



I.E.S PALOMERAS  
VALLECAS

# Desarrollo de Interfaces

2º Desarrollo de Aplicaciones Multiplataforma



I.E.S. VILLABLANCA

**UT.3 Introducción a la generación de interfaces gráficas de usuario con editores visuales:  
Componentes básicos.**

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Contenidos

En esta Unidad de Trabajo vamos a tratar los siguientes puntos:

- Introducción. *Diapositivas 3-5*
- Componente Input Field. *Diapositivas 6-20*
- Componente Scrollbar. *Diapositivas 21-34*
- Componente Slider. *Diapositivas 35-49*

Material Adicional a esta presentación:

- Software:
  - UT3\_Ejemplo.unitypackage

Prácticas:

- Práctica\_UT3\_IntroduccionInterfaceGrafica

Vídeos:

- UT3\_Slider

## Contenidos

### Criterios de Evaluación:

- Se ha creado un interfaz gráfica utilizando los asistentes de un editor visual.
- Se han utilizado las funciones del editor para ubicar los componentes de la interfaz.
- Se han modificado las propiedades de los componentes para adecuarlas a las necesidades de la aplicación.
- Se ha analizado el código generado por el editor visual.
- Se ha modificado el código generado por el editor visual.
- Se han asociado a los eventos las acciones correspondientes.
- Se ha desarrollado una aplicación que incluye el interfaz gráfico obtenido.

## Introducción

En esta U.T. vamos a trabajar los componentes básicos de una GUI. Estos componentes son:

- **Input Field:** Nos permitirá interactuar con el usuario, solicitando información.
- **Scrollbar:** Nos permitirá mostrar paneles, en los que la información contenida en los mismos, se desplaza en horizontal o vertical, haciendo una GUI, más amigable.
- **Slider:** Control que nos dejará jugar con valores reales entre una cota inferior y una cota superior.
- **Toggle:** Opciones de selección. Importante para la realización de operaciones booleanas.
- **Dropdown:** Listas desplegables. Nos permitirán ofrecer varias opciones al usuario.

- **Raw Image:** Este componente nos permite introducir en nuestra GUI:

- Texturas.
- Sprites.
- Vídeos.

**Para poder entender la U.T.3, se debe haber adquirido los conceptos de la U.T.2.**



**Las Texturas y los Sprites se han trabajado en la U.T.2.**

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

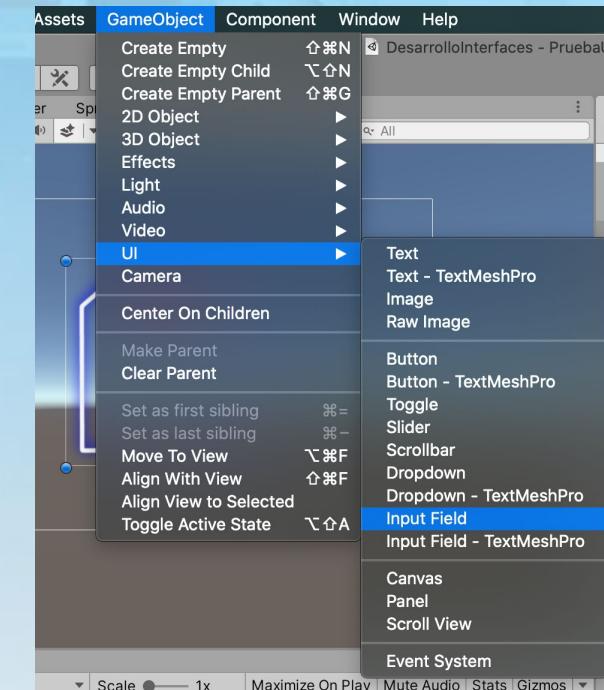
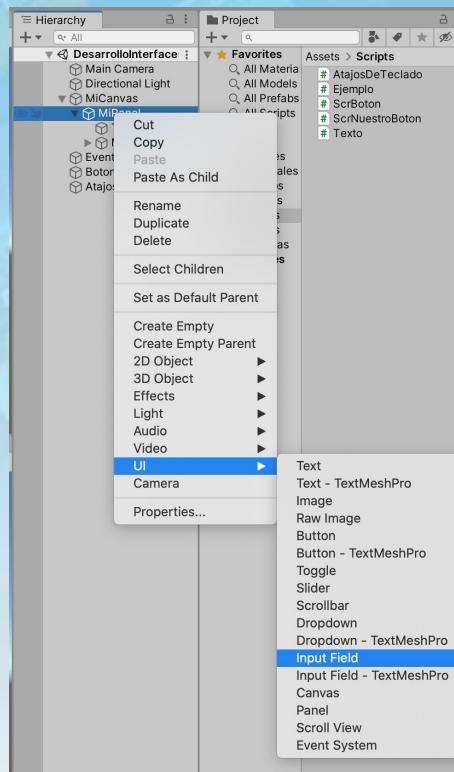
## Componentes básicos.

### Input Field

Este componente nos ofrece la oportunidad de poder interactuar con el usuario de forma directa, solicitando algún dato con el que vamos a trabajar desde nuestros Script.

Para insertar este componente, podemos seleccionar entre las dos opciones ya vistas anteriormente:

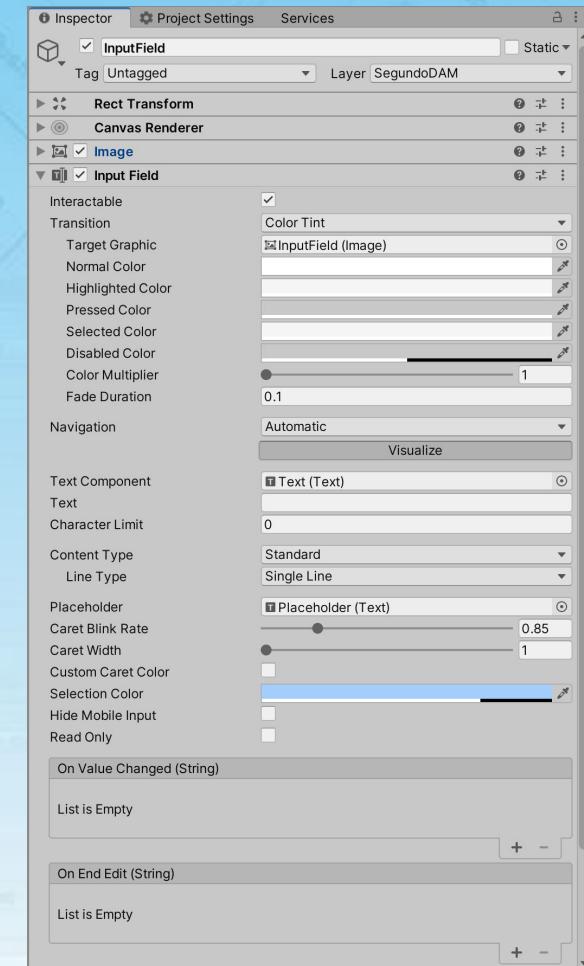
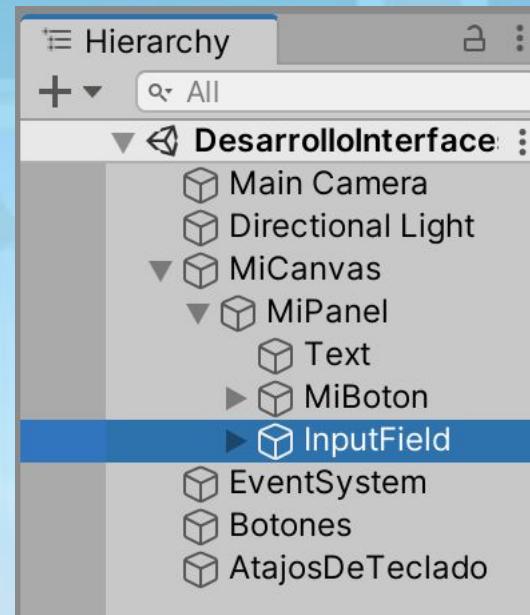
- Menú contextual.
- Barra de Menús.



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Input Field

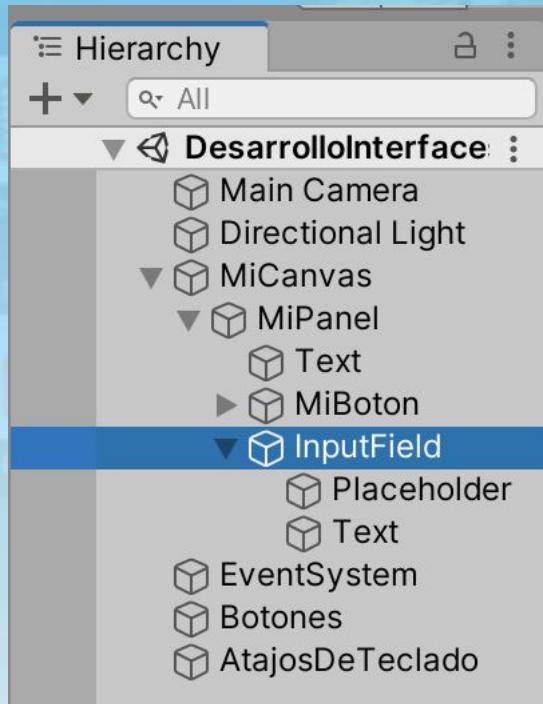
Cuando insertamos "Input Field" en nuestro Panel, lo que obtenemos en nuestra ventana "Scene", "Hierarchy" e "Inspector" es:



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Input Field

En lo primera que nos vamos a fijar es en la ventana Hierarchy, para comprobar cómo está compuesto el GameObject Input Field.



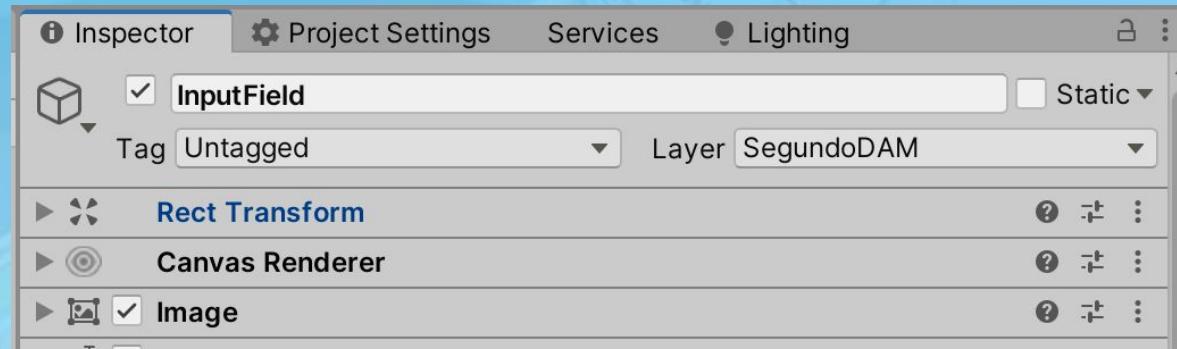
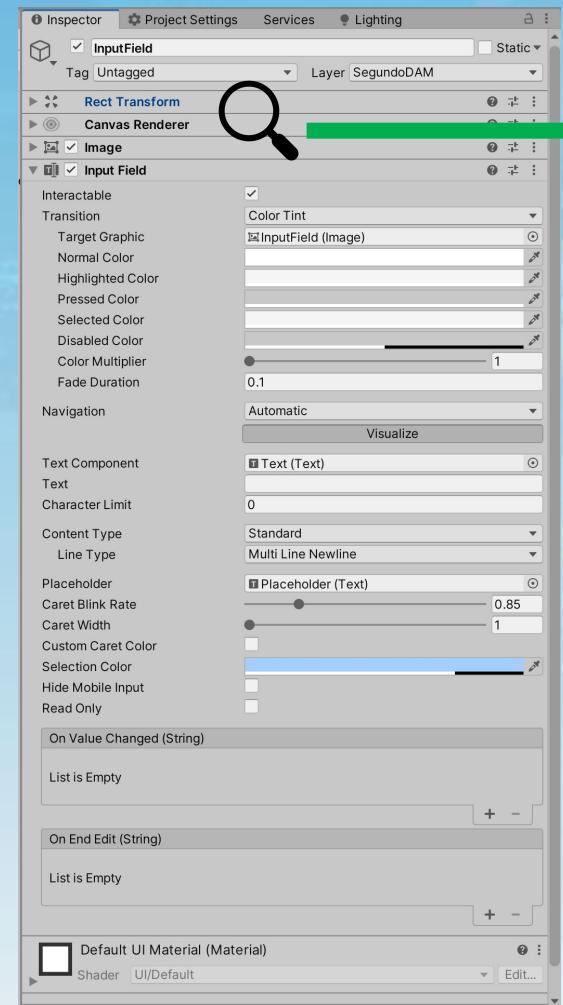
Comprobamos que tenemos un GameObject padre “InputField” y dos GameObject hijos “Placeholder” y “Text”. Vamos a fijarnos en la ventana “Inspector” de cada uno de estos GameObject.



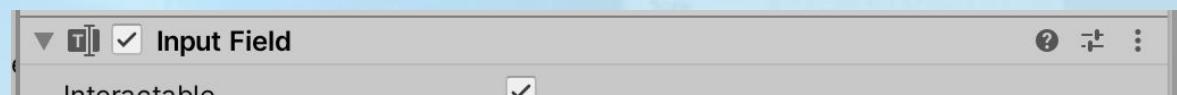
**Muchos de los componentes de la ventana Inspector ya han sido vistos en la U.T.2. En esta U.T. únicamente se mencionan.**

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Input Field

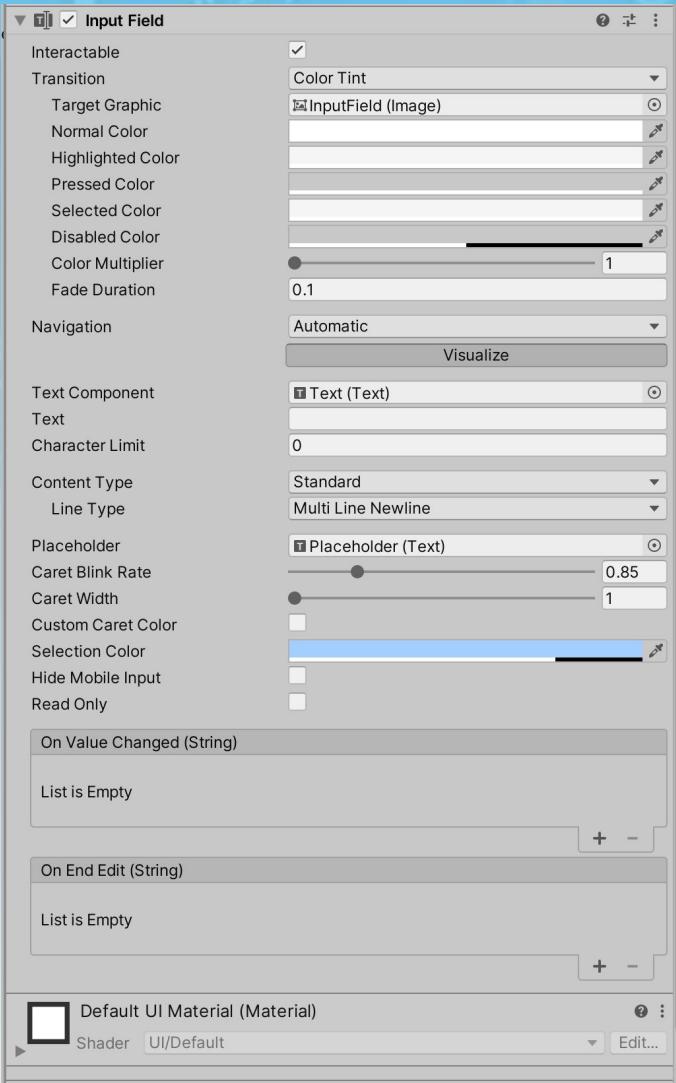


Los primeros componentes del GameObject padre “InputField”, ya los hemos visto y hemos trabajado con ellos. En esta ocasión, nos centraremos en el componente “Input Field”.



