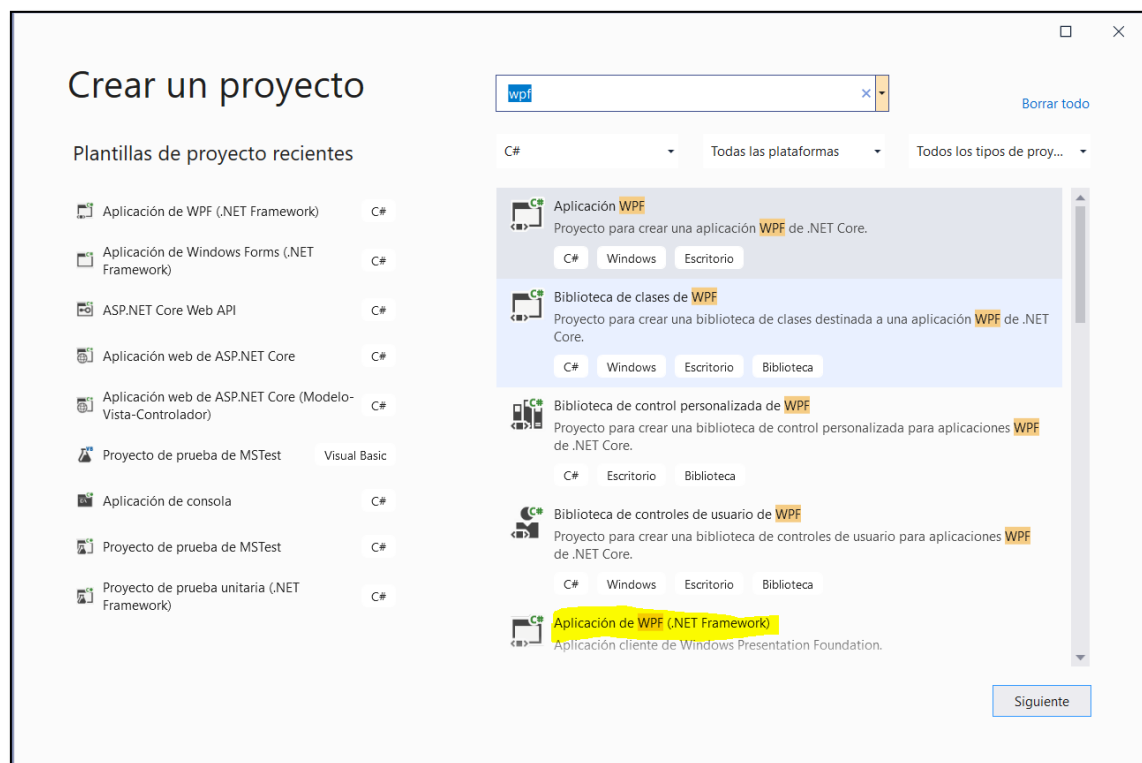


WPF

- WPF es una API que pertenece al framework .NET para la creación de interfaces de usuario en Windows.
- Utiliza el lenguaje XAML (similar a HTML).
- Sucesor de Windows Forms con nuevas funcionalidades como 3D, tipografía avanzada y documentos similares a PDF.

