



I.E.S PALOMERAS  
VALLECAS

# Desarrollo de Interfaces

2º Desarrollo de Aplicaciones Multiplataforma



I.E.S. VILLABLANCA

**UT.2 Introducción a la generación de interfaces gráficas de usuario con editores visuales:**

**Componentes contenedores básicos.**

**Parte D**

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Contenidos

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

- GameObject Text y Button. *Diapositivas 3-43*
- Prefabs. *Diapositivas 44-48*

Material Adicional a esta presentación:

- Vídeos:

- UT2-ButtonText.
- UT2-FontRichText *En este vídeo veremos como cargar fuentes de texto y trabajar con texto enriquecido.*
- UT2-ImageType
- UT2-TransitionButton

Prácticas:

- Práctica\_UT2\_IntroduccionInterfaceGrafica

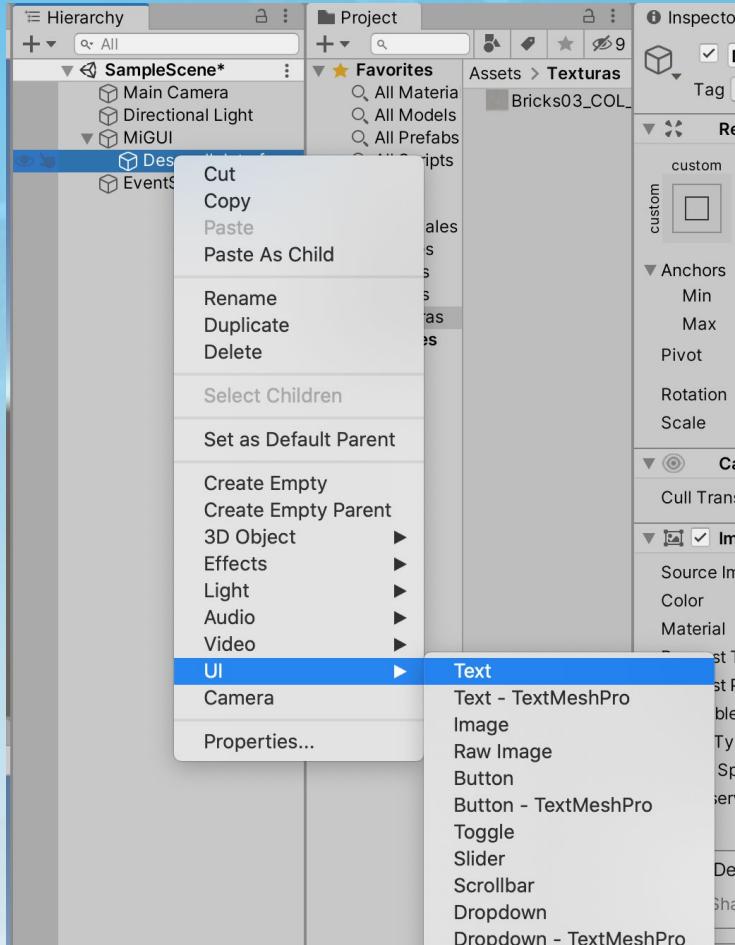
## UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

- El GameObject de tipo Text lo vamos a utilizar para mostrar información al usuario.
- Veremos cómo utilizar el campo Text de una forma estándar y sus parámetros de configuración.
- Para insertar un campo de tipo Text, podemos hacerlo como siempre, a través del menú contextual y de la barra de menús de Unity.
- Recordar que el campo Text debe estar emparentado con el Panel que lo contiene.