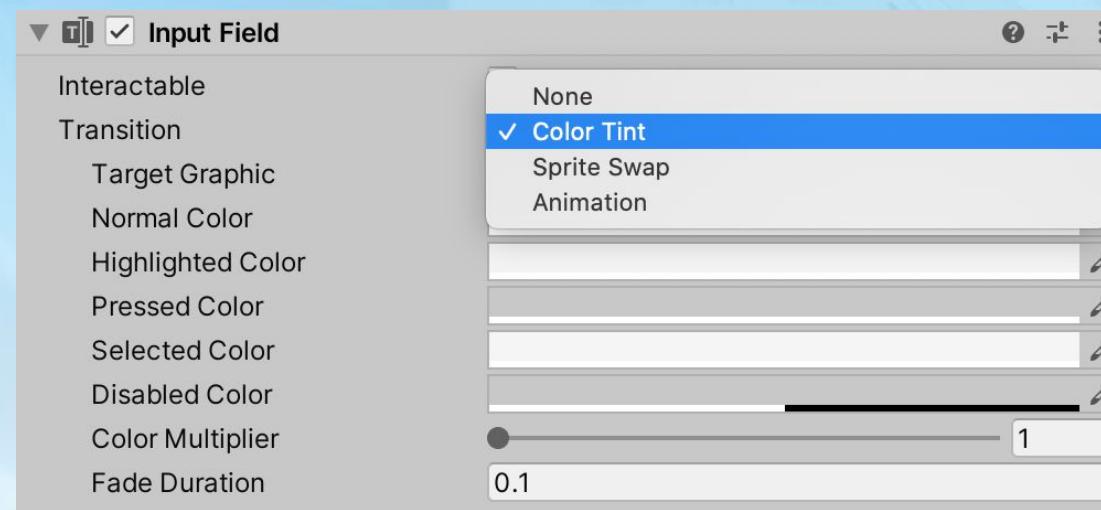
# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Input Field



Los parámetros de este componente son bastante intuitivos y con algunos ya hemos trabajado, como los primeros.

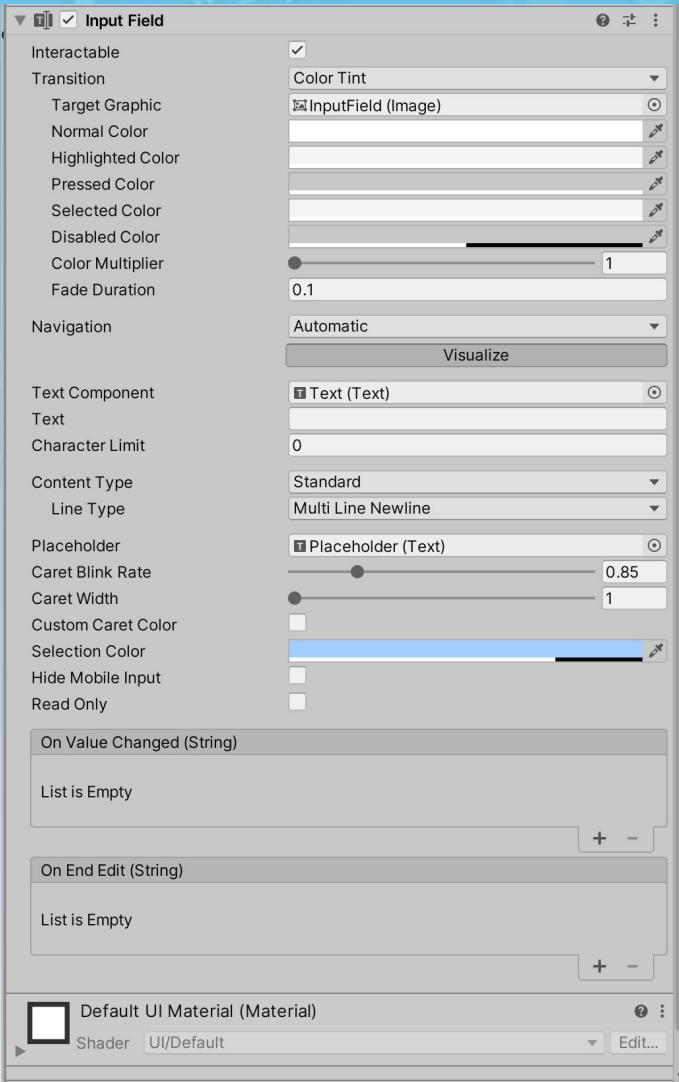
- **Interactable:** Valor Booleano para habilitar/deshabilitar el InputField.
- **Transition:** Este parámetro funciona exactamente igual que en los GameObject Buttons. Revisar la U.T.2.



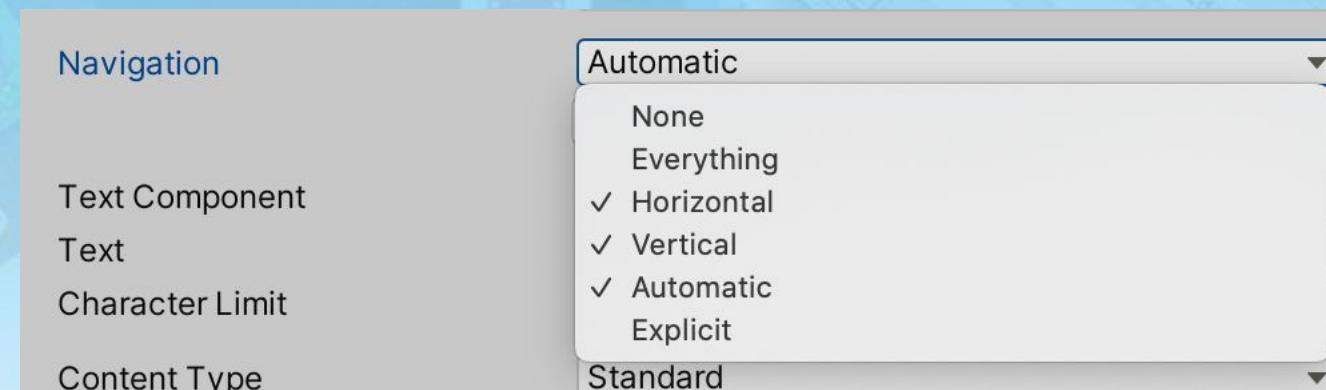
# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

## Componentes básicos.

### Input Field



- **Navigation:** Nos permite configurar cómo nos desplazamos entre los GameObjects situados dentro del panel. Tenemos varias formas de configurarlo: horizontal, vertical, automáticamente, ninguna y explícitamente.

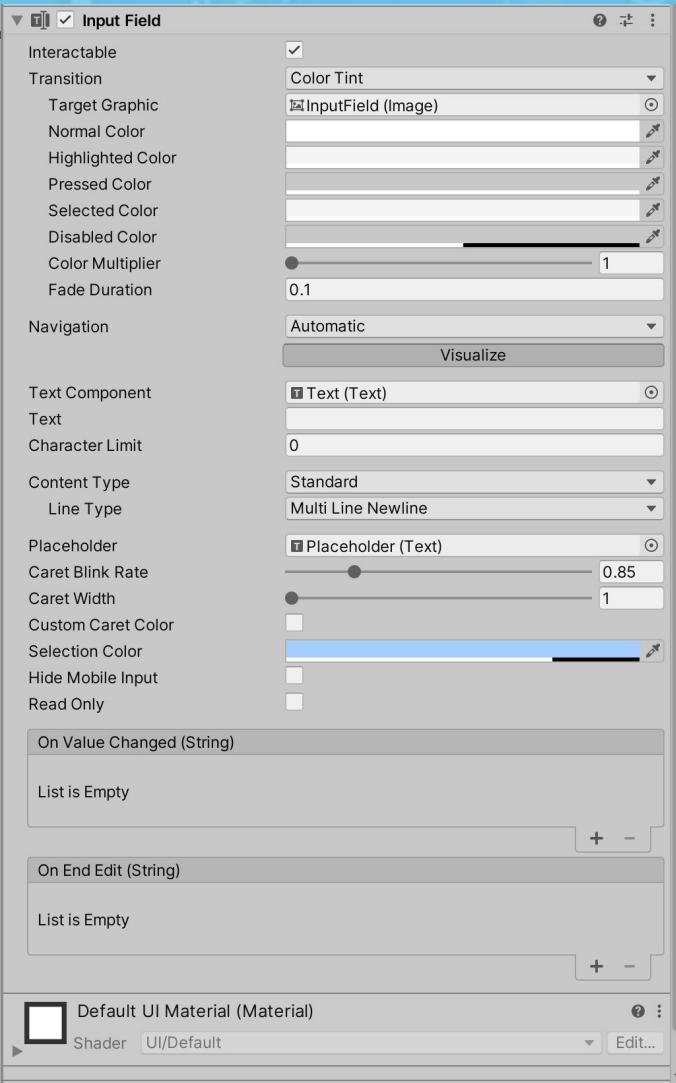


- **Visualize:** Muestra unas flechas en la ventana Scene, en la que se puede observar el orden establecido en la navegación desde las teclas de dirección del teclado.

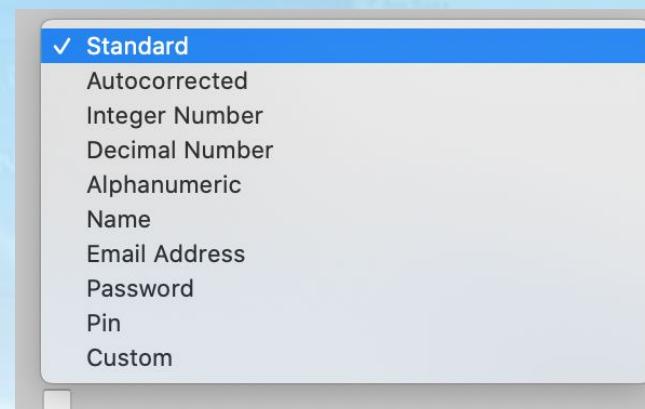
# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

## Componentes básicos.

### Input Field

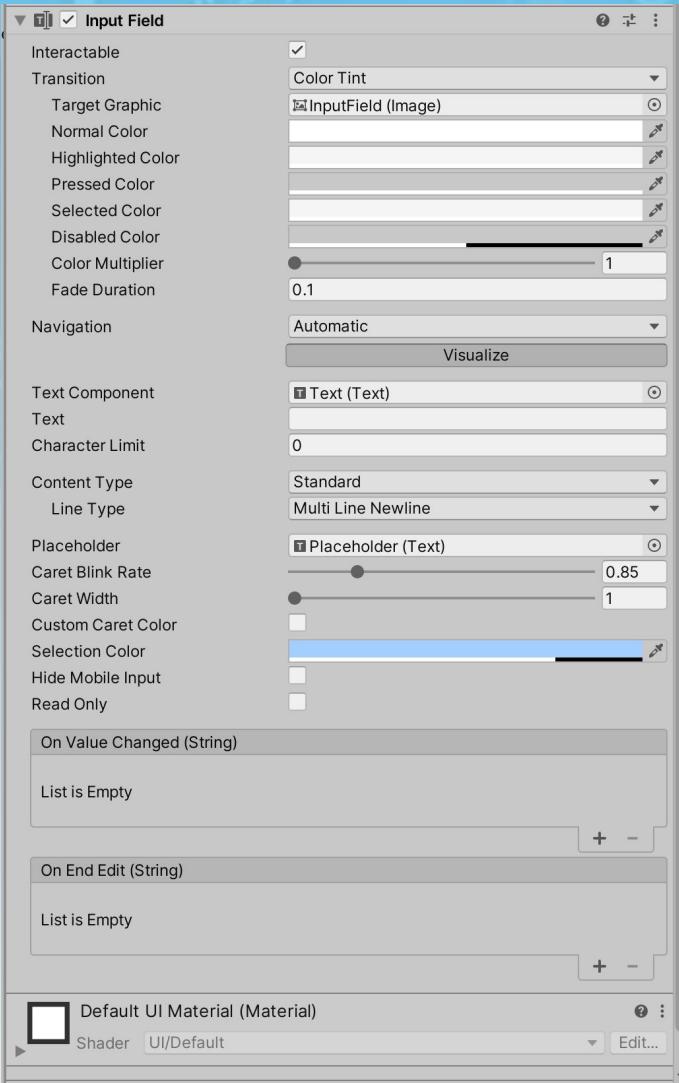


- **Text Component:** Se indica que GameObject de tipo Text, vamos a utilizar como contenedor de salida del texto.
- **Text:** En este campo, se guarda el texto introducido desde el proyecto.
- **Character Limit:** Podemos indicar el número de caracteres que se pueden introducir.
- **Content Type:** El tipo de dato que vamos a guardar en el Input Field.

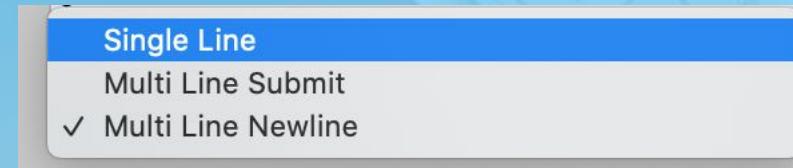


# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Input Field



- **Line Type:** Tipo de línea que queremos utilizar.

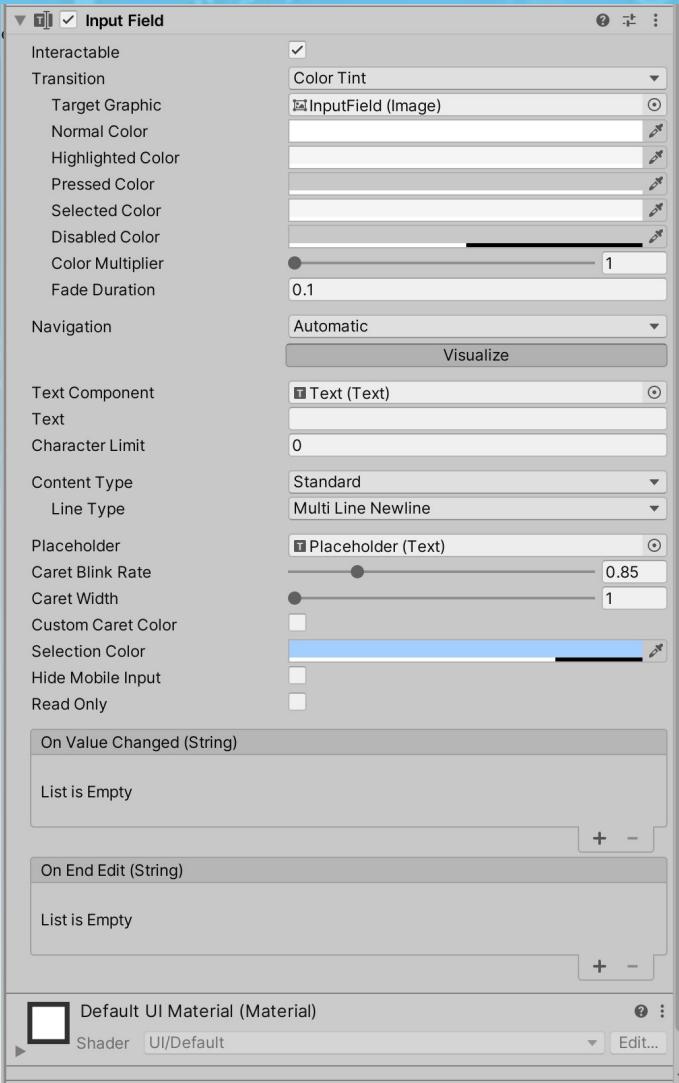


La diferencia entre “Multi Line Submit” y “Multi Line Newline” es que en la primera solo se crea una nueva línea cuando es necesario y en la segunda, podemos crear una línea pulsando enter.

- **Placeholder:** Esto lo veremos al trabajar con el GameObject Placeholder.
- **Caret Blink Rate:** Para indicar el número de parpadeos del cursor cuando el GameObject InputField esté activo.
- **Caret Width:** Especifica el ancho del cursor.