Actividad 20

1. Creamos un nuevo proyecto de “Aplicación de WPF (.NET Framework)”



Con el nombre “HolaMundo” y con la versión de FrameWork 4.6.1

Configure su nuevo proyecto

Aplicación de WPF (.NET Framework) C# XAML Windows Escritorio

Nombre del proyecto

HolaMundo

Ubicación

C:\Users\AntoniaRubio\source\repos

Solución

Crear nueva solución

Nombre de la solución ⓘ

HolaMundo

☐ Colocar la solución y el proyecto en el mismo directorio

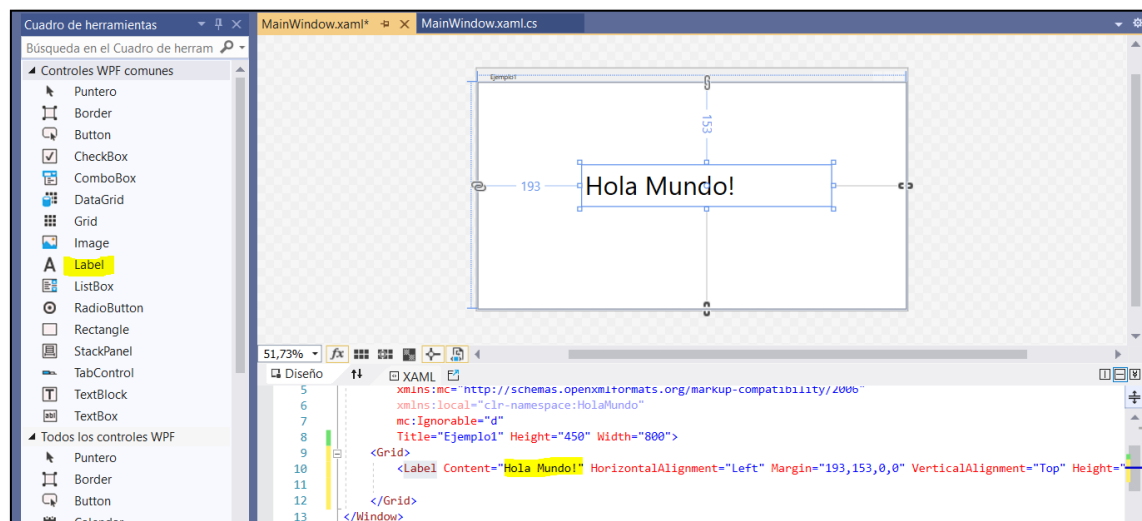
Framework

.NET Framework 4.6.1

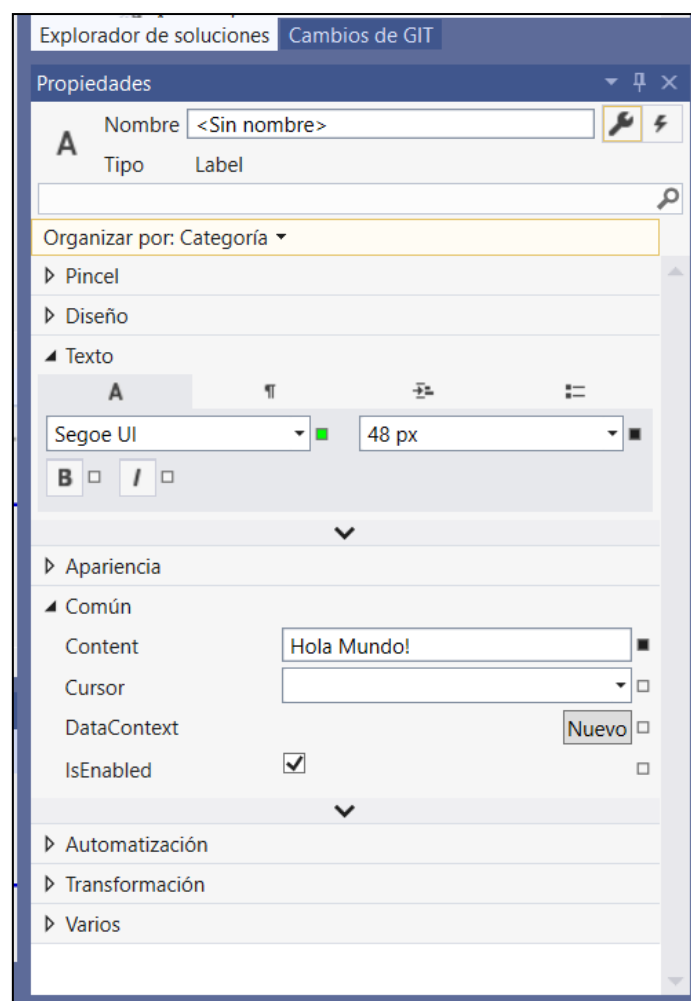
2. Editamos el titulo de la interfaz en la etiqueta “Title” y el tamaño en “Height” y “Width”.

```
51,73%
Diseño
XAML
1 <Window x:Class="HolaMundo.MainWindow"
2     xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3     xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4     xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
5     xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
6     xmlns:local="clr-namespace:HolaMundo"
7     mc:Ignorable="d"
8     Title="Ejemplo1" Height="450" Width="800">
9 <Grid>
```

