxItem>Item 1</ListBoxItem>
  <ListBoxItem>Item 2</ListBoxItem>
  <ListBoxItem>Item 3</ListBoxItem>
  <ListBoxItem>Item 4</ListBoxItem>
  <ListBoxItem>Item 5</ListBoxItem>
  <ListBoxItem>Item 6</ListBoxItem>
  <ListBoxItem>Item 7</ListBoxItem>
  <ListBoxItem>Item 8</ListBoxItem>
</ListBox>
```





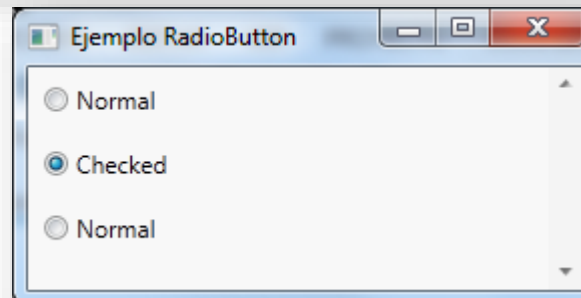
# Controles.

## Propiedades en XAML

### ■ Controles de Selección: RadioButton

- Representa un botón que se puede seleccionar por un usuario.
- La propiedad de IsChecked de RadioButton puede establecerse haciendo clic, pero puede estar desactivada sólo mediante programación.
- Propiedades, métodos y eventos

```
<RadioButton Margin="8">Normal</RadioButton>  
<RadioButton Margin="8" IsChecked="true">Checked</RadioButton>  
<RadioButton Margin="8">Normal</RadioButton>
```





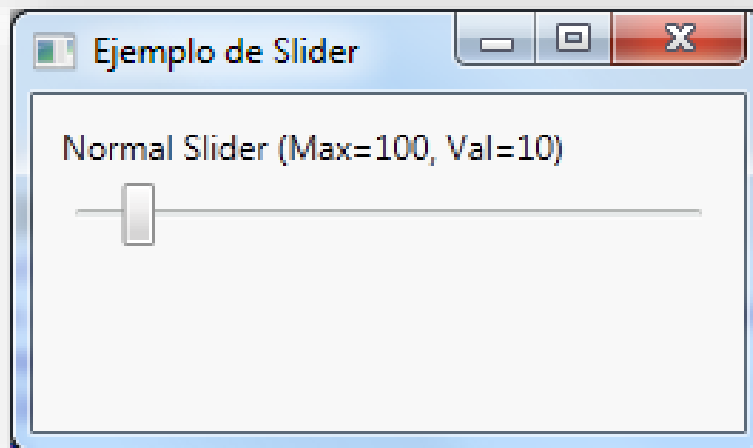
# Controles.

## Propiedades en XAML

### ■ Controles de Selección: Slider

- Representa un control que permite al usuario seleccionar un valor de un intervalo determinado moviendo un control.
- Propiedades, métodos y eventos

```
<Slider Maximum="100" Value="10"></Slider>
```



# Direcciones web de consulta

- <http://es.wikipedia.org/wiki/Xml>
- [Introducción a WPF](#)
- [Introducción a XAML](#)
- <http://wpftutorial.net/>
- <http://www.java2s.com/Code/CSharp/Windows-Presentation-Foundation/CatalogWindows-Presentation-Foundation.htm>
- <http://www.c-sharpcorner.com>
- [Programming WPF \(O'Reilly\)](#)
- <http://www.dotnetheaven.com>
- <http://msdn.microsoft.com/es-es/library/vstudio/ms752347.aspx>