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

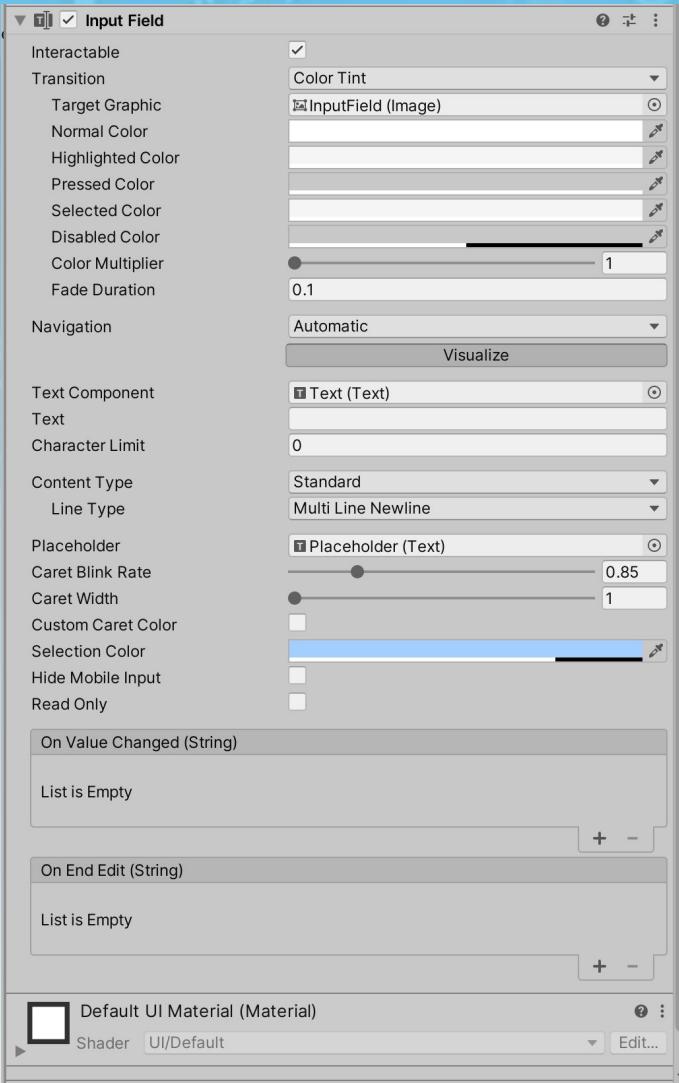
## Input Field



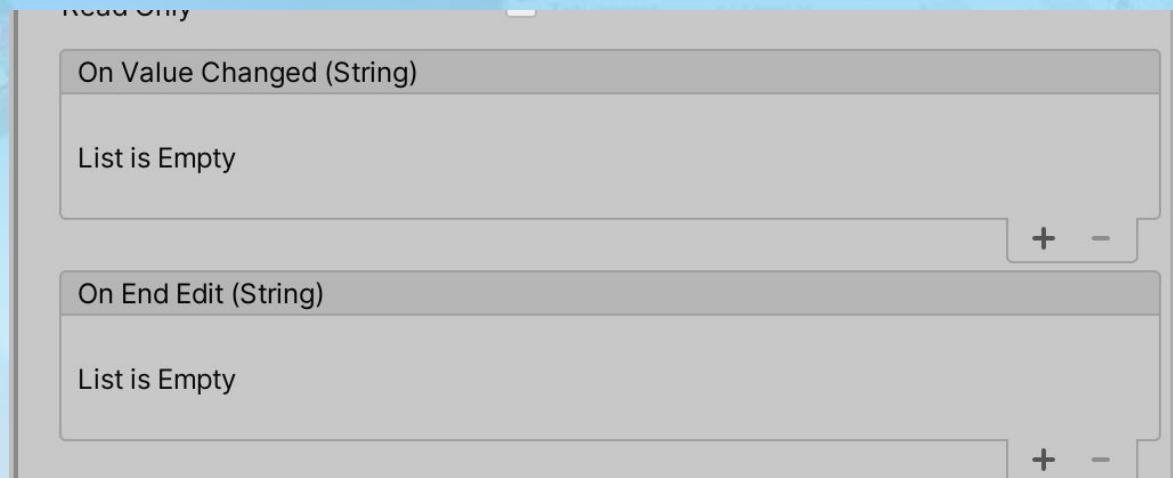
- **Custom Caret Color:** Personalizar el color del cursor de texto.
- **Selection Color:** Indica un color de selección del texto. Por defecto, el que se ve en la imagen.
- **Hide Mobile Input:** Esto solo funciona en dispositivos iOS.
- **Read Only:** No permitir que el usuario pueda modificar el valor del campo Text.

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

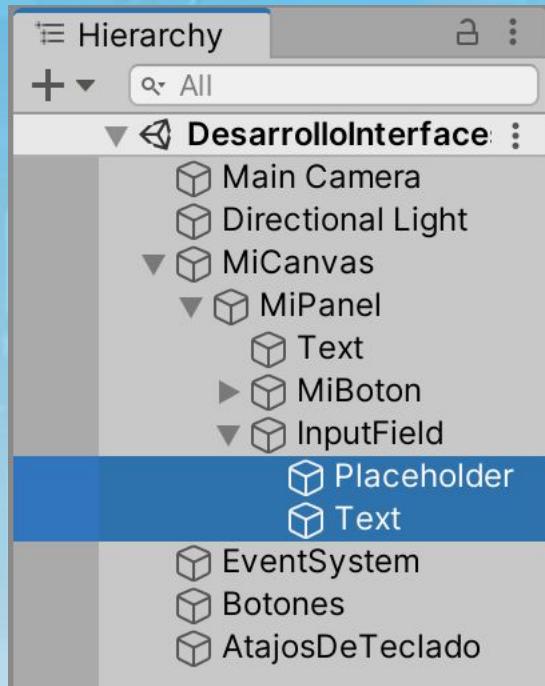
## Input Field



A continuación nos encontramos con dos parámetros para controlar los eventos Change y el End Edit. Estos parámetros funcionan igual que el evento OnClick de los GameObjects Buttons.



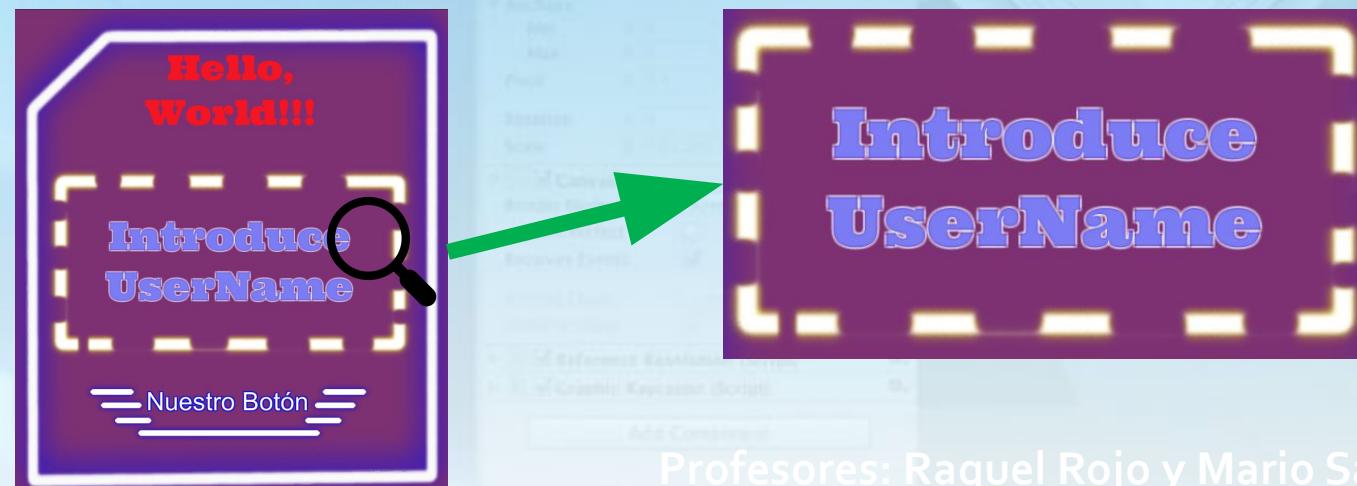
# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.



Veamos, ahora, los dos hijos del GameObject Input Field. Como se puede observar en la imagen, estos son:

- Placeholder
- Text

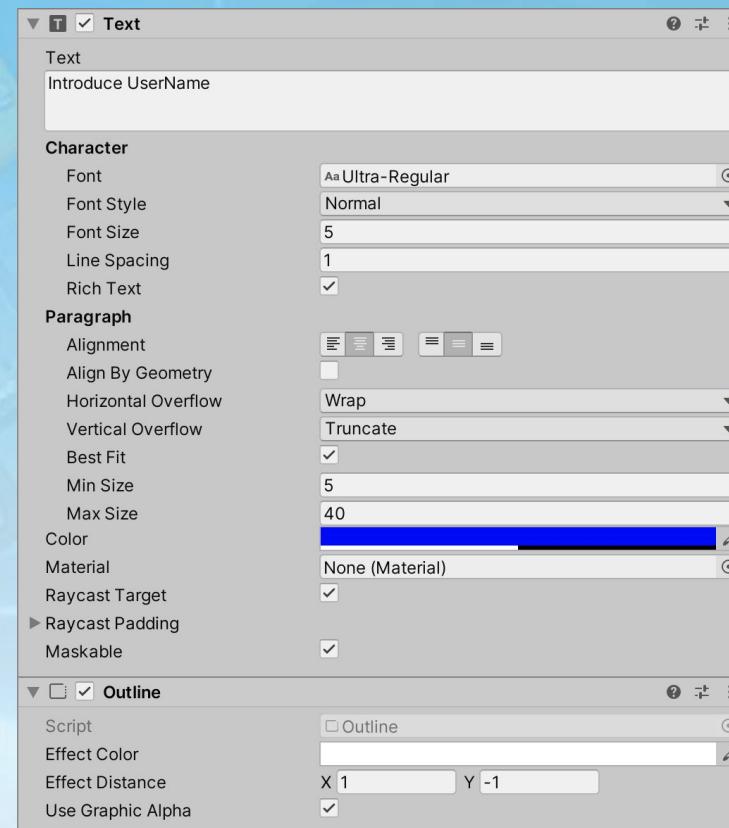
El **GameObject Placeholder** lo utilizamos para configurar el Texto que se muestra por defecto. En nuestro caso “Introduce UserName”.



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Input Field

En este caso, hemos configurado el Placeholder con un color azulado y le hemos añadido el componente "Outline" blanco. La configuración se muestra en la imagen.



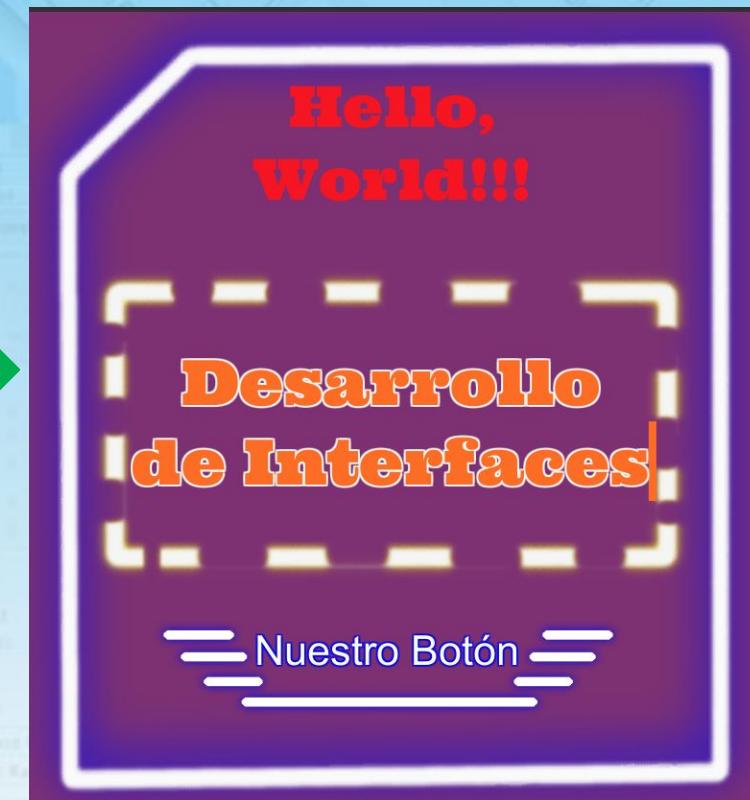
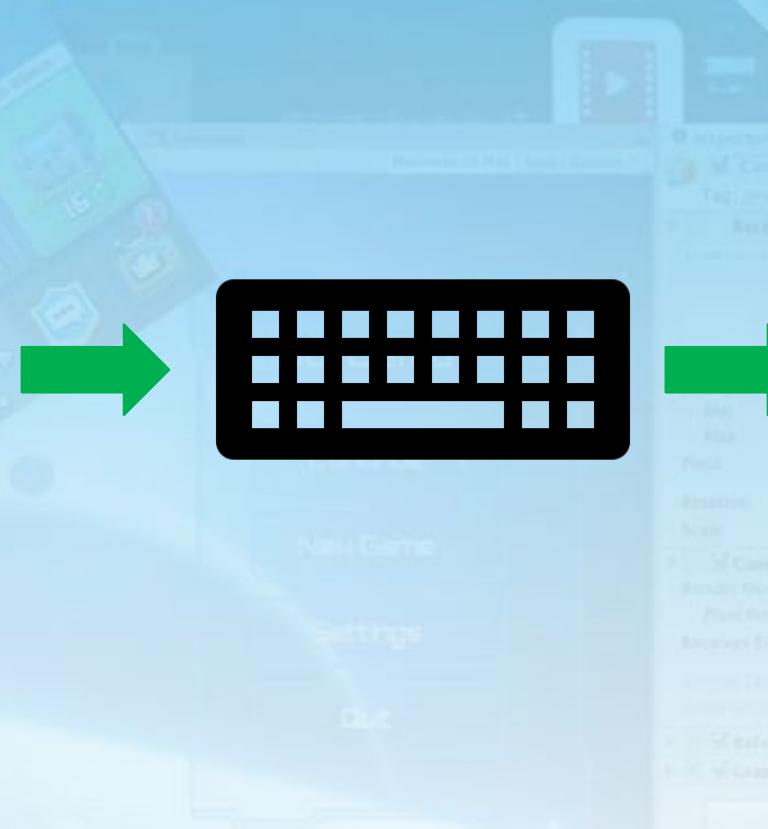
Los parámetros del componente “Text” y del componente “Outline”, han sido vistos en la U.T.2.

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

## Componentes básicos.

Input Field

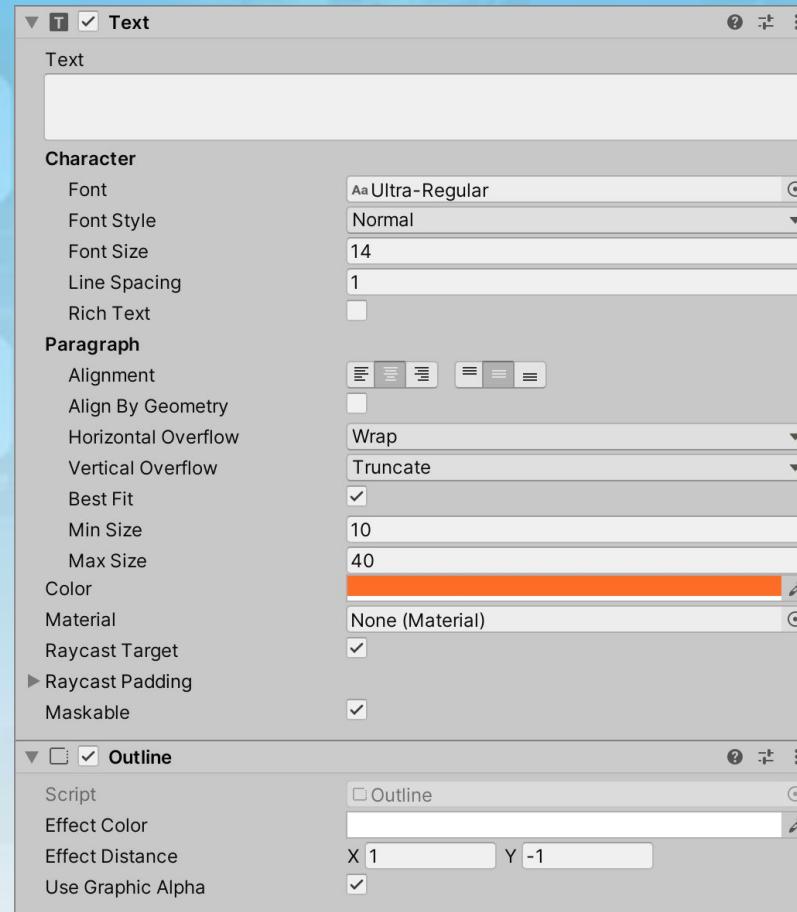
En el caso del GameObject hijo “Text”, lo que configuramos es el formato del texto que introduce el usuario. En nuestro caso, utiliza la fuente Ultra-Regular, con un color Naranja y un componente Outline de color blanco.



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Input Field

En la siguiente imagen se muestra como se ha configurado el componente Text del GameObject “Text”.



Los parámetros del componente “Text” y del componente “Outline”, han sido vistos en la U.T.2.