3. Añadimos un elemento “Label” y editamos su contenido como “Hola Mundo!”.



Para editar el tipo de letra y el tamaño utilizamos las propiedades del elemento.

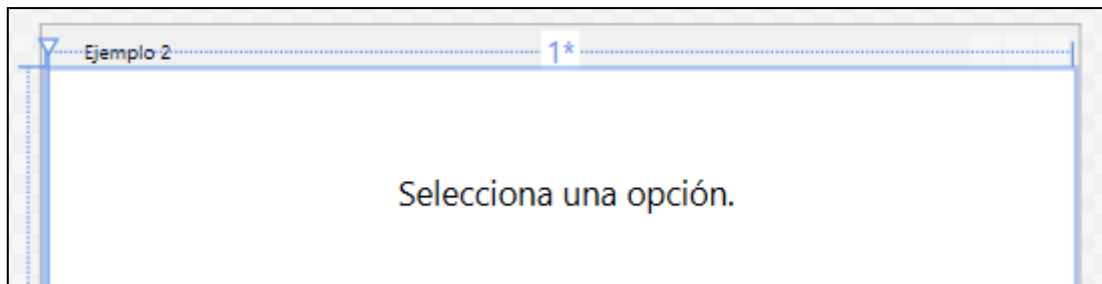


O añadiendo FontSize en la etiqueta.

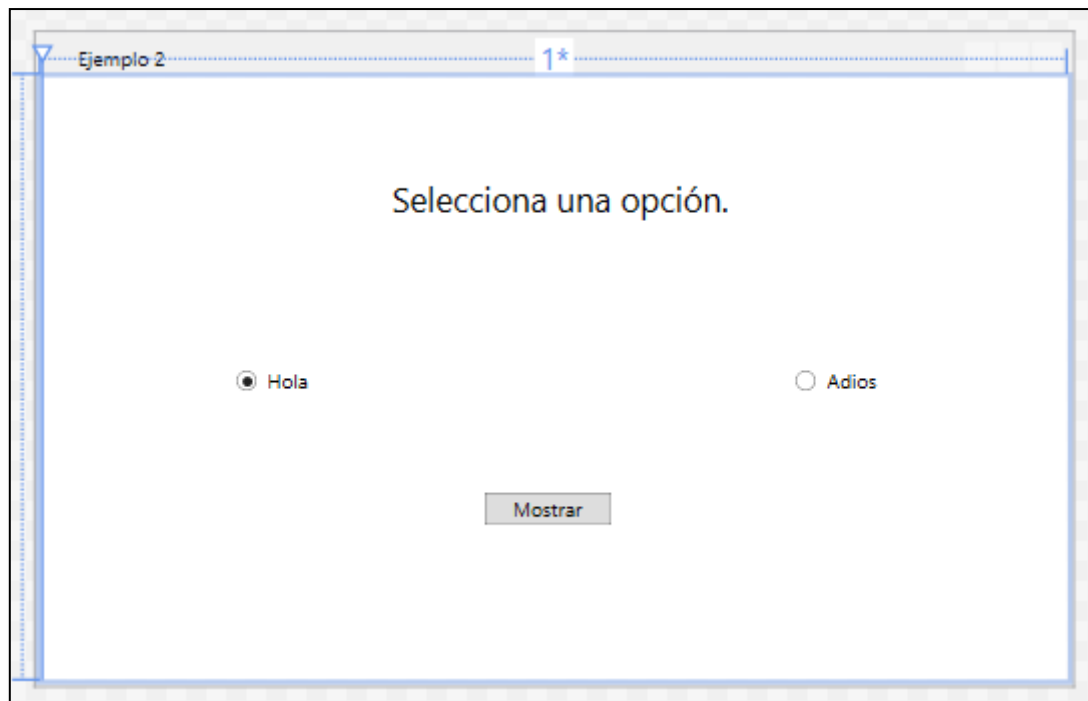
```
" Margin="193,153,0,0" VerticalAlignment="Top" Height="77" Width="462" FontSize="48"/>
```

