Tema 1.4.8. Controles de usuario

Los **controles de usuario**, representados en **WPF** por la **clase UserControl,** unen XAML y código en un contenedor reutilizable, con la misma interfaz, la misma funcionalidad. Puede ser utilizado en sitios diferentes e incluso en diferentes aplicaciones.

Un **control de usuario** actúa de forma muy parecida a una ventana - es un área donde puedes colocar otros controles y un fichero de código donde puedes interactuar con esos controles. El fichero que contiene el **control de usuario** también acaba con la extensión .xaml y su correspondiente fichero de código-detrás con .xaml.cs - como una ventana. La raíz del xaml es un poco diferente.

```
<UserControl x:Class="Prueba.UserControl1"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
             xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
             xmlns:local="clr-namespace:Prueba" mc:Ignorable="d" d:DesignHeight="450"
             d:DesignWidth="800">
    <Grid>

    </Grid>
</UserControl>
```

Estos controles nos permiten simplificar la aplicación y trabajar con propiedades definidas por nosotros mismos. Anteriormente vimos como las propiedades de controles como el botón (Content, Width , Height, etc.)

```
<Button Width="" Height="" Padding="" Background="" HorizontalAlignment="" />
```

Las depency/Properties de los controles de usuario nos permiten definir nuevas propiedades para comunicarnos entre el XAML y el c#.

Este pseudo tutorial nos permitirá crear una aplicación y optimizarla gracias a los controles de usuario.

PARTE 1

1) Creamos un proyecto vacío.

2) Modificamos las propiedades de la Main Window:

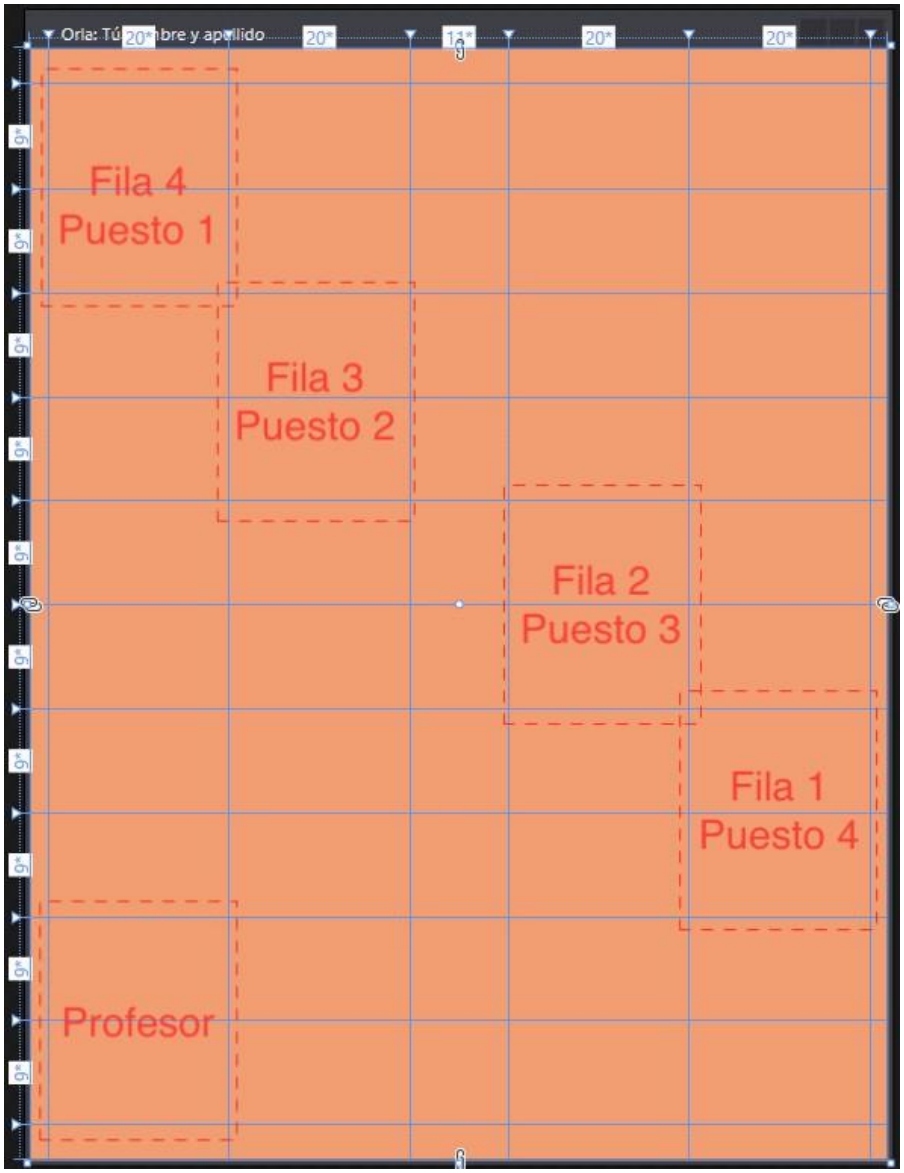
```
Title="Orla: Tú nombre y apellido" Height="800"
Width="600" MinHeight="800" Icon="/Imagenes/anuario.png"
MinWidth="600" MaxHeight="1920" MaxWidth="1080"
Background="#F49d6E">
```