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Input Field

El código para acceder al texto es el que se muestra en la siguiente imagen.

```
* Description: Desde este script, creamos una
* variable de tipo ScrBoton y llamamos a sus
* métodos.
* Author: Raquel Rojo y Mario Santos.
* Palomeras Vallecas - Villablanca
*/
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

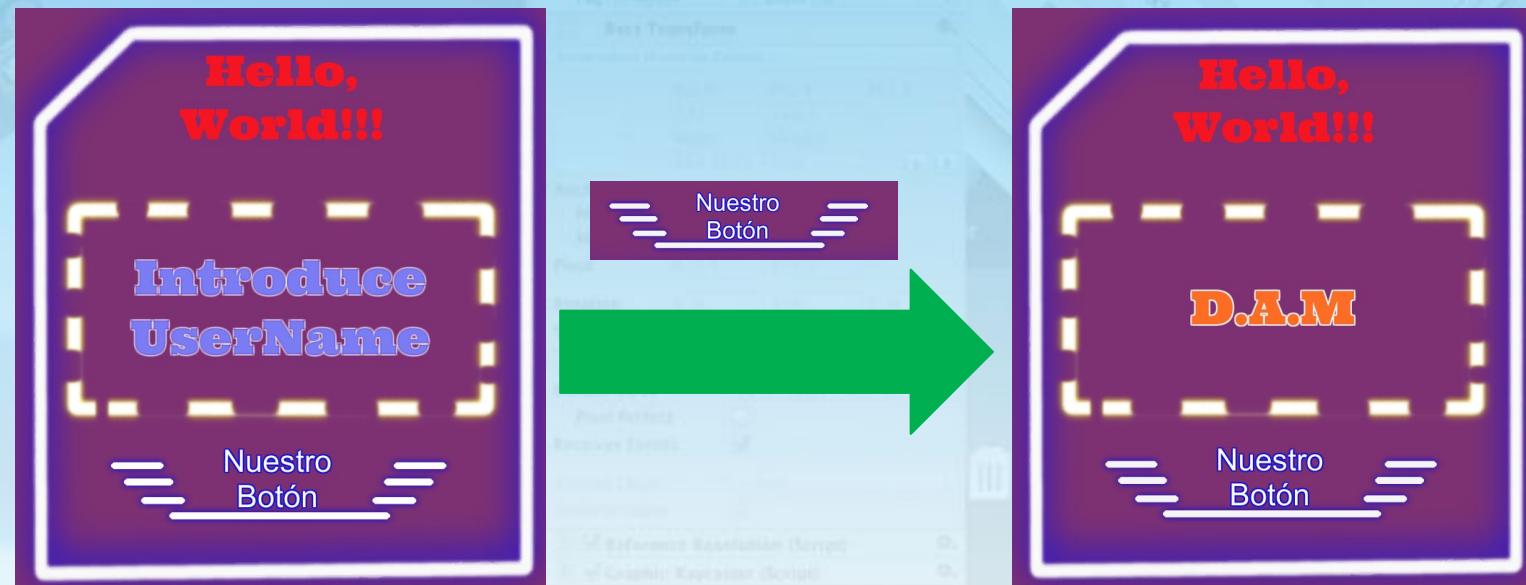
public class ScrNuestroBoton : MonoBehaviour
{
    //declaramos una variable de tipo ScrBoton
    public ScrBoton btnNuestroBoton;

    //declaramos la variable Private IfCampoDeTexto
    //Esta variable la vamos a utilizar para apuntar
    //Al GameObject de tipo InputField
    private GameObject IfCampoDeTexto;

    //Esta función la cargamos en el botón de nuestro
    //proyecto ejemplo.
    public void Input_Field()
    {
        //Indicamos a la variable al GameObject al que debe apuntar.
        IfCampoDeTexto = GameObject.Find("InputField");

        //Leemos/escribimos de su campo text.
        IfCampoDeTexto.GetComponent<InputField>().text = "D.A.M";
    }
}
```

La función "Input\_Field()" se cargará en nuestro proyecto ejemplo, y cuando lo pulsemos, aparecerá en el InputField la palabra "D.A.M".

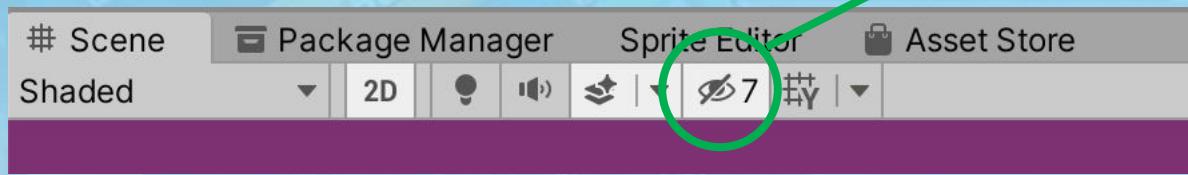


- El GameObject de tipo “Scrollbar” lo podemos utilizar para desplazar información (texto, imágenes, botones, etc) dentro de un contenedor.
- Se puede hacer un desplazamiento, horizontal, vertical u horizontal y vertical.
- Para trabajar con este GameObject, se deben de trabajar con varios componentes.
- Para ver como funciona, vamos a crear un panel nuevo que denominamos “PnlScrollBar” (se puede llamar como se quiera). Le damos la misma apariencia que el Panel con el que hemos estado trabajando hasta ahora.

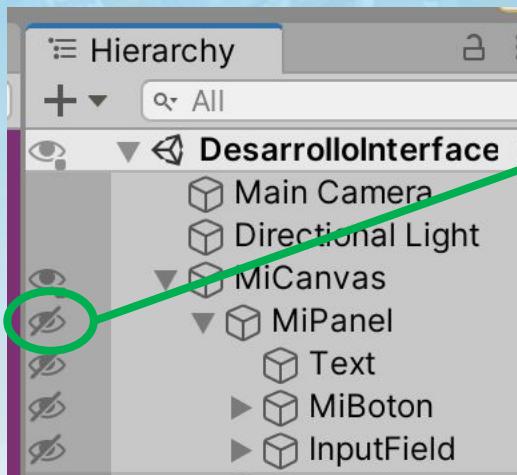
# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Scroll Bar

Para que no estén visibles los dos paneles en la ventana Scene, desactivar el panel “MiPanel” como se muestra en la imagen.



**Activar este botón**

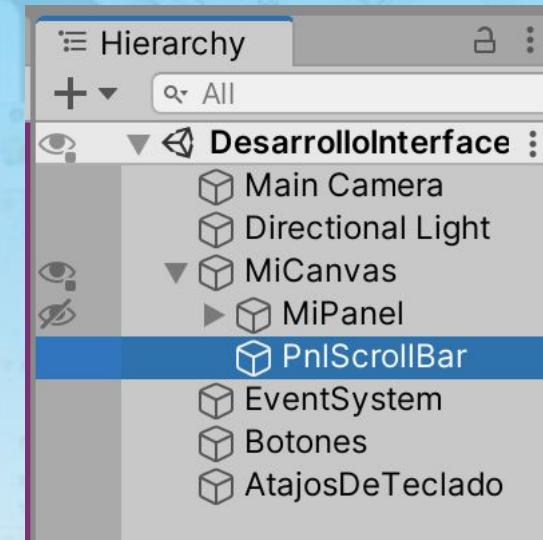
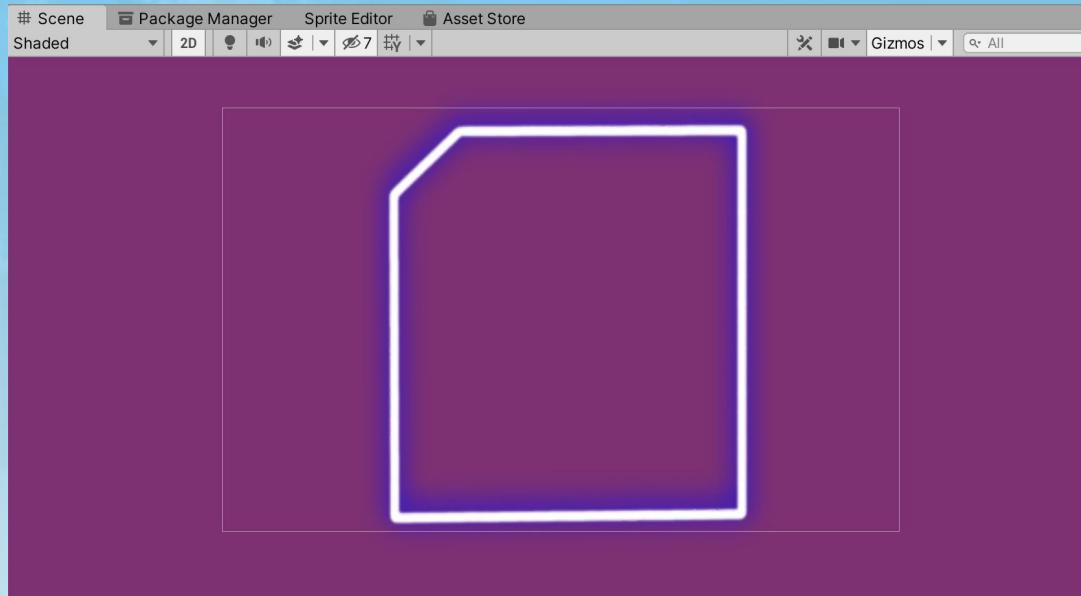


**Cliqueamos sobre este botón hasta que aparezca el ojo tachado. De esta forma ocultamos en panel en la ventana Scene.**

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Scroll Bar

Una vez creado nuestro panel “PnlScrollBar”, la ventana Scene y la ventana Hierarchy quedan como se muestra en la imagen.

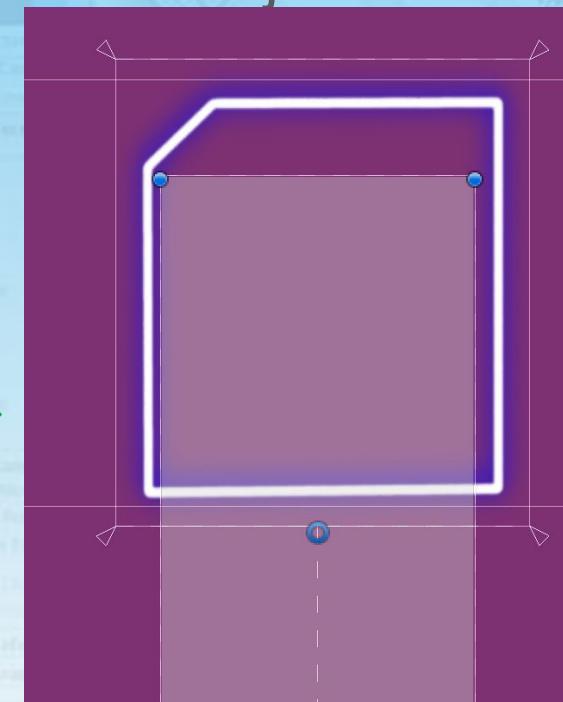
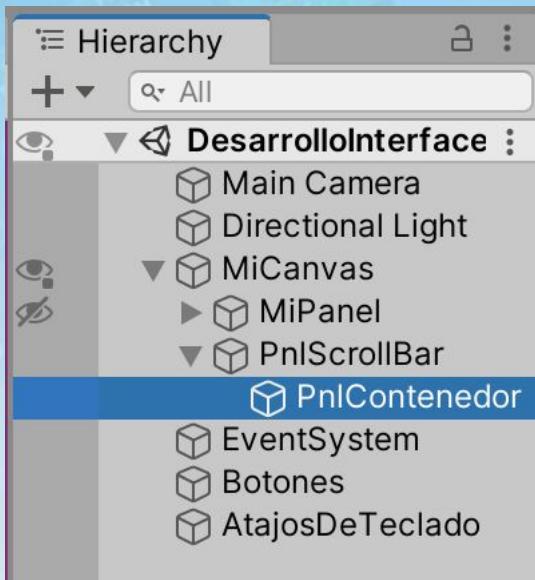


Fijarse que el panel PnlScrollBar es hijo del Canvas “MiCanvas”.

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Scroll Bar

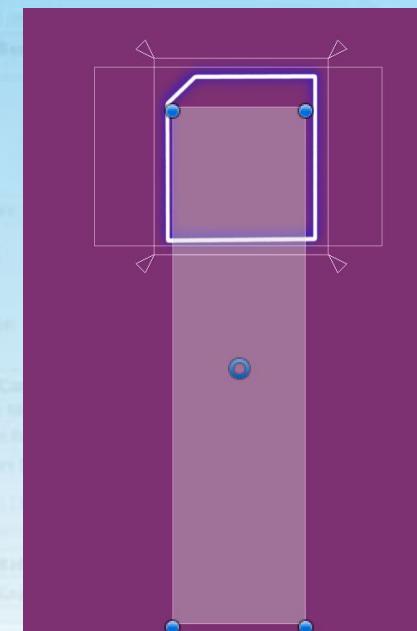
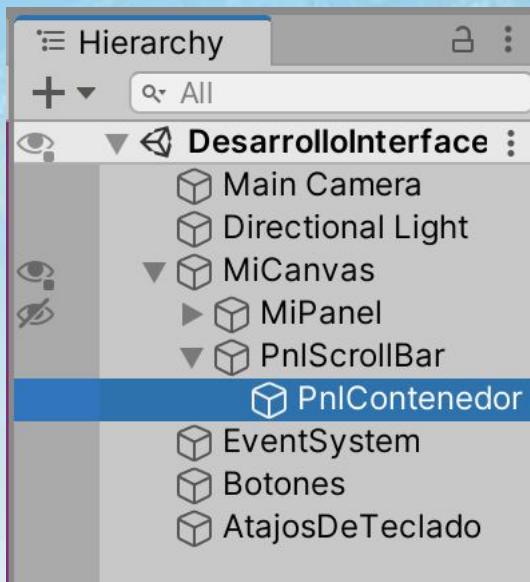
- El siguiente paso es crear el contenedor que va a tener la información que queremos que se muestre cuando nos desplazamos de arriba abajo o de derecha a izquierda.
- Este contenedor va a ser un GameObject de tipo "Panel" denominado "PnlContenedor", y debe estar emparentado al GameObject de tipo "Panel", denominado "PnlScrollBar".



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Scroll Bar

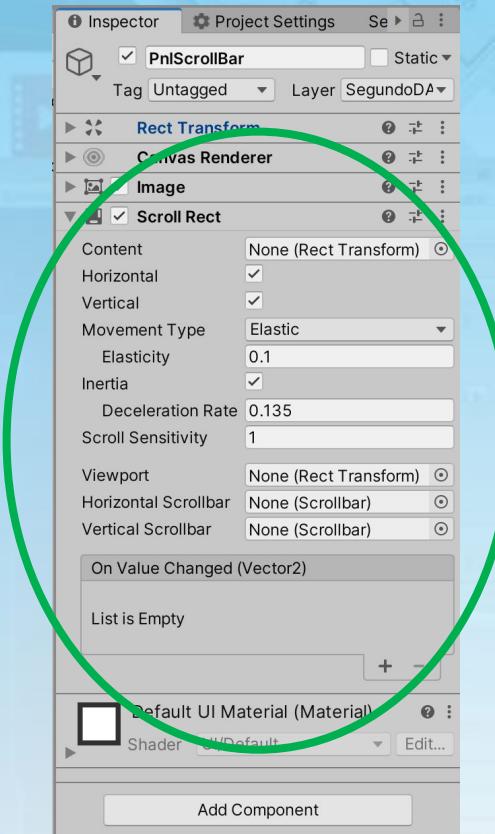
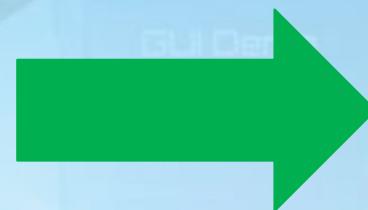
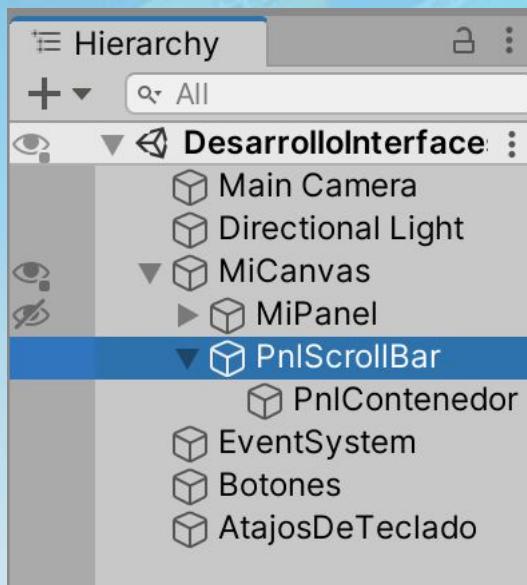
- En nuestro ejemplo, que es desplazar texto de arriba abajo, debes de escalar el GameObject de tipo “Contenedor” de tal forma que entre todo el texto que queremos que se vea. Hay que recordar que únicamente se va a ver el texto en el área delimitada por el GameObject “PnlScrollBar”.
- En este ejemplo, vamos a hacer el GameObject “PnlContenedor”, mucho más largo que el GameObject “PnlScrollBar”.