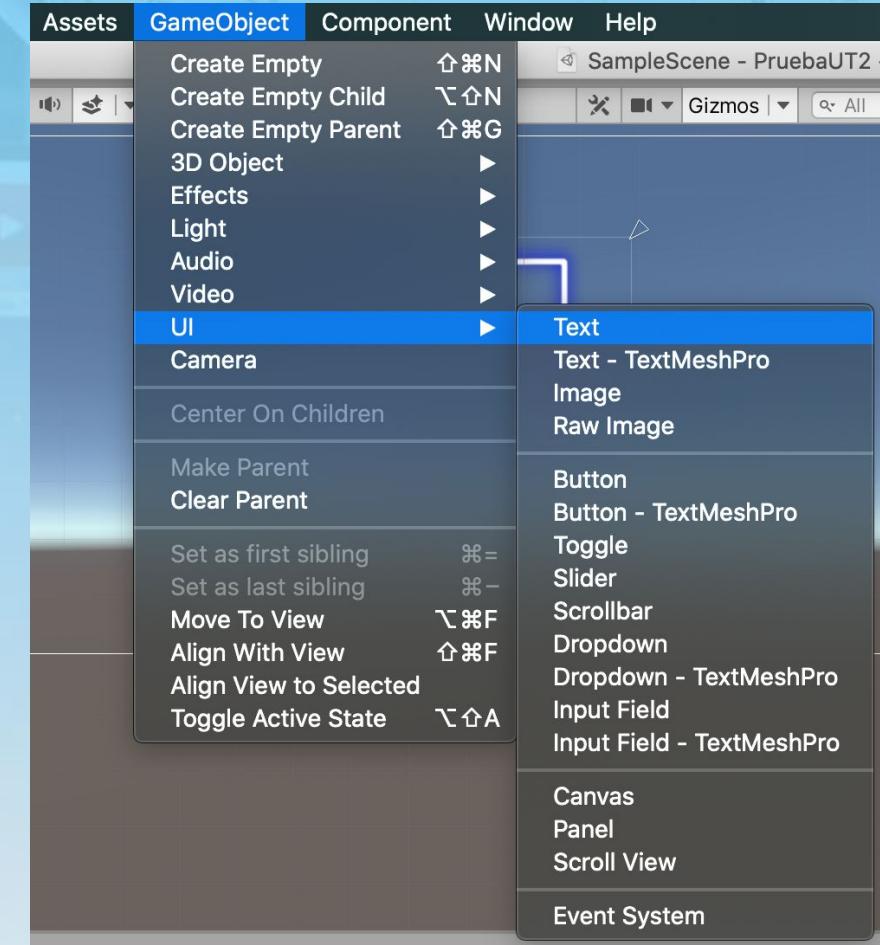


Ver el vídeo UT2\_ButtonText

## Menú Contextual



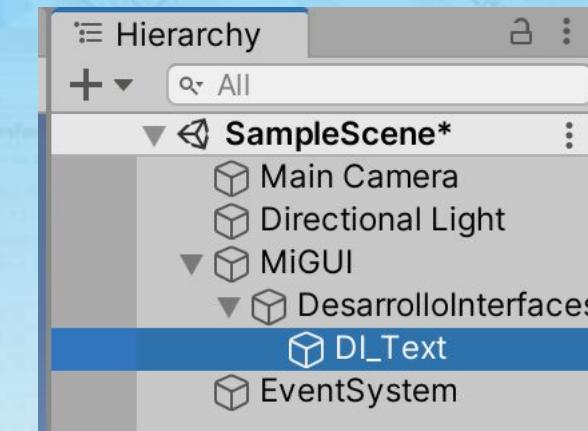
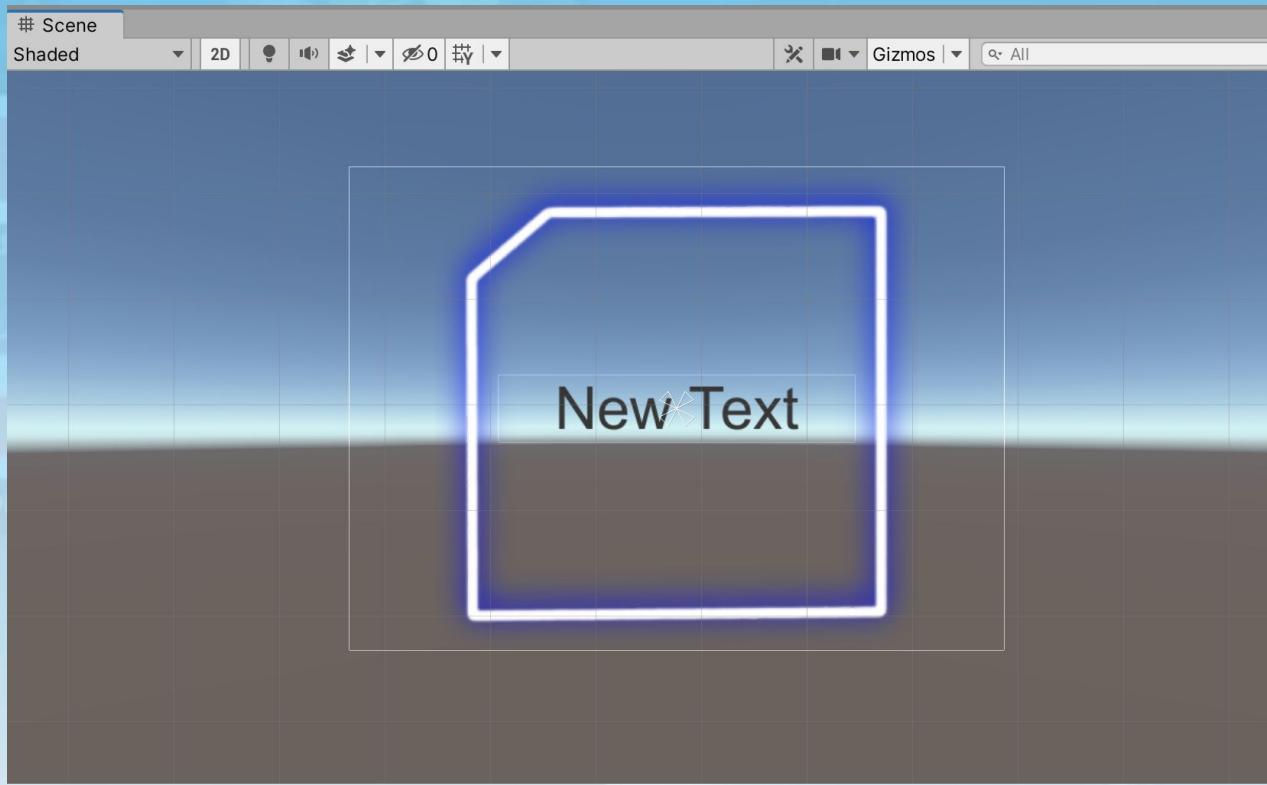
## Barra de Menús



# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

Text

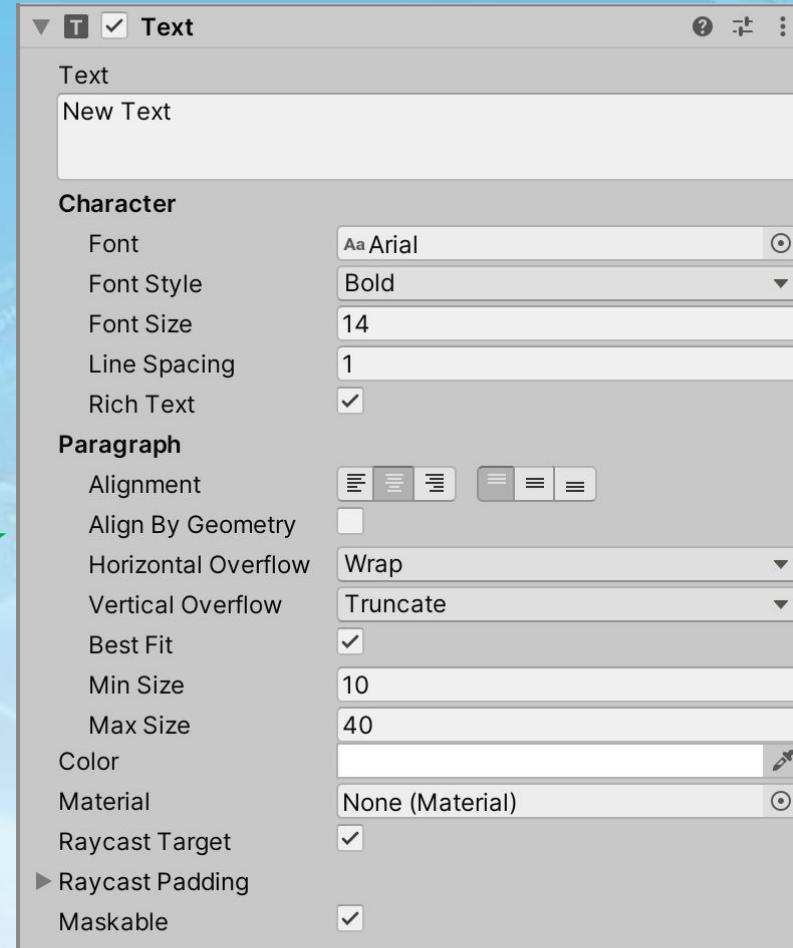
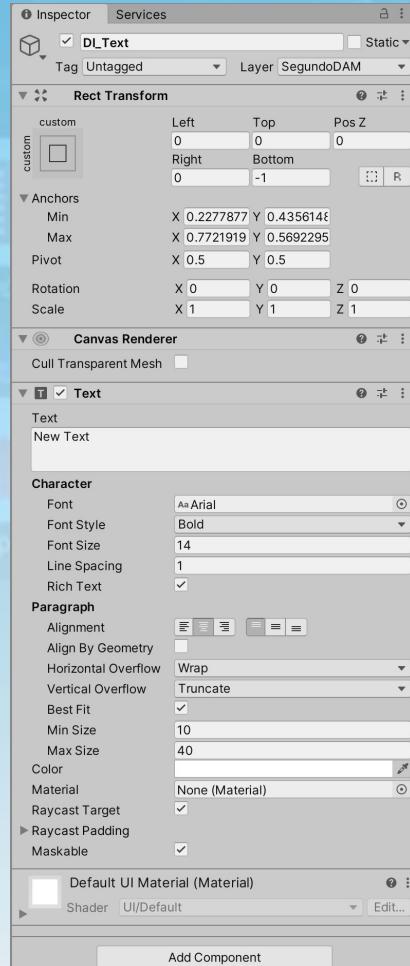
Una vez insertado un campo Text, las ventanas Hierarchy y Scene, quedarán como se muestra en las imágenes.



# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

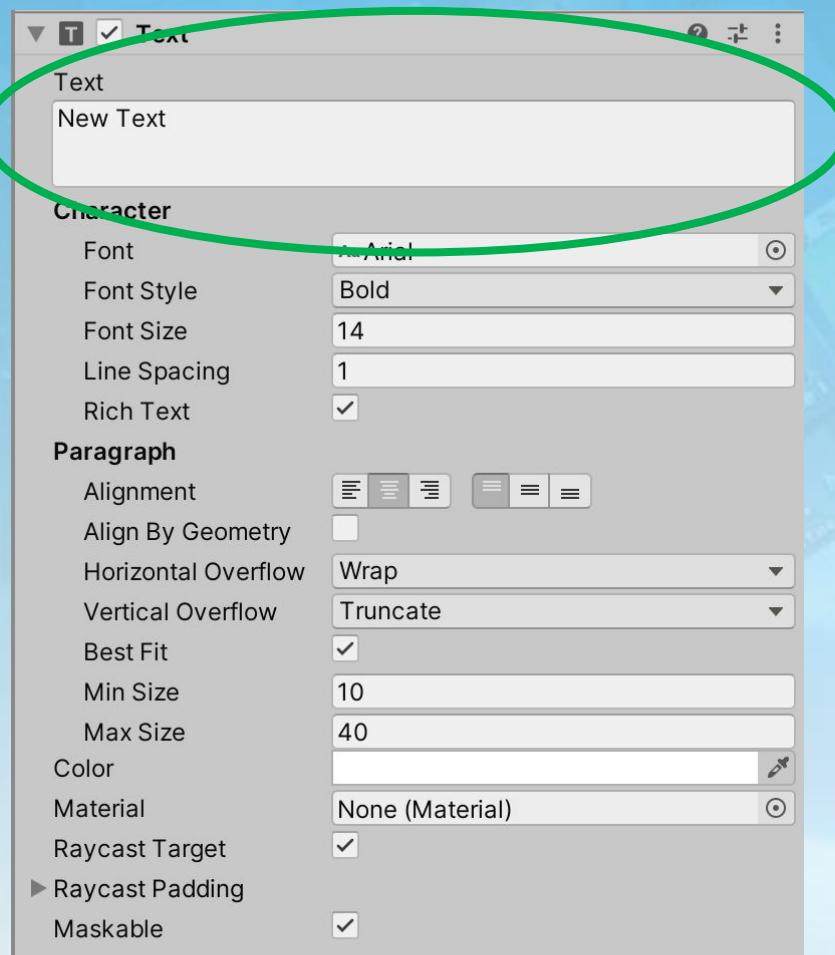
Text

Y la ventana Inspector nos muestra todas las propiedades de Text.



Nos vamos a centrar en las propiedades del componente **Text**. Las del Rect Transform las hemos visto en las presentaciones anteriores.