Tendrás que descargar un icono, llamarle" anuario.png" y guardarlo en la carpeta "Imagenes" que tendrás que crear dentro de tu proyecto.

3) Creamos el Grid de la clase

```
<Grid.RowDefinitions>
        <RowDefinition Height="3*"/>
        <RowDefinition Height="9*"/>
        <RowDefinition Height="9*"/>
        <RowDefinition Height="9*"/>
        <RowDefinition Height="9*"/>
        <RowDefinition Height="9*"/>
        <RowDefinition Height="9*"/>
        <RowDefinition Height="9*"/>
        <RowDefinition Height="9*"/>
        <RowDefinition Height="9*"/>
        <RowDefinition Height="3*"/>
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="2*"/>
        <ColumnDefinition Width="20*"/>
        <ColumnDefinition Width="20*"/>
        <ColumnDefinition Width="11*"/>
        <ColumnDefinition Width="20*"/>
        <ColumnDefinition Width="20*"/>
        <ColumnDefinition Width="2*"/>
    </Grid.ColumnDefinitions>
```

La Main Window te debería quedar similar a esta:



Se han marcado varios de los puestos para facilitar la comprensión más adelante. Las filas 0 y 11 servirán de margen. Las filas 1 y 2 formarán nuestra fila 4 de ordenadores en la clase, la 3 y 4 la tercera fila, así sucesivamente hasta las 9 y 10 que formarán la fila que tiene el puesto del profesor.