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Scroll Bar

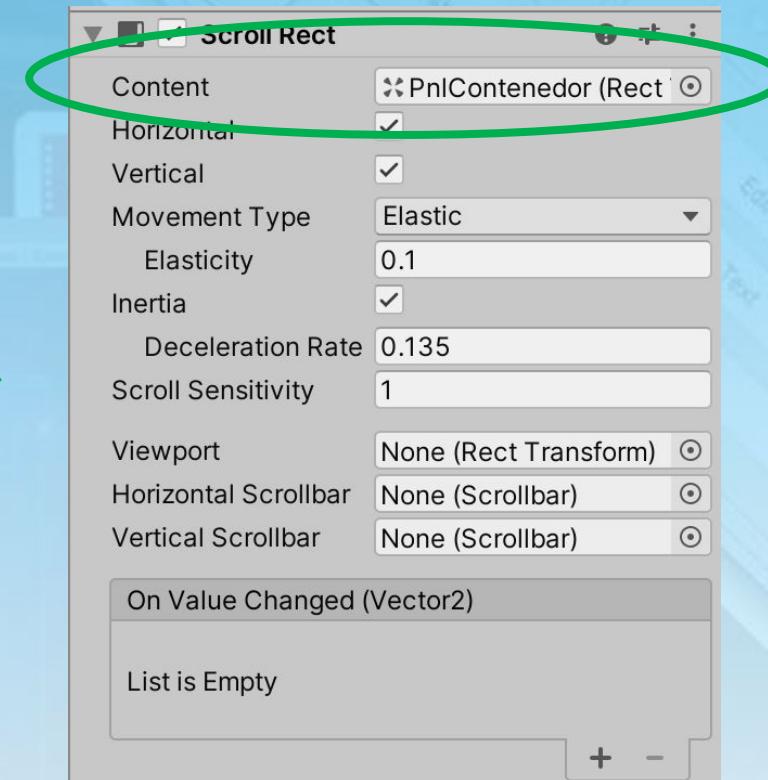
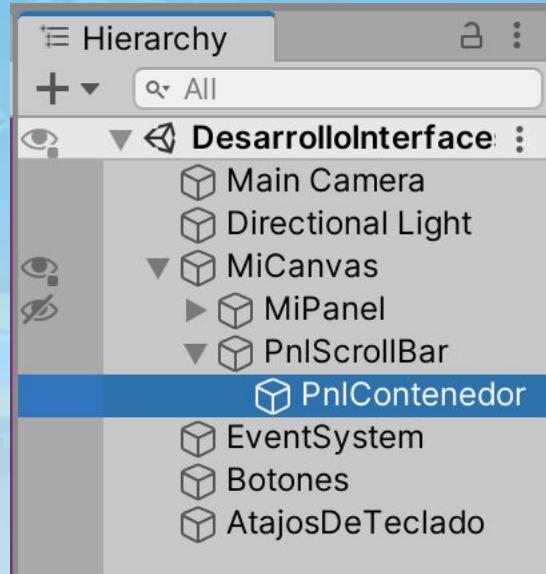
- Seleccionamos el GameObject “PnlScrollBar” y vamos a su ventana “Inspector”.
- Añadimos un nuevo componente de tipo “Scroll Rect”.



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Scroll Bar

- Arrastramos el GameObject “PnlContenedor” dentro del apartado “Content” del componente “Scroll Rect”.

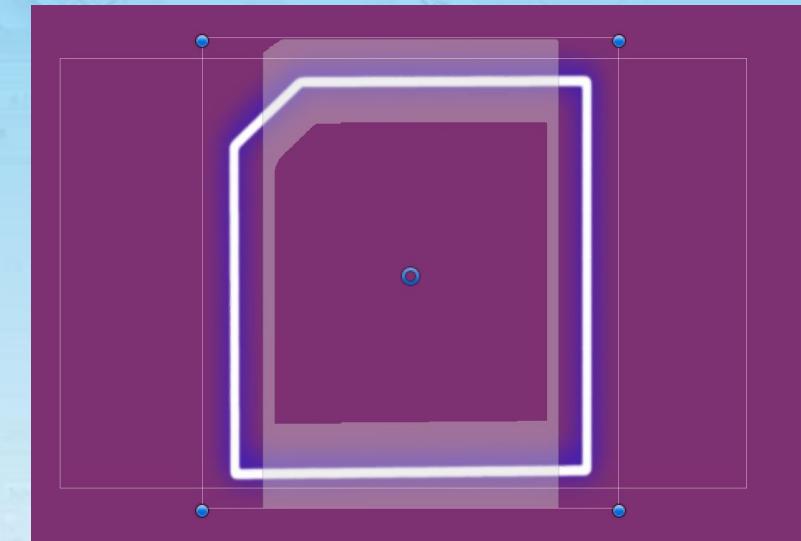
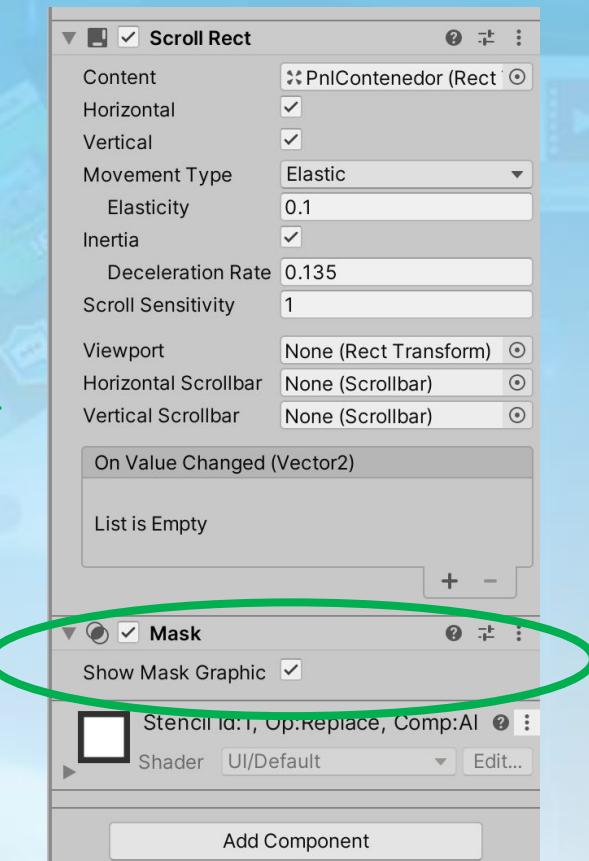
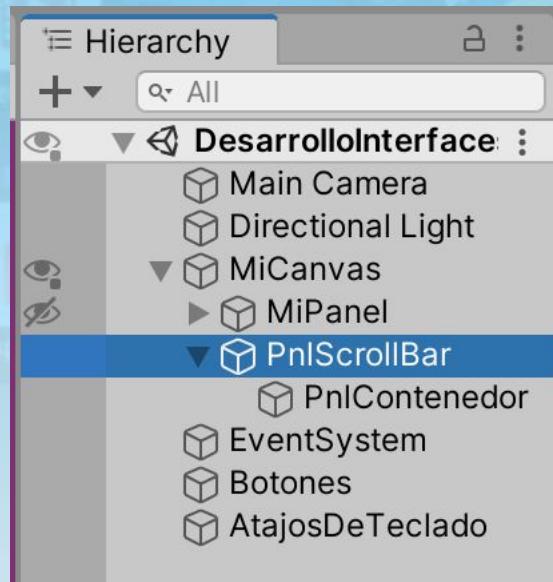


**Para introducir el PnlContenedor dentro del componente Scroll Rect, debe estar seleccionado el GameObject PnlScrollBar**

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Scroll Bar

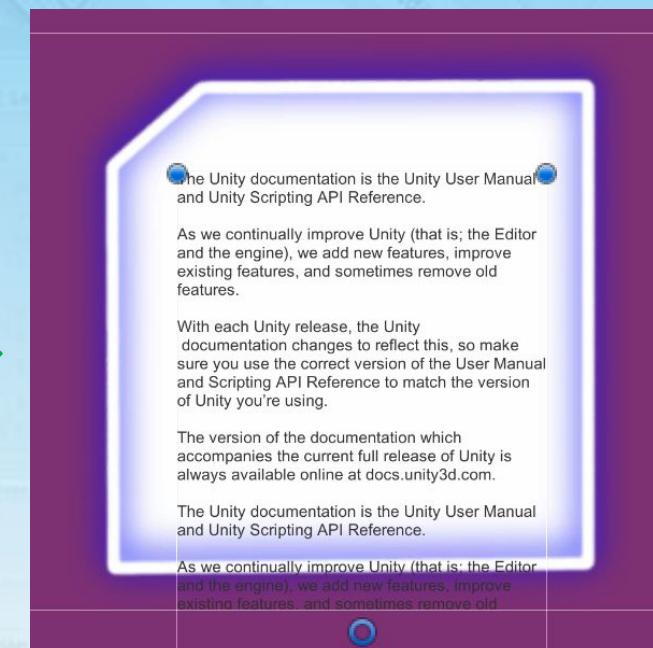
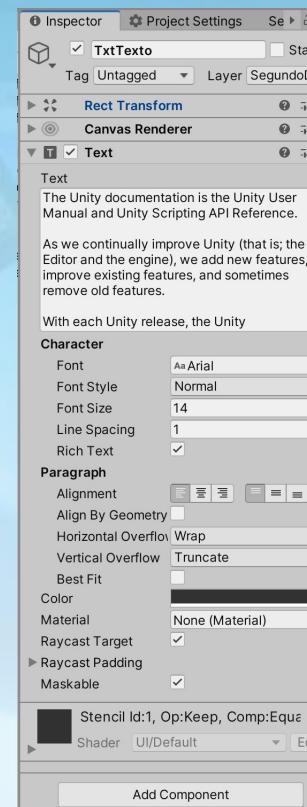
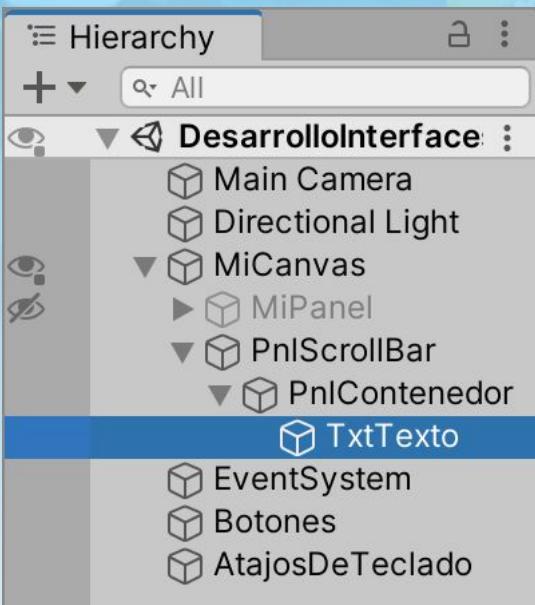
- Para hacer que la información, únicamente se vea en nuestra área delimitada por el GameObject “PnlScrollBar”, debemos añadirle a este, un componente de tipo “Mask”



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Scroll Bar

- Introducimos un GameObject de tipo “Text”, que denominamos “TxtTexto”, como hijo del GameObject “PnlContenedor”.
- Dentro “TxtTexto”, introducimos el texto que queremos que se visualice cuando desplazamos el ScrollBar.

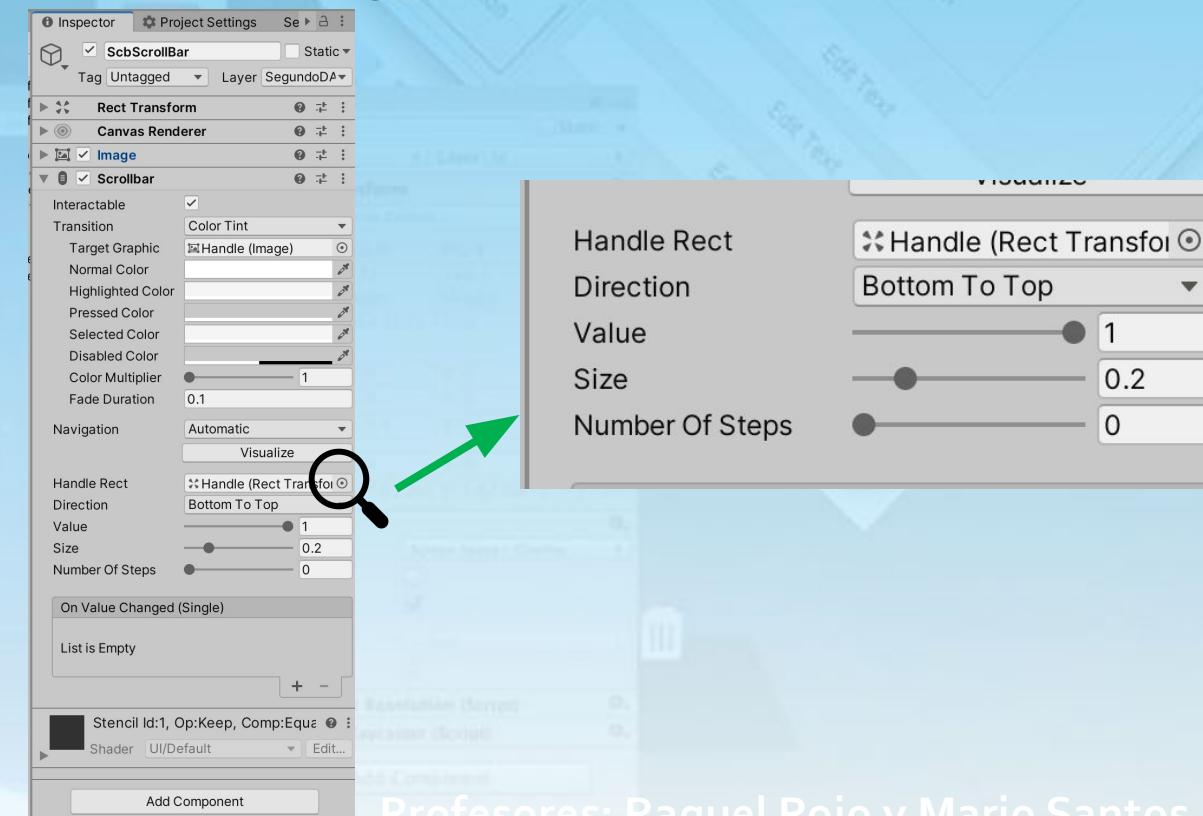
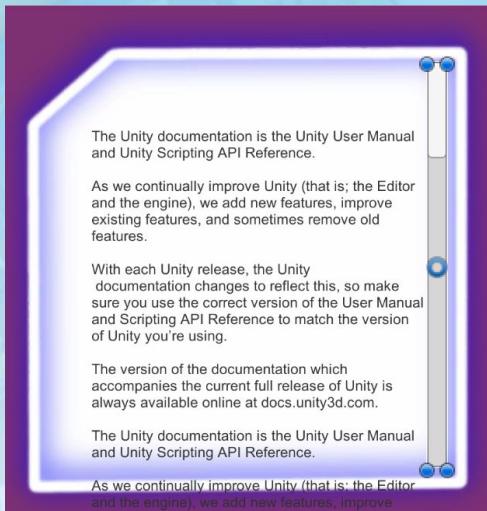
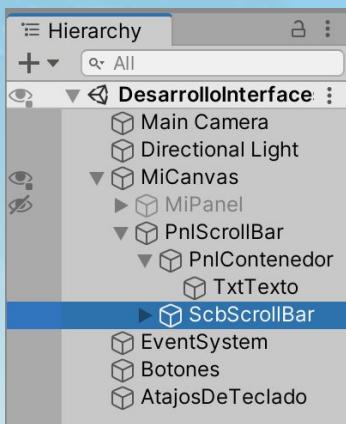


# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

## Componentes básicos.

### Scroll Bar

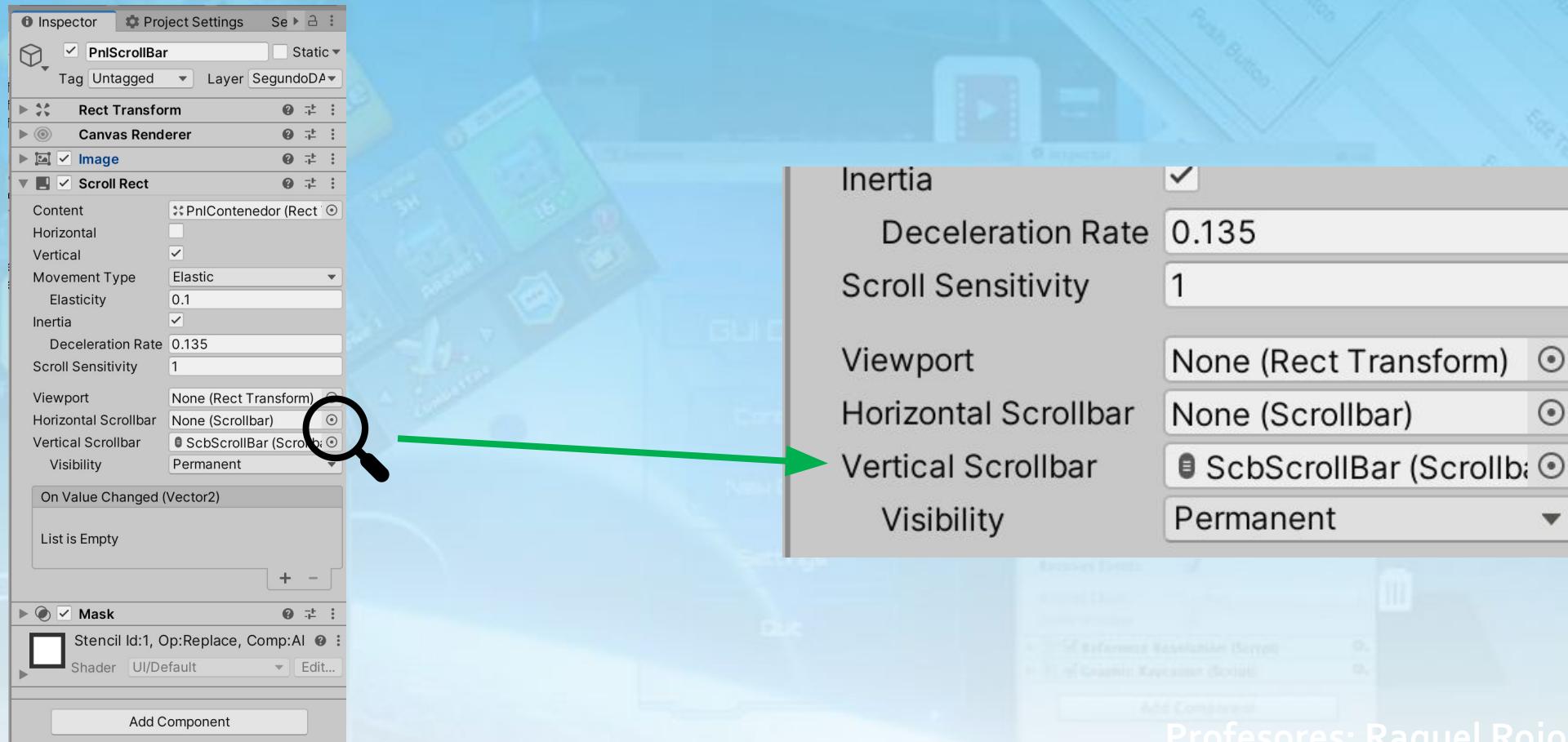
- Debemos emparentar dentro del GameObject “PnlScrollBar”, un GameObject de tipo “Scrollbar” que vamos a denominar “ScbScrollBar”.
- En nuestro ejemplo, que es desplazar texto de arriba abajo, “ScbScrollBar” se debe de configurar como se muestra en la figura.



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Scroll Bar

- Por último, debemos indicarle al componente "Scroll Rect", del GameObject "PnlScrollBar", el GameObject de tipo "Scrollbar" que vamos a utilizar

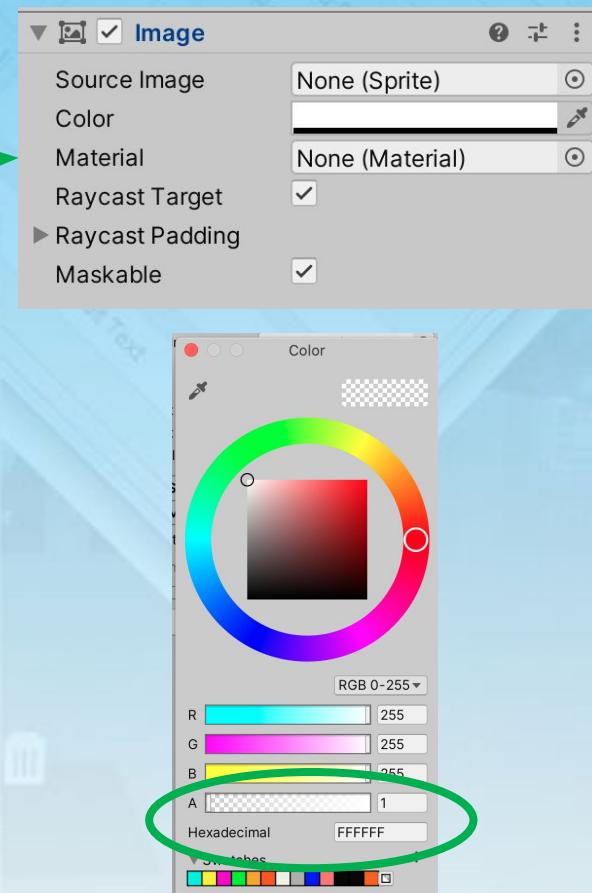
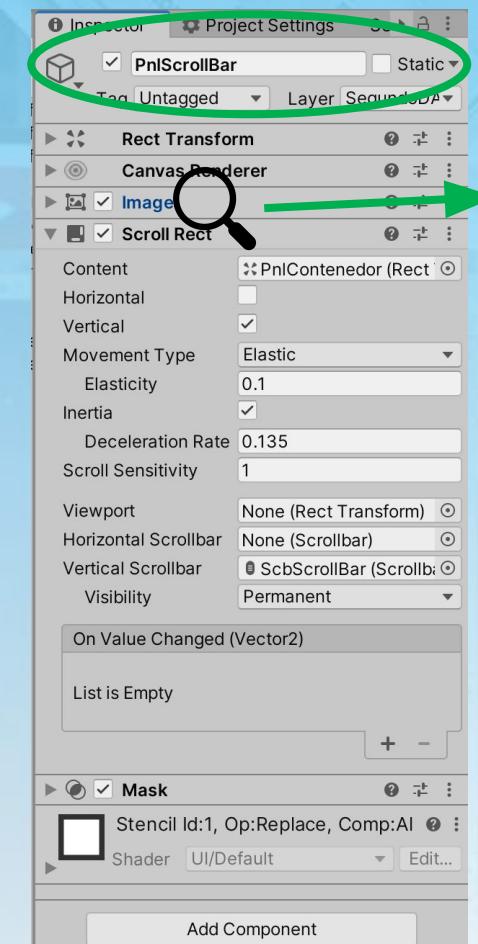
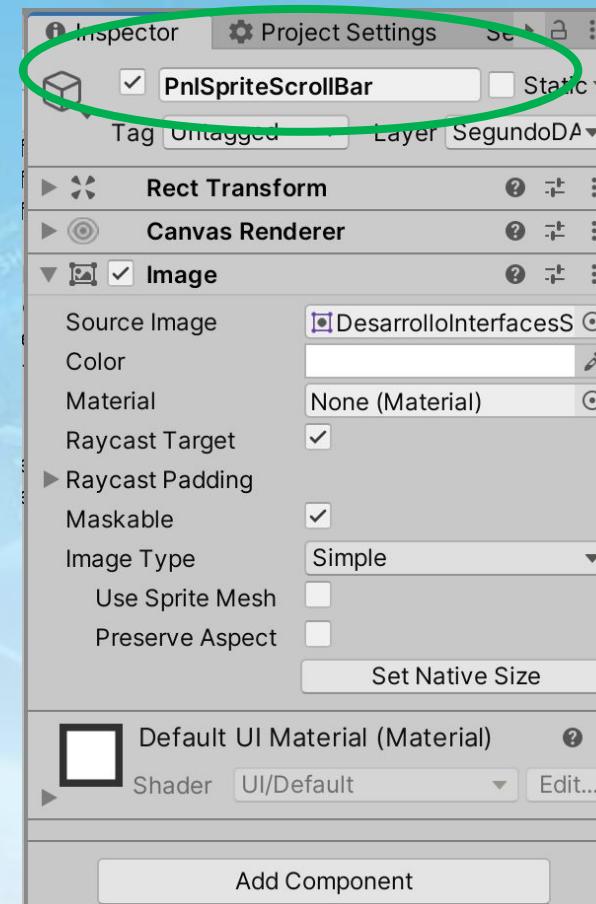
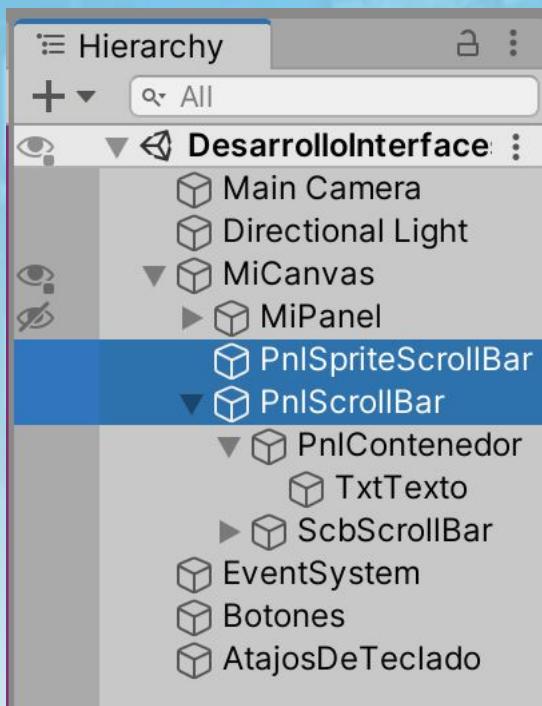


# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

## Componentes básicos.

### Scroll Bar

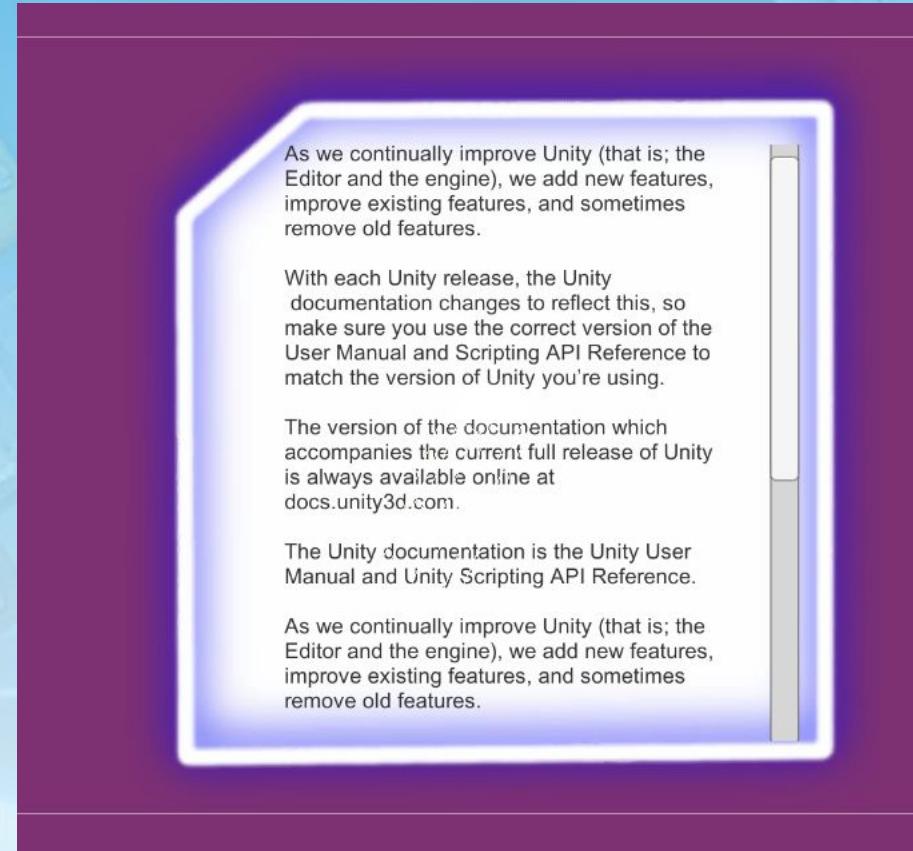
Al utilizar un Sprite con transparencias, lo más fácil es crear dos paneles, en uno configuraremos el ScrollBar y en el otro el Sprite.



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

Scroll Bar

El resultado final es el que se muestra en la imagen.



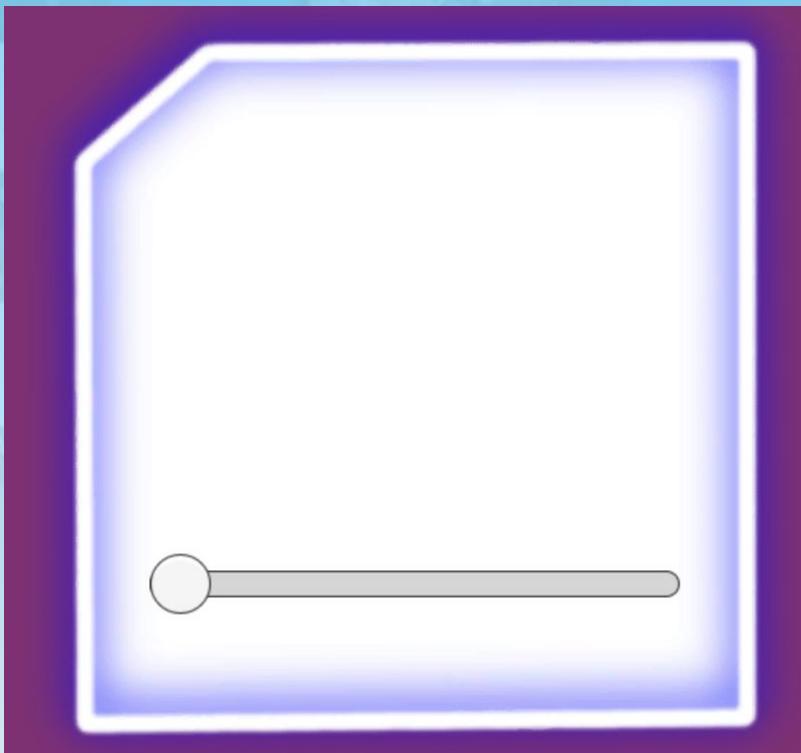
Junto a esta presentación se adjunta el paquete **UT3\_Presentacion** utilizado para la explicación.

Profesores: Raquel Rojo y Mario Santos.

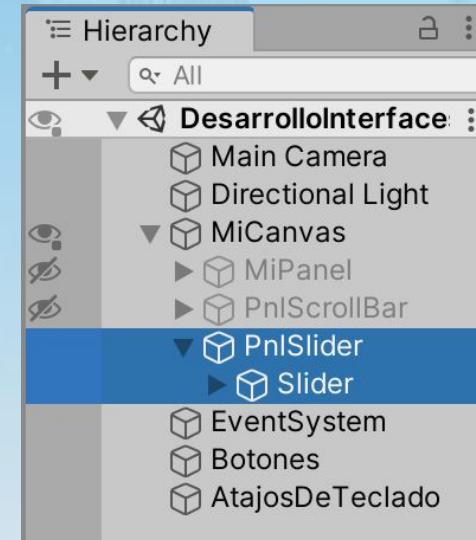
# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

Slider

Un componente de tipo Slider es un tirador que se desplaza por una barra (horizontal o vertical) entre dos valores dados, un mínimo y un máximo.



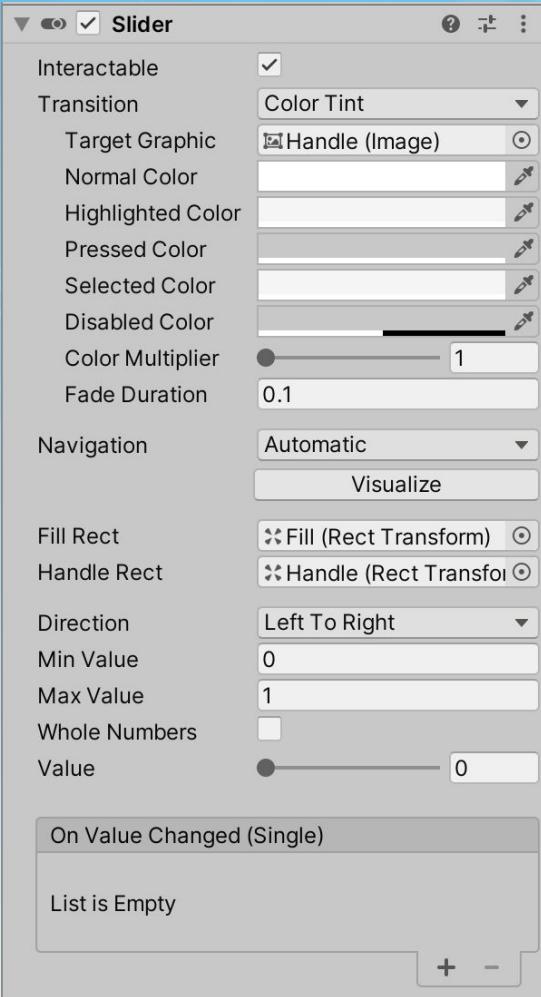
En este caso hemos añadido dentro del Canvas un panel “PnlSlider” y dentro hemos añadido el componente “Slider”.