Versión 2 Hola mundo

1. Crear un nuevo proyecto como hemos realizado anteriormente.
2. Añadimos un TextBlock y le cambiamos el atributo Text.



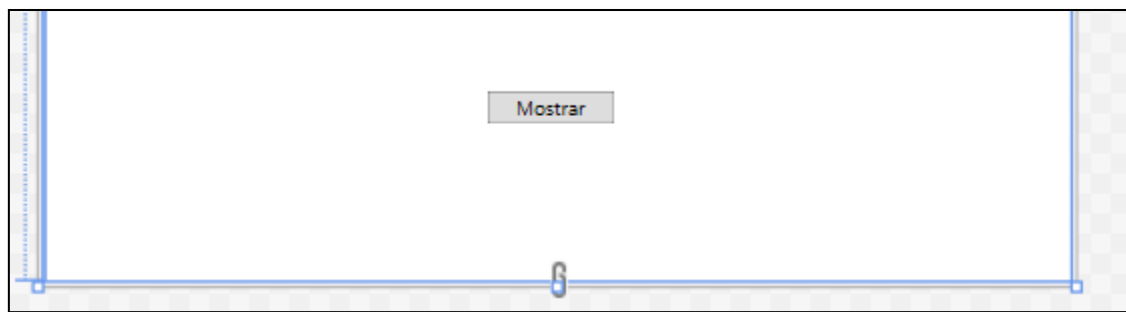
3. Añadimos dos RadioButton a los que les editaremos el texto y le añadiremos los controles "HelloButton" y "GoodbyeButton".



```
</Grid.ColumnDefinitions>
<TextBlock HorizontalAlignment="Left" Margin="210,60,0,0" TextWrapping="Wrap" Text="Selecciona una opción." VerticalAlignment="Top" FontSize="20" Grid.ColumnSpan="2"/>
<RadioButton x:Name="HelloButton" Content="Hola" HorizontalAlignment="Left" Margin="116,175,0,0" VerticalAlignment="Top" IsChecked="True" Grid.ColumnSpan="2"/>
<RadioButton x:Name="GoodbyeButton" Content="Adios" HorizontalAlignment="Left" Margin="451,175,0,0" VerticalAlignment="Top" Grid.ColumnSpan="2"/>
<Button Content="Mostrar" HorizontalAlignment="Left" Margin="266,249,0,0" VerticalAlignment="Top" Width="75" Click="Button_Click" Grid.ColumnSpan="2"/>
```