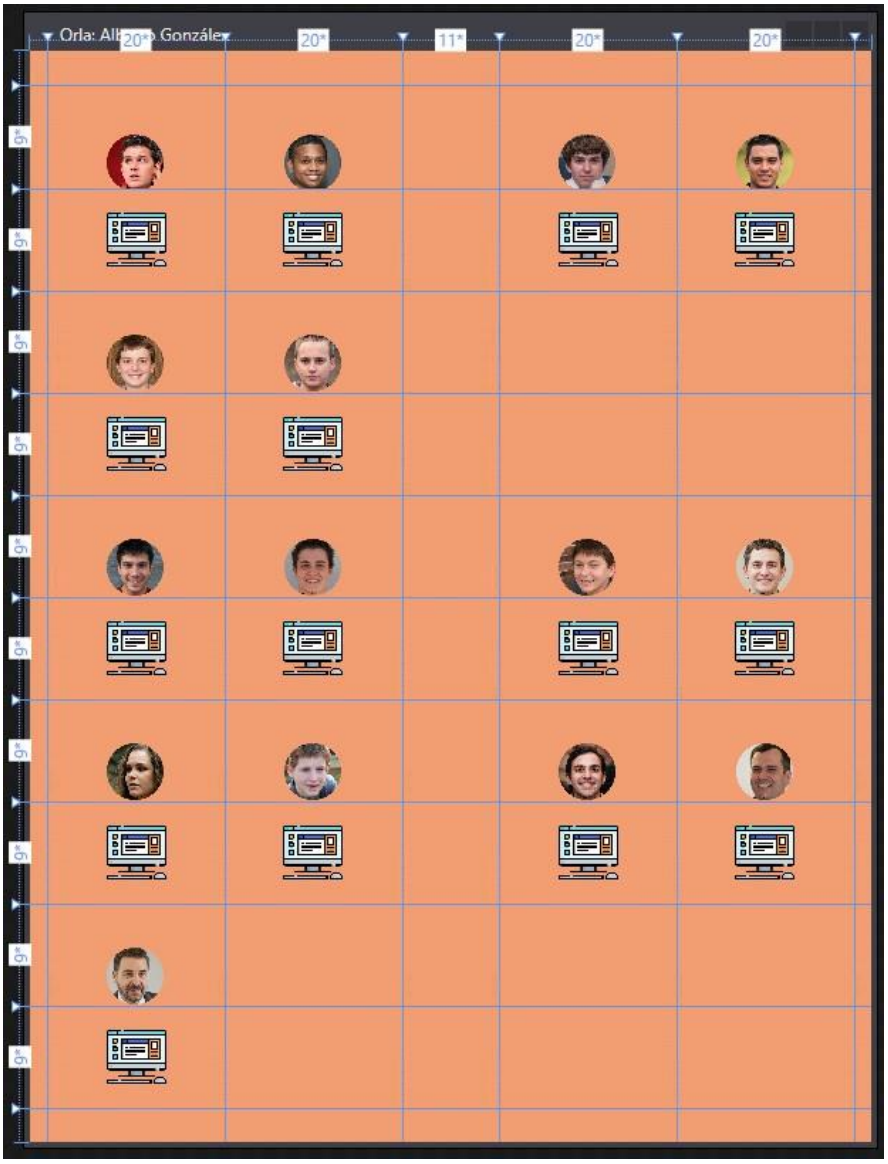
4) Rellenaremos los contenedores con StackPanel, TextBlock, botones e imágenes.

```
        <!-- Fila 4, Puesto 1 -->
  <StackPanel Grid.Row="1" Grid.Column="1" Orientation="Vertical" >
     <Viewbox Height="20" MaxWidth="100" StretchDirection="DownOnly">
       <TextBlock   Text="Aritz   P"   FontStretch="Expanded"   x:Name="LabelPuesto41"
Background="Transparent" HorizontalAlignment="Center" VerticalAlignment="Center"/>
     </Viewbox>
     <Button    UseLayoutRounding="True"    x:Name="Persona41"    Margin="15,15,15,0"
BorderThickness="0"  Height="40" Width="40">
          <Button.OpacityMask>
              <ImageBrush ImageSource="/Imagenes/usuario.png"/>
          </Button.OpacityMask>
          <Button.Background>
                <ImageBrush Stretch="UniformToFill" ImageSource="/Imagenes/AritzA.jpg"
/>
          </Button.Background>
     </Button>
  </StackPanel>
  <Image Grid.Row="2" Grid.Column="1" Source="/Imagenes/PC.png" Margin="15"/>
```

Como podrás comprobar tenemos un StackPanel que ocupa la parte de arriba del puesto, y lo rellena con un TextBlock y un botón. La parte de abajo la ocupa la imagen de un ordenador.

Fíjate que el botón tiene configurada una máscara y una imagen de fondo.

5) Ahora, tendrás que copiar este código y modificarlo para el resto de los puestos de la clase. Tendrás que cambiar las fotos del alumnado (Recuerda meterlas en la carpeta de Imagenes), y los nombres. El resultado tendrá que ser similar a este (Faltan los nombres encima de, las fotos, pero a ti sí que te tienen que aparecer):



6) Ahora vamos a programar el funcionamiento que pide la actividad:

a)“**On Hover”** que muestra el nombre al pasar el ratón por **encima de la foto**.