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Slider

Vamos a fijarnos en la ventana Inspector del componente “Slider”.



Los primeros parámetros de configuración son los que ya hemos utilizado en otros componentes:

- Interactable.
- Transition.
- Navigation.

Los parámetros “Fill Rect” y “Handle Rect” los veremos más adelante, y nos vamos a centrar en:

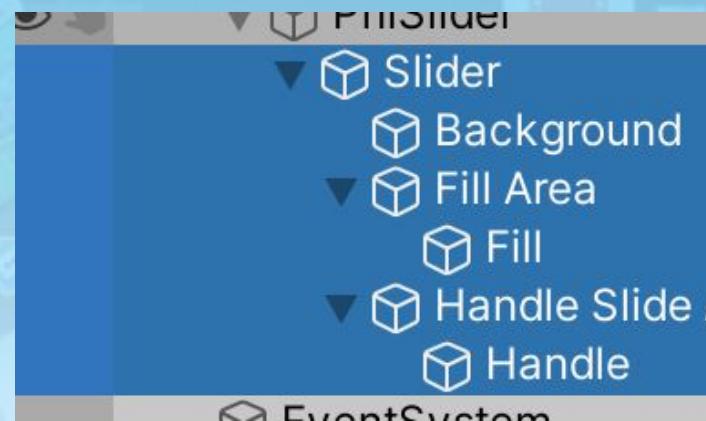
- Direction: horizontal, vertical, de derecha a izquierda, viceversa, etc.
- Min Value: La cota inferior.
- Max Value: La cota superior.
- Whole Numbers: El tirador se desplaza en valores enteros.
- Value: Valor en el que va a estar posicionado el tirador.



**El parámetro que más vamos a utilizar es el de “Value”**

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

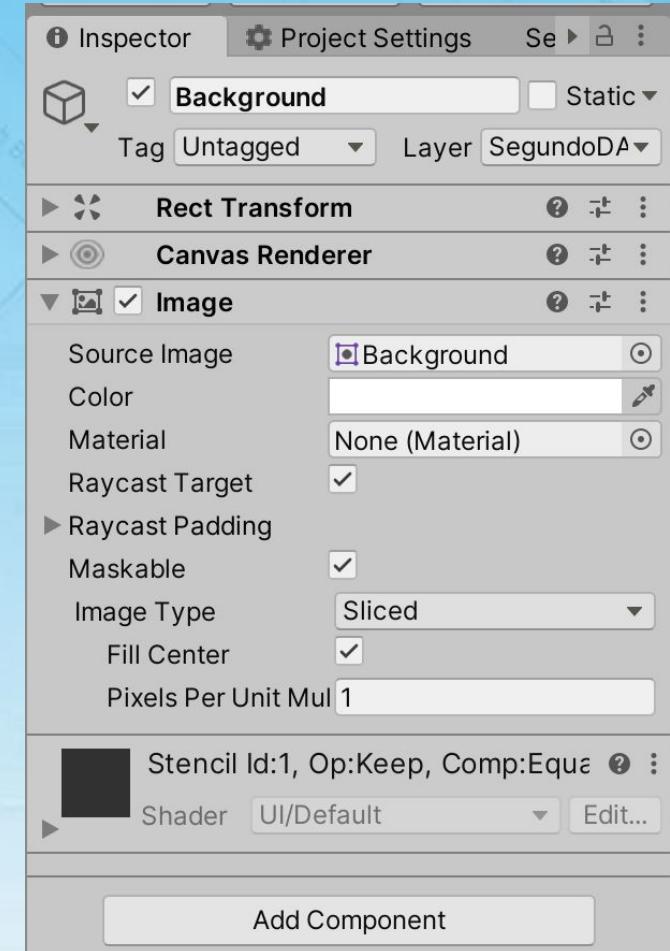
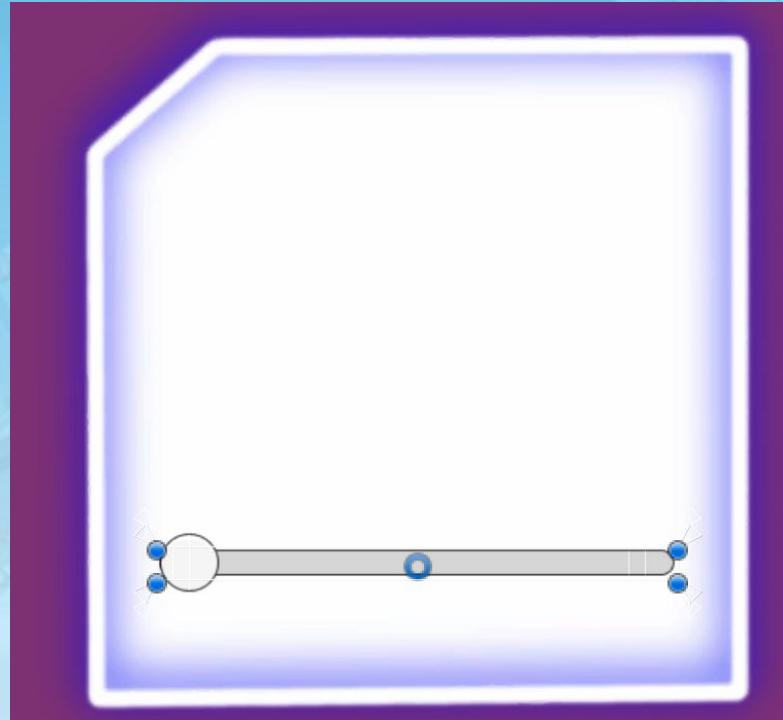
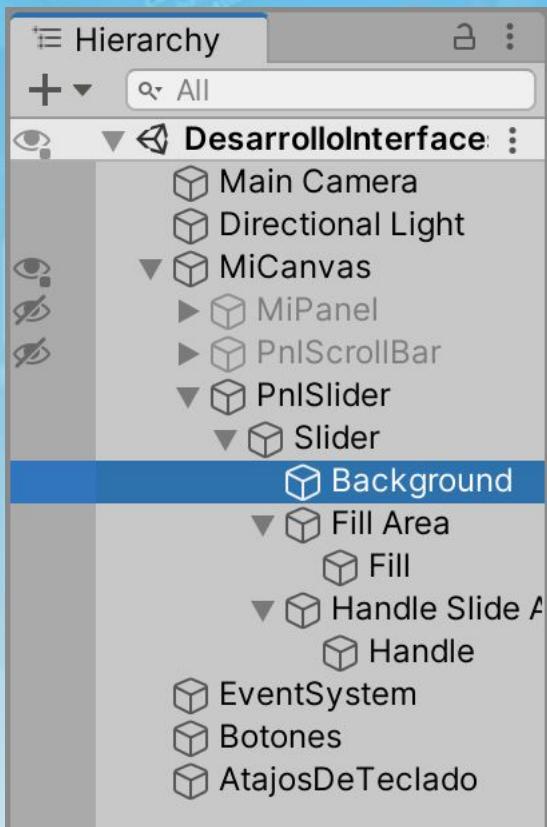
Cuando insertamos un GameObject o componente del Canvas, de tipo “Slider”, se genera un GameObject padre denominado Slider y los GameObjects hijos que se muestran en la imagen.



El primer GameObject hijo que nos encontramos es el de “Background”. Desde Background podemos configurar el color, la apariencia, etc, de la línea (por defecto en gris), por la que se mueve el tirador.

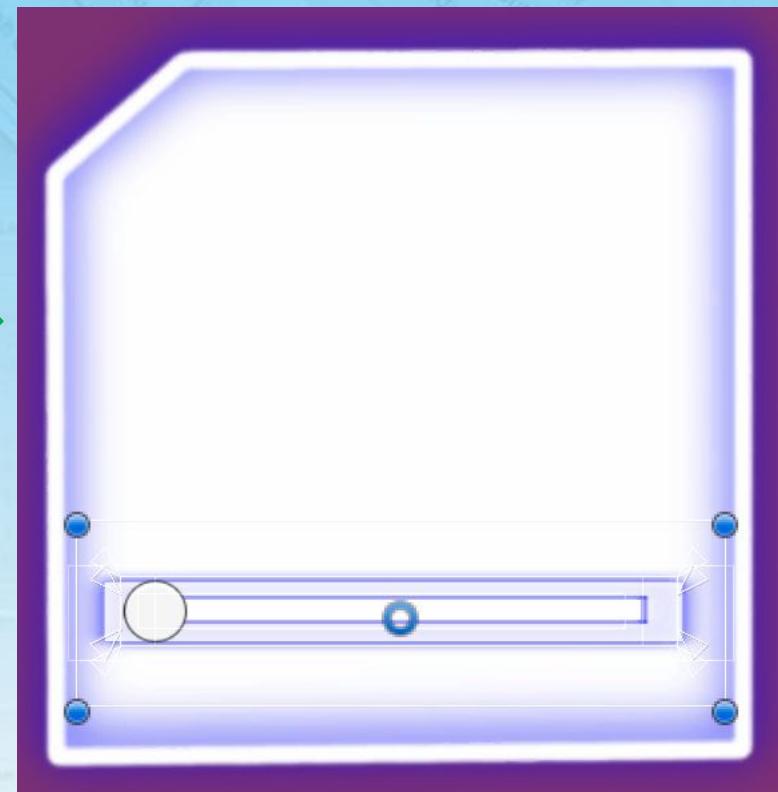
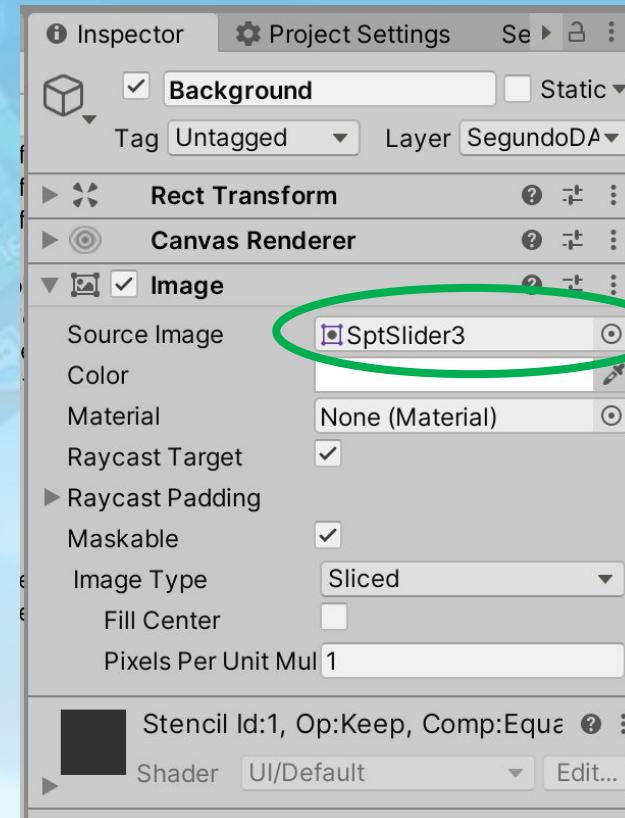
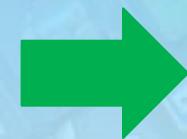
# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

Slider



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

Desde este GameObject hijo, podemos crear un Sprite y personalizarlo de la forma más adecuada para nuestro proyecto, por ejemplo:



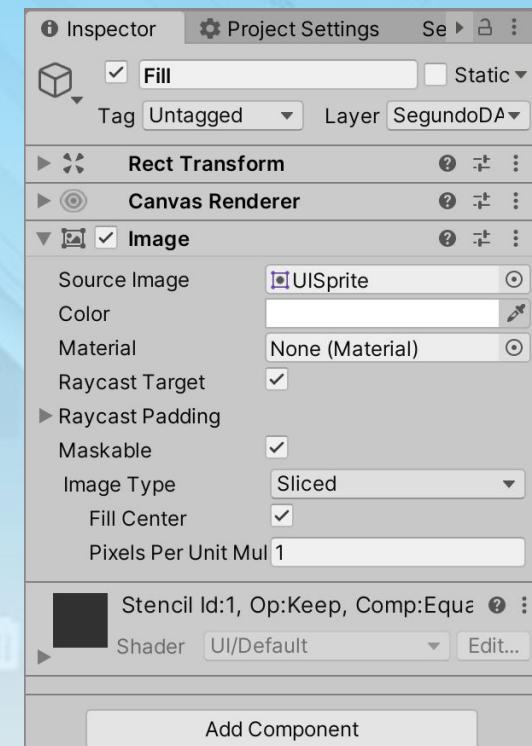
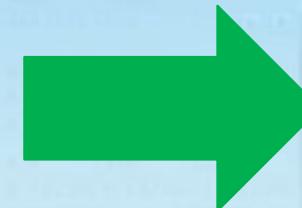
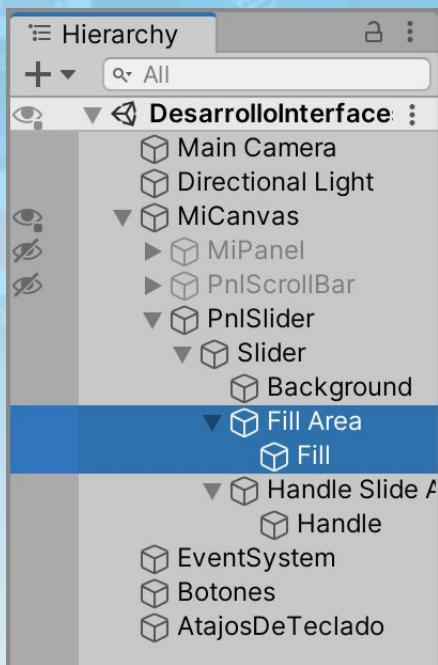
Sprite SptSlider3

 Tenéis acceso a estos Sprites dentro del paquete U.T.3\_Ejemplo.unitypackage

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

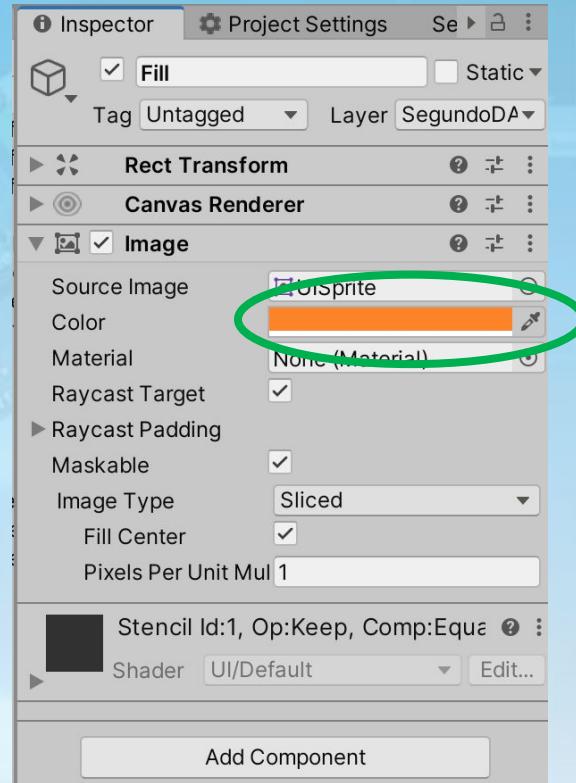
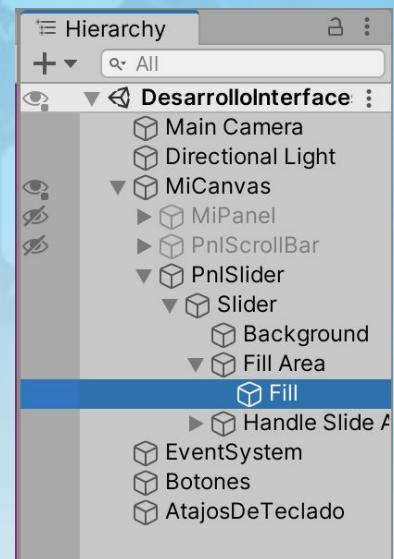
## Slider

El siguiente GameObject hijo que nos encontramos es el que parametriza el área de relleno, “Fill Area”, que se genera al desplazar el tirador. Este GameObject hijo, tiene a su vez otro hijo que es el que contiene los parámetros que vamos a modificar para modificar esta área.