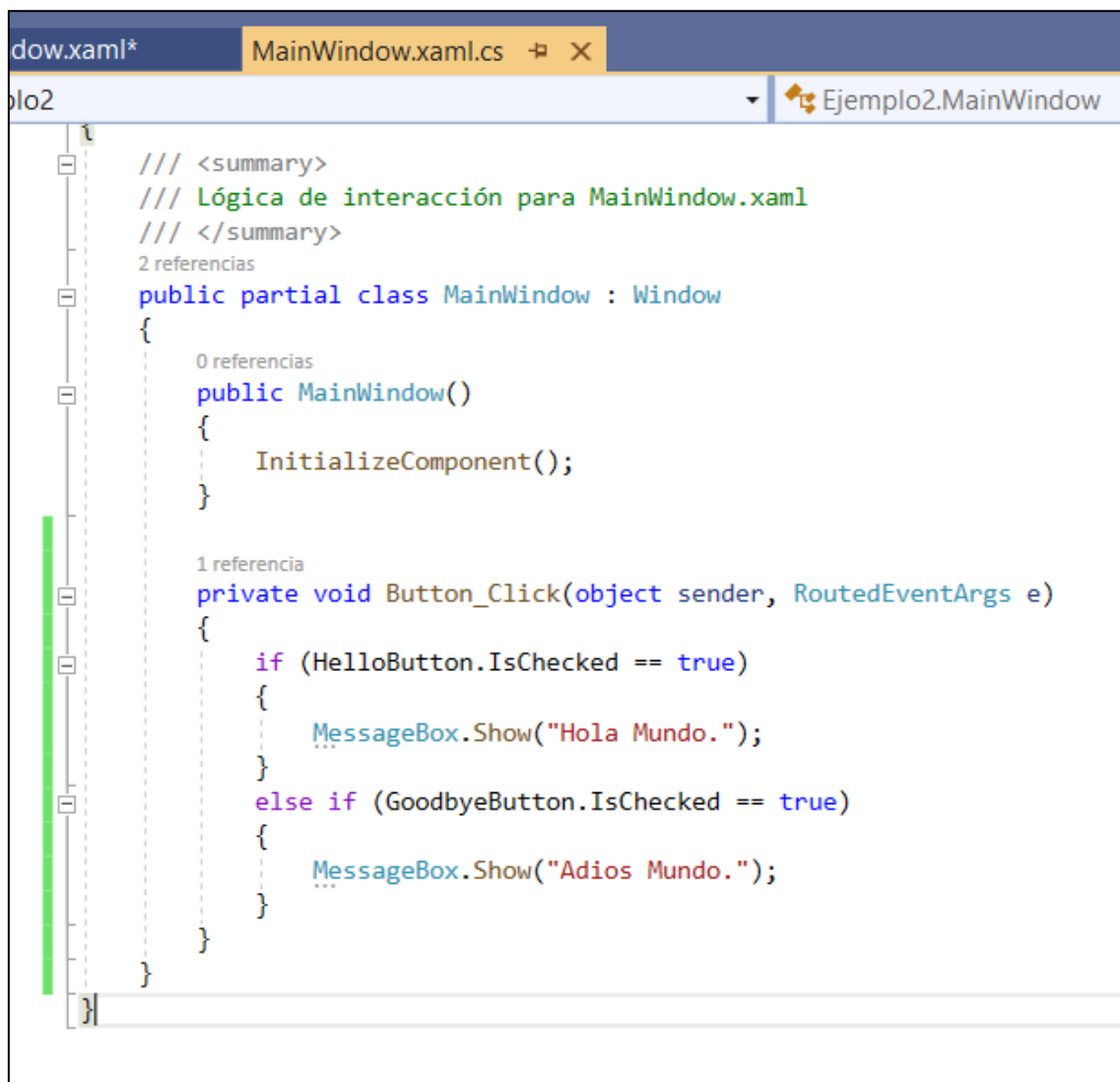
Añadimos la etiqueta IsChecked="True" al primer RadioButton.

4. Añadimos un Botón y cambiamos su etiqueta Content.



```
<RadioButton x:Name="GoodbyeButton" Content="Adios"
<Button Content="Mostrar" HorizontalAlignment="Left"/>
```

5. Damos funcionalidad al botón "Mostrar".



Ejercicio:

Realizar una interfaz similar a la siguiente.

The image shows a window titled "Rectangulos" with a standard OS-style title bar. Inside the window, there is a text label "Introduce angulo:" followed by an empty text input field. Below this, there are two labels, "Antes" and "Despues", positioned above two vertical rectangles. The rectangle under "Antes" is light blue, and the rectangle under "Despues" is light green. Both rectangles are currently oriented vertically.

El segundo rectángulo debe de girar en función al ángulo que se introduzca por parámetro.

This image shows the same "Rectangulos" window, but with the text input field containing the value "90". The "Antes" rectangle remains vertical and light blue. The "Despues" rectangle, which is light green, is now rotated 90 degrees counter-clockwise, appearing horizontal. This demonstrates the rotation functionality based on the input angle.