Añadiremos las acciones relacionadas con el Hover:

```
MouseEnter="MouseEnter41"   MouseLeave="MouseLeave41"
```

Fíjate que tienen el 41 en el nombre porque pertenecen a las acciones del puesto 1 de la fila 4.

El código que tendremos que añadir en el c# será el siguiente:

```
private void MouseEnter41(object sender, MouseEventArgs e)
        {
              LabelPuesto41.Text = "Nombre del alumno en el
puesto 41";
        }

                private void MouseLeave41(object sender,
MouseEventArgs e)
        {
            LabelPuesto41.Text = "";
        }
```

b) **"On click"** que muestre el email al hacer un click encima de la foto.

Añadiremos la acción relacionada con el click:

```
Click="MouseClick41"
```

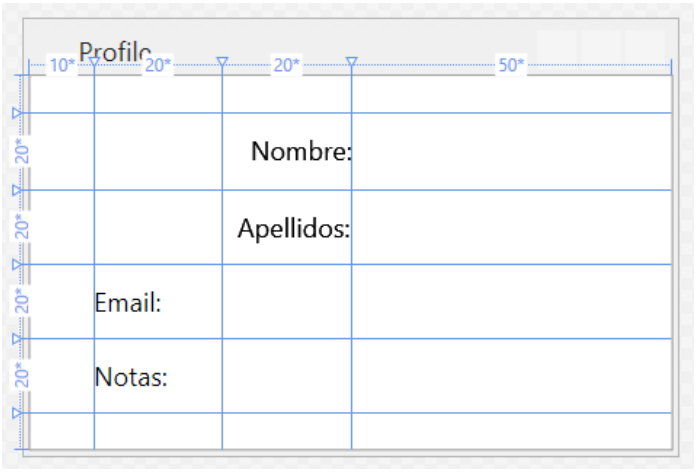
Fíjate que tiene el 41 en el nombre porque pertenece a las acciones del puesto 1 de la fila 4.

El código que tendremos que añadir en el c# será el siguiente:

```
private void MouseClick41(object sender, RoutedEventArgs e)
    {
        LabelPuesto41.Text = "email_alumno@educacion.navarra.es";
    }
```

c) "**On Double click"** sobre la foto, que abra una NUEVA VENTANA (la he llamado "Profile") con la ficha del alumno/a.

Primero tendremos que crear otra ventana donde mostraremos el perfil del usuario. Crea una nueva ventana y aquí tienes el código para organizar la información:



Profile.xaml

```
<Grid>

        <Grid.RowDefinitions>
            <RowDefinition Height="10*"/>
            <RowDefinition Height="20*"/>
            <RowDefinition Height="20*"/>
            <RowDefinition Height="20*"/>
            <RowDefinition Height="20*"/>
            <RowDefinition Height="10*"/>
        </Grid.RowDefinitions>

        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="10*"/>
            <ColumnDefinition Width="20*"/>
            <ColumnDefinition Width="20*"/>
            <ColumnDefinition Width="50*"/>
        </Grid.ColumnDefinitions>
    <Image x:Name="Profile_Image" Grid.Row="1" Grid.RowSpan="2"
Grid.Column="1"/>
     <TextBlock Text="Nombre:" Grid.Row="1" Grid.Column="2"
VerticalAlignment="Center" HorizontalAlignment="Right"/>
            <TextBlock    x:Name="Profile_Nombre"    Grid.Row="1"
Grid.Column="3"                          VerticalAlignment="Center"
HorizontalAlignment="Center" FontStretch="Expanded"/>

        <TextBlock Text="Apellidos:" Grid.Row="2" Grid.Column="2"
VerticalAlignment="Center" HorizontalAlignment="Right"/>
           <TextBlock x:Name="Profile_Apellidos" Grid.Row="2"
Grid.Column="3"                          VerticalAlignment="Center"
HorizontalAlignment="Center" FontStretch="Expanded"/>

         <TextBlock Text="Email:" Grid.Row="3" Grid.Column="1"
Grid.ColumnSpan="3" VerticalAlignment="Center"
HorizontalAlignment="Left"/>
                <TextBlock x:Name="Profile_Email" Grid.Row="3"
Grid.Column="2" Grid.ColumnSpan="2" VerticalAlignment="Center"
HorizontalAlignment="Center" FontStretch="Expanded"/>

            <TextBlock Text="Notas:" Grid.Row="4" Grid.Column="1"
Grid.ColumnSpan="3"                      VerticalAlignment="Center"
HorizontalAlignment="Left"/>
                <TextBlock x:Name="Profile_Notas" Grid.Row="4"
Grid.Column="2" Grid.ColumnSpan="2" VerticalAlignment="Center"
HorizontalAlignment="Center" FontStretch="Expanded"/>

    </Grid>
```