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

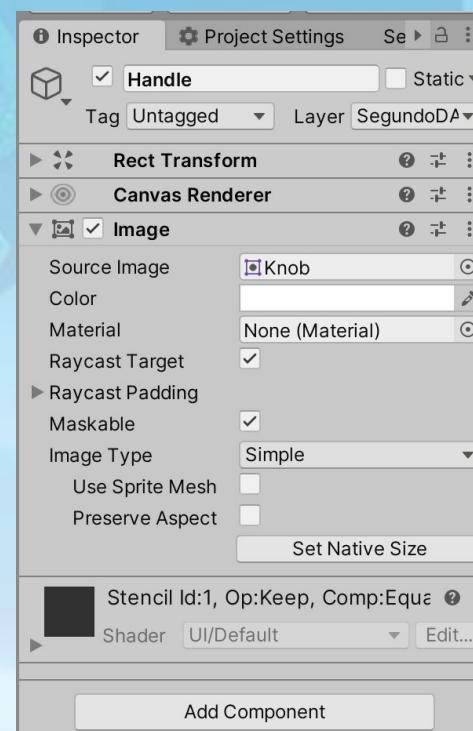
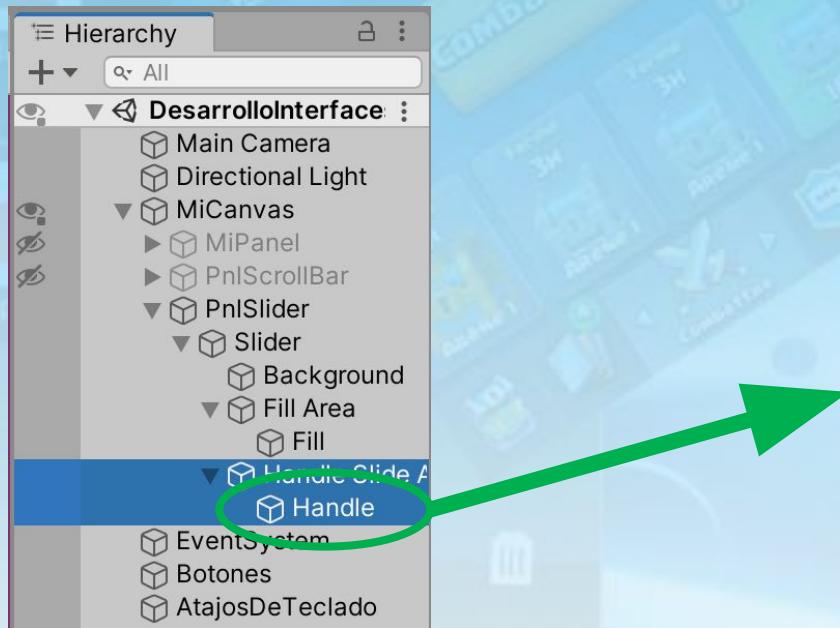
Vamos a configurar que esta área de relleno “Fill”, sea de color naranja, para ello seguir los pasos que se muestran en las imágenes.



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

## Slider

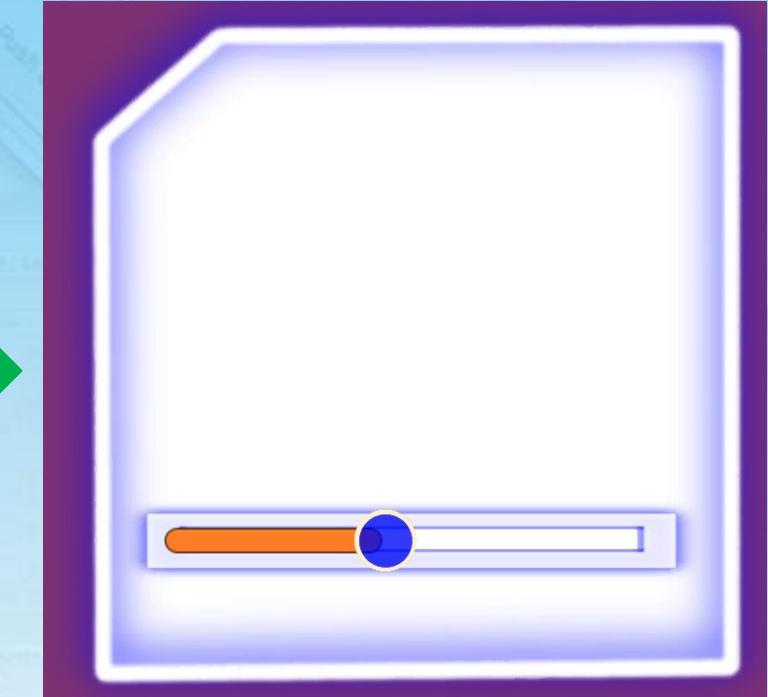
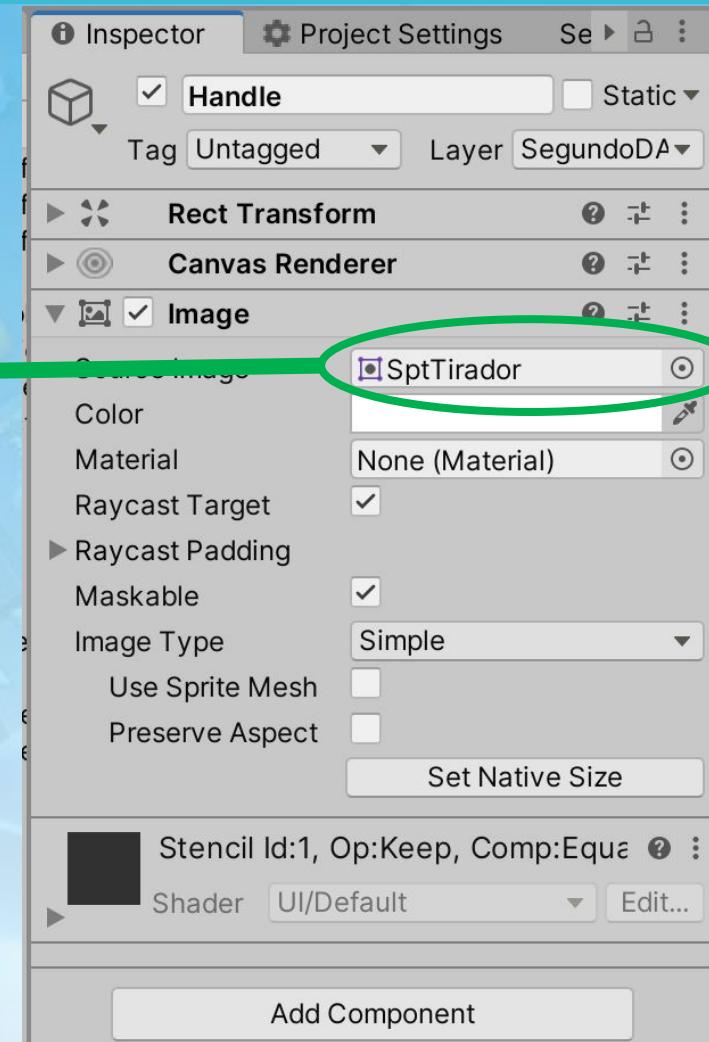
Por último nos encontramos con el GameObject hijo “Handle Slide Area” que a su vez tiene emparentado otro GameObject denominado “Handle”. Desde este último, podremos configurar los parámetros de comportamiento y de personalización del tirador del Slider.



Vamos a personalizar este tirador con un Sprite creado en Gimp.



SptTirador

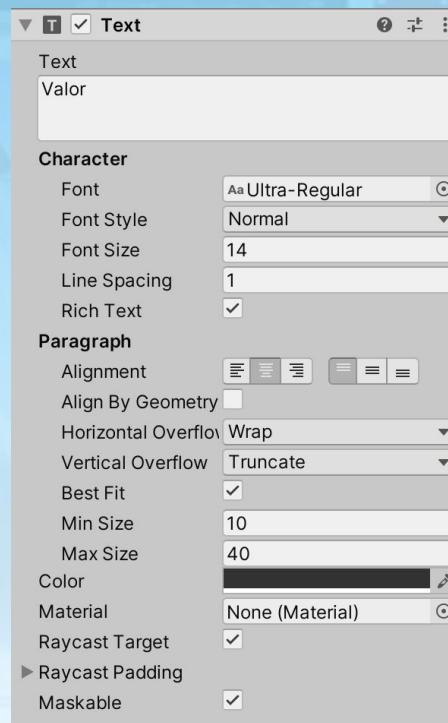
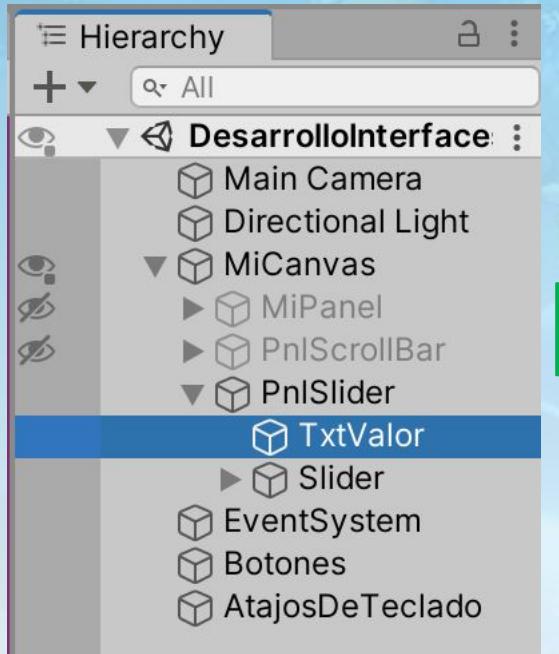


# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

Slider

Veamos ahora el código asociado al Slider. Para ello, nos vamos a servir del siguiente ejemplo: Según se va desplazando el tirador del Slider, en un GameObject de tipo “Text”, vamos a ir mostrando su valor.

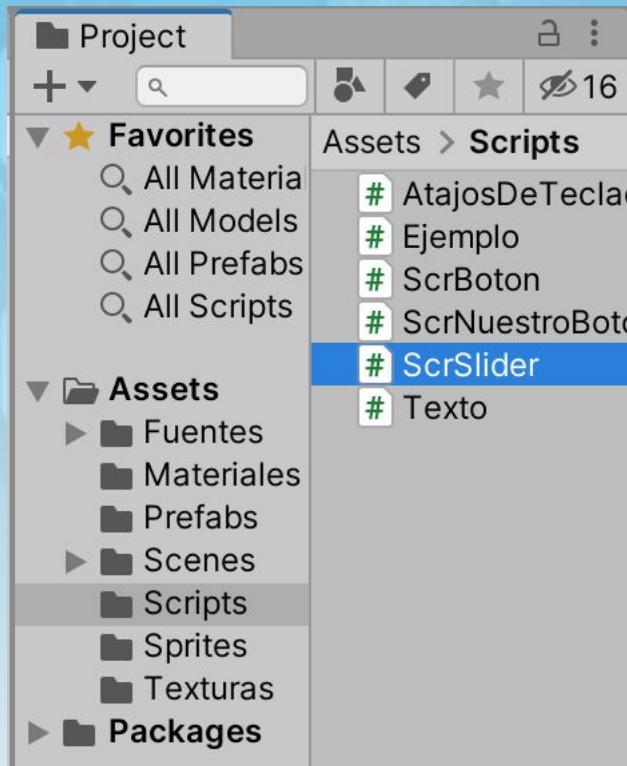
Lo primero que vamos a hacer es insertar el GameObject de tipo “Text” dentro de nuestro PnISlider.



# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

Slider

Creamos el Script “SptSlider” y lo guardamos dentro de la carpeta Script de nuestro proyecto.



```
/*
 * Date: 8 de Septiembre de 2020.
 * Class: ScrSlider.
 * Description: Desde este script, trabajamos con el
 * valor que nos devuelve el Slider
 * Author: Raquel Rojo y Mario Santos.
 * Palomeras Vallecas - Villablanca
 */
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI; //No olvidar utilizar esta librería.

public class ScrSlider : MonoBehaviour
{
    //Variable privada que vamos a utilizar
    //para apuntar al TxtValor de nuestro
    //panel PnlSlider.
    private GameObject txtValor;

    // Start is called before the first frame update
    void Start()
    {
        txtValor = GameObject.Find("TxtValor");
    }

    // Update is called once per frame
    void Update()
    {
        txtValor.GetComponent<Text>().text = (GetComponent<Slider>().value).ToString();
    }
}
```

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

Es interesante pararse en la siguiente línea de código.

```
// Update is called once per frame
void Update()
{
    txtValor.GetComponent<Text>().text = (GetComponent<Slider>().value).ToString();
}
```

La parte izquierda del igual, no tiene nada nuevo. Cogemos el Componente “Text” del GameObject “TxtValor”, a través de la variable que le apunta “txtValor”.

Una vez seleccionado este componente, vamos a introducir sobre la propiedad “text” el valor que nos devuelve el Slider.

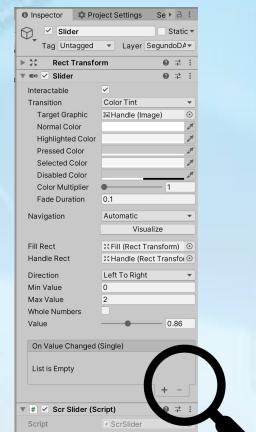
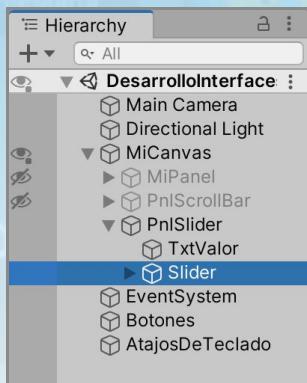
```
txtValor.GetComponent<Text>().text =
```

# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

Lo nuevo e interesante, se encuentra a la derecha del igual.

```
= (GetComponent<Slider>().value).ToString();
```

Lo primero que se observa, es que para acceder al componente Slider del GameObject “SldValor”, no estoy utilizando ningún puntero al GameObject. Esto es posible porque el Script “ScrSlider” está asociado al GameObject “SldValor”, con lo que el sistema entiende, si no le decimos nada explícitamente, que estoy trabajando sobre el GameObject en el que he asociado el script.



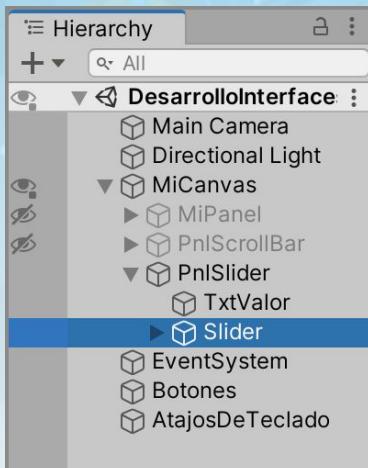
# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

Slider

```
= GetComponent<Slider>().value.ToString();
```

Lo segundo que observamos es la orden “.ToString()”. Hay que recordar, que el valor que nos devuelve el parámetro value del GameObject “SlidValor”, es de tipo “float”, pero lo intentamos representar en un GameObject de tipo Text “TxtValor”. Para poder hacer esto, hay que convertirlo a String.

Por último, es interesante echarle un vistazo a la ventana “Inspector”. Si observáis tenemos el siguiente parámetro del que no hemos hablado hasta ahora...

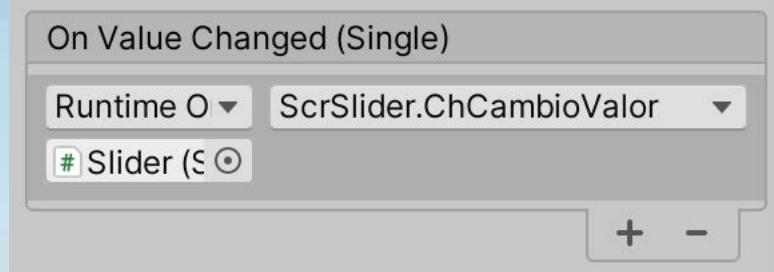


# UT3.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes básicos.

Slider

En este parámetro, podemos cargar una función pública que se va a ejecutar cada vez que se produzca el evento Changed de nuestro componente Slider. Funciona exactamente igual que con el evento OnClick visto en la U.T.2. En las imágenes se muestra el código en una función pública “ChCambioValor()” y cargada en el GameObject “Sl valor”.

```
public void ChCambioValor()
{
    txtValor.GetComponent<Text>().text = (GetComponent<Slider>().value).ToString();
}
```



[Ver el vídeo UT3\\_Slider](#)

Profesores: Raquel Rojo y Mario Santos.