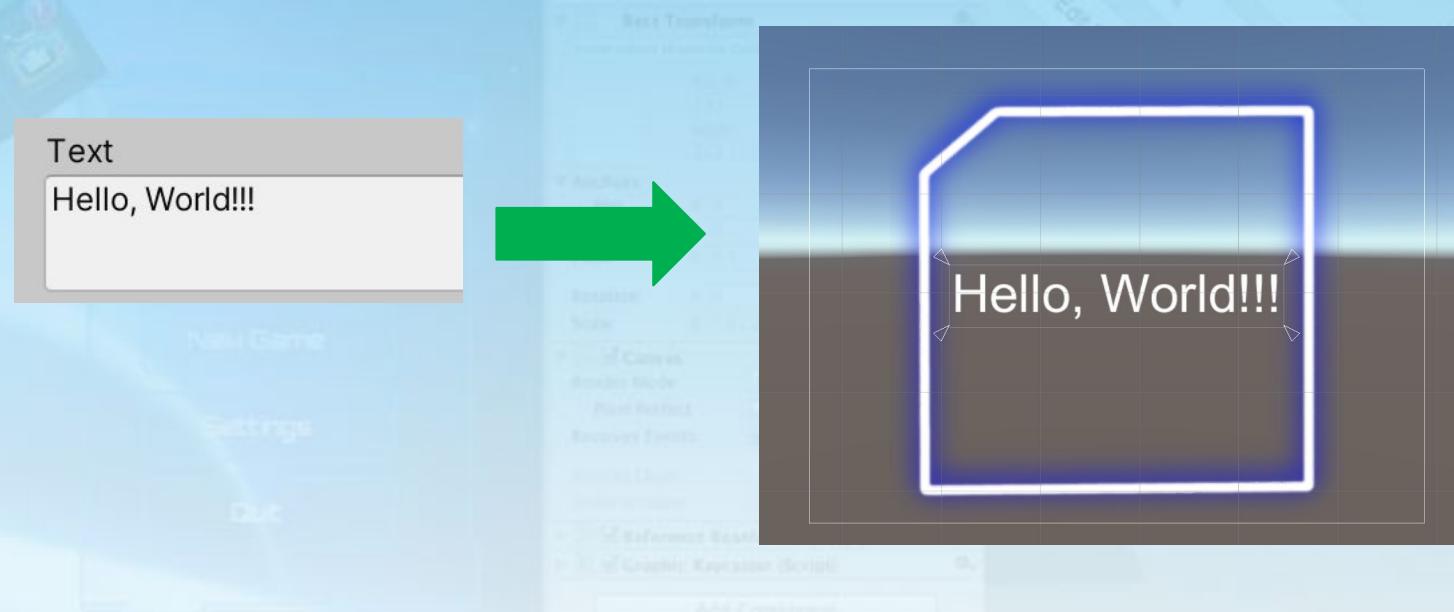
# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.