El código de la clase de la NUEVA VENTANA (yo la he llamado Profile):

```
public Profile(String Nombre, String Apellidos,String Email, String Foto)
        {
            InitializeComponent();
            Profile_Nombre.Text = Nombre;
            Profile_Apellidos.Text = Apellidos;
            Profile_Email.Text = Email;
             Profile_Image.Source = new BitmapImage(new Uri(string.Format(@"..\..\{0}", Foto), UriKind.Relative));
        }
```

Lo último será añadir la última acción relacionada con el dobleclick:

```
MouseDoubleClick="Persona41_MouseDoubleClick"
```

Fíjate que tiene el 41 en el nombre porque pertenece a las acciones del puesto 1 de la fila 4.

Y el código que tendremos que añadir en el c# será el siguiente:

```
private void Persona41_MouseDoubleClick(object sender, MouseButtonEventArgs e)
        {
                        Profile win2 = new Profile("Nombre", "Apellidos", "email_alumno@educacion.navarra.es", "Imagenes/AritzP.jpg");
            win2.Show();
        }
```

Fíjate que el código crea una nueva ventana, y le pasa por parámetro los 4 strings que nos hacen falta para la ventana profile.

Hasta este punto no deberías haber tenido ningún problema ya que todo lo hemos dado en clases anteriores. Ahora vamos a crear un control de usuario con **DependecyProperties** para facilitarnos la tarea de realizar el resto de la orla de clase.

Imagina que una vez aquí tuvieras que copiar y pegar el código 14 veces más, cambiar todos los nombres, emails, fotos, nombres de variables, nombres de acciones, etc. Además de que sería muy costoso en tiempo, tu código sería una ******. No me entiendas mal, pero si tienes que realizar un cambio, por pequeño que sea, tendrás que editarlo 15 veces. Por no hablar de que, si es un proyecto en el que estás participando en grupo, es una locura para que el resto de participantes en el proyecto entiendan tus intenciones.

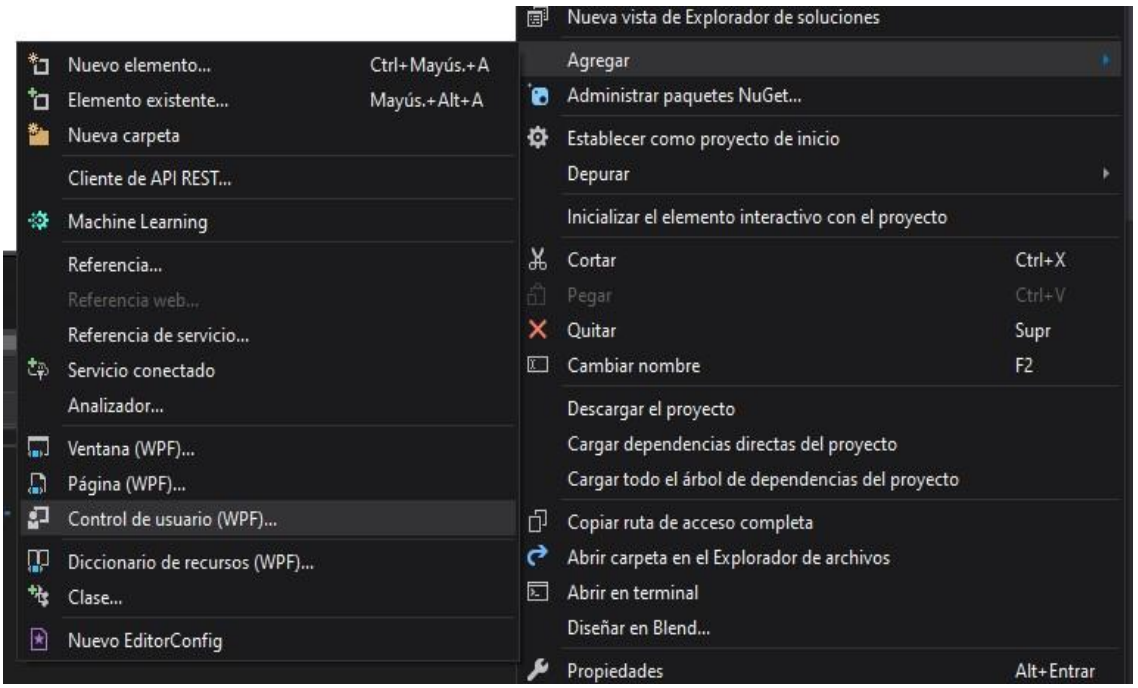
Para esto WPF cuenta con los controles de usuario, y el funcionamiento es similar a los componentes que creamos en el tema de prototipado con Figma. Pasamos a la PARTE 2 del Tutorial.

PARTE 2

Duplica ahora la carpeta del proyecto/aplicación/repositorio que has creado en la PARTE 1, y añádele al nombre "_con Control de Usuario"

Vamos a ver ahora la manera de **simplificar el proceso de creación** de la aplicación, y para ello sigue los siguientes pasos:

1) Para crear un nuevo control de usuario iremos al proyecto en el explorador de soluciones y con el botón derecho del ratón podremos agregar un nuevo control de usuario WPF:



Agregar →Control de usuario

2) Al crear el **Control de usuario (WPF)** tendremos tanto un archivo XAML, como su homónimo en c#. Llámalo "Puesto.xaml"

El funcionamiento es similar al del resto de ventanas. Definiremos en el XAML el estilo y los controles que queremos reutilizar. Puedes directamente coger el código que hemos colocado en la MainWindow y reutilizarlo para crear tu primer control de usuario. La primera parte se ha definido de esta manera:

```
<UserControl x:Class="Orla.Puesto"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
             xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
             xmlns:local="clr-namespace:Orla"
             mc:Ignorable="d"
             x:Name="root"
             d:DesignHeight="450" d:DesignWidth="400">
    <Grid>
        <Grid.RowDefinitions>

            <RowDefinition Height="10*"/>
            <RowDefinition Height="45*"/>
            <RowDefinition Height="45*"/>

        </Grid.RowDefinitions>

        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="100*"/>
        </Grid.ColumnDefinitions>

        <Viewbox MaxWidth="100" StretchDirection="DownOnly">
            <TextBlock FontStretch="Expanded" x:Name="LabelPuesto" Grid.Row="0" Grid.Column="0" Background="Transparent"
                       HorizontalAlignment="Center" VerticalAlignment="Center"/>
        </Viewbox>

        <Button x:Name="Persona" Grid.Row="1" Grid.Column="0" Margin="15,15,15,0" BorderThickness="0"
                MouseEnter="Persona_MouseEnter" MouseLeave="Persona_MouseLeave" Click="Persona_Click" MouseDoubleClick="Persona_MouseDoubleClick">
            <Button.OpacityMask>
                <ImageBrush ImageSource="Imagenes/usuario.png" Stretch="Uniform"/>
            </Button.OpacityMask>
            <Button.Background>
                <ImageBrush ImageSource="{Binding Foto, ElementName=root}" Stretch="Uniform"/>
            </Button.Background>
        </Button>

        <Image Grid.Row="2" Grid.Column="0" Source="/Imagenes/PC.png" Margin="15"/>

    </Grid>
</UserControl>
```

Puesto.xaml

Fíjate que, aunque se parece mucho, y de hecho este código está basado en lo que hemos hecho anteriormente, hay algunos cambios importantes:

- Los únicos controles que tienen acciones asociadas son el **TextBlock** y el **Button**, y los nombres de estos controles ya no dependen del lugar en el que se encuentran.
- Vamos a utilizar los nombres "**LabelPuesto**" y "**Persona"** para los dos controles respectivamente.
- Otro aspecto a tener en cuenta son las **acciones del botón** para las cuales también utilizaremos nombres genéricos:

  Persona_MouseEnter, Persona_MouseLeave, Persona_Click y Persona_MouseDoubleClick.

3) Vamos ahora con el **código en c# del Control de Usuario (Puesto.xaml.cs)**. Aquí es donde ocurre la magia. El primer paso sería definir las acciones que ya teníamos definidas en el MainWindow.cs y que no queríamos copiar y pegar tantas veces:

```
private void Persona_MouseEnter(object sender, MouseEventArgs e)
        {
            LabelPuesto.Text = Nombre + " " + Apellidos;
        }

private void Persona_MouseLeave(object sender, MouseEventArgs e)
        {
            LabelPuesto.Text = "";
        }

private void Persona_Click(object sender, RoutedEventArgs e)
        {
            LabelPuesto.Text = Email;
        }

private void Persona_MouseDoubleClick(object sender, MouseButtonEventArgs e)
        {
            Profile win2 = new Profile(Nombre, Apellidos, Email, Foto);
            win2.Show();
        }
```

Puesto.xaml.cs

Hay que fijarse que el código que hay dentro de las funciones es similar, pero este código utiliza variables. Estas variables son propiedades del control de usuario que vamos a definir a continuación.

4) Para definir las **DependencyProperties** nos hacen falta 2 partes:

a) La definición de DependencyProperty:

```
public static DependencyProperty NombreProperty = DependencyProperty.Register("Nombre",
typeof(string), typeof(Puesto), new PropertyMetadata(string.Empty));
```

En la definición estamos definiendo el “nombre de la propiedad”, el tipo (String), el propietario de la misma (la misma clase), y que la inicializamos con un String Vacío.

b) La segunda parte es la definición de la variable para poder utilizarla en la clase:

```
0 referencias | 0 cambios | 0 autores, 0 cambios
public Puesto()
{
    InitializeComponent();
}

2 referencias | 0 cambios | 0 autores, 0 cambios
public string Nombre
{
    get {return (string)GetValue(NombreProperty); }
    set { SetValue(NombreProperty, value); }
}
```

El **get** simplemente coge el valor de la propiedad "**NombreProperty**" y lo mete en la variable "**Nombre**", y el **set**, al contrario, Mete en la variable "**NombreProperty**" el valor que tenga la variable "Nombre".

Ahora faltaría añadir las **otras 3 propiedades** (Apellidos, Email y Foto), que tienes que realizar de la misma forma.

5) Ahora que ya tenemos creado el control de usuario con sus propiedades, **vamos a utilizarlo**.

Simplemente colocaremos este control en la **MainWindow.xaml**, dentro del contenedor que queramos, igual que hacemos con el resto de los controles (tenemos que eliminar u ocultar el código creado en la PARTE 1 que ya no necesitamos).

Y por ejemplo, en este caso, vamos a añadir el que está en la fila 2 y columna 2. Hay que tener en cuenta que ahora nuestro control de usuario ya tiene dos filas, luego podríamos reducir la definición inicial del **Grid** en el **MainWindow** en 5 filas menos, pero las que quedan deberán ser del doble de altura. Por esto ahora el control de usuario hace uso del **RowSpan**:

```
<local:Puesto  Nombre="Jone" Apellidos="Sola Lozano" Email="josoloz@educacion.navarra.es"
               Foto="Imagenes/Alumnado/Jone.png" Grid.Row="2" Grid.Column="2" Grid.RowSpan="1"/>
```

6) Añade todos l@s alumn@s, y haz que funcione sin errores. Más fácil imposible. :)

Aspecto final de la MainWindow.xaml