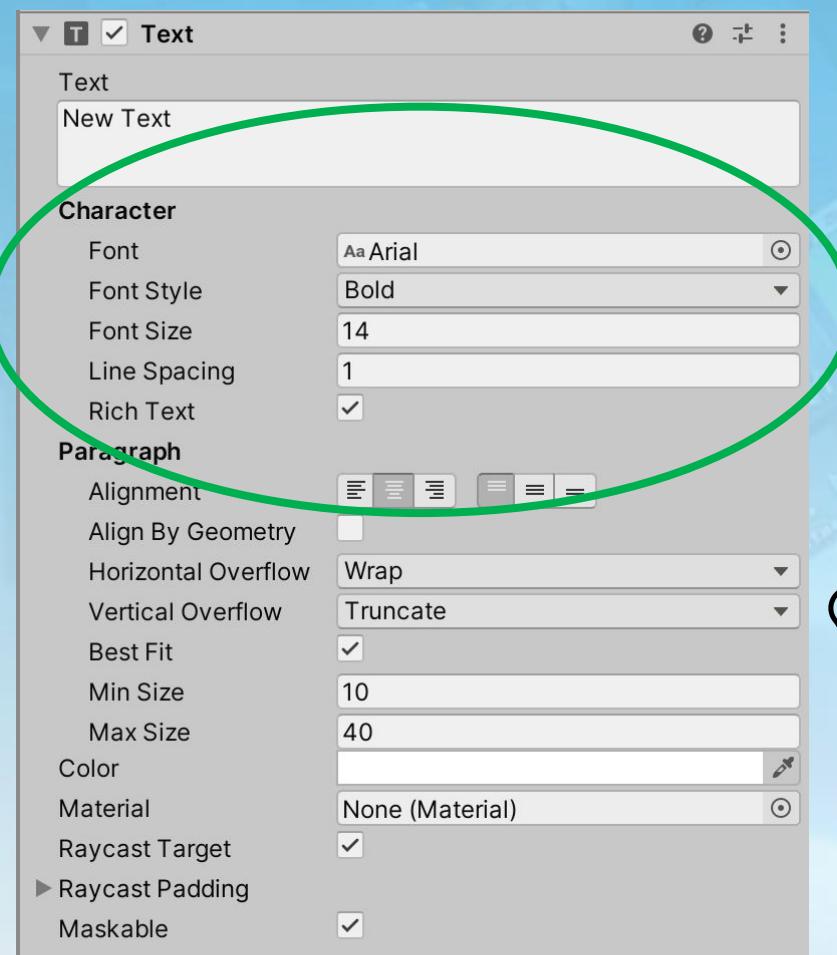
- **Text:**

En este parámetro vamos a introducir el texto que luego podrá ver el usuario por la GUI. El valor que se introduce en este parámetro, se puede introducir desde la ventana inspector o desde un script, como veremos más adelante.

Si borramos “New Text” e introducimos “Hello, World!!!” obtenemos en nuestra ventana Scene...



# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.



## ● Character:

Esta es una agrupación de parámetros que actúan sobre los caracteres:

- **Font:** La fuente utilizada para mostrar el texto.
- **Font Style:** Normal o Regular, Bold o negrita, Italic o cursiva y Bold and Italic o negrita y cursiva.
- **Font Size:** Tamaño de la fuente, por defecto 14.
- **Line Spacing:** Espacio de interlineado.
- **Rich Text:** Texto enriquecido.

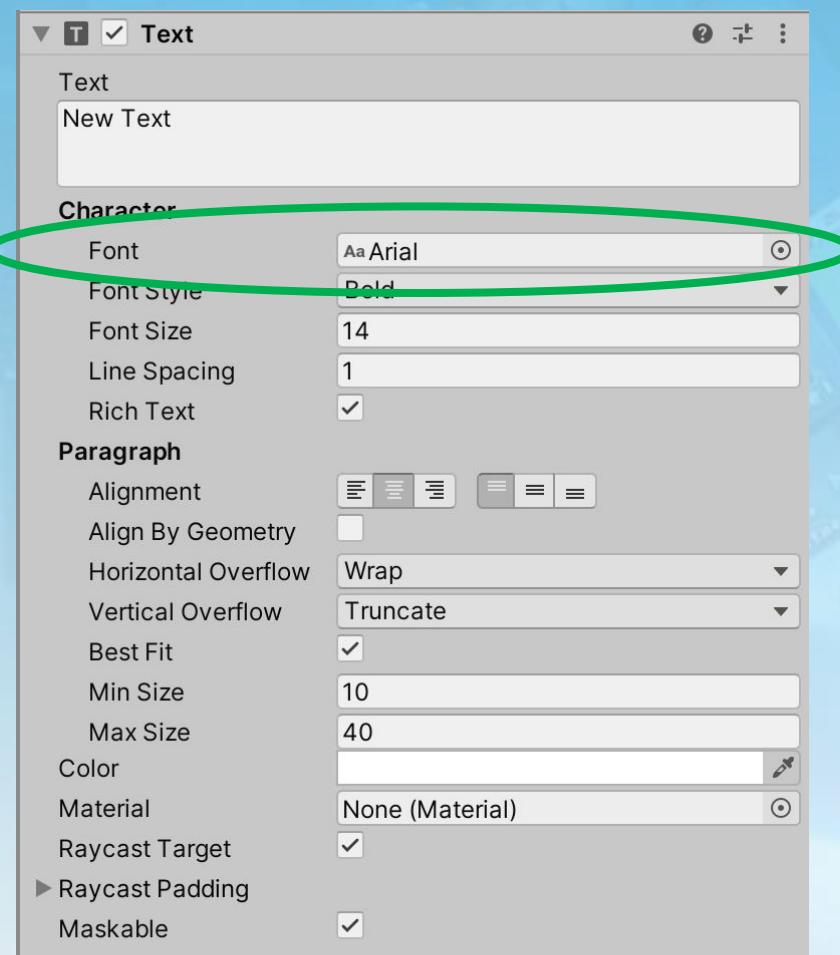


Estos parámetros no tienen mayor dificultad, pero vamos a pararnos un momento en los parámetros:

- **Font**
- **Rich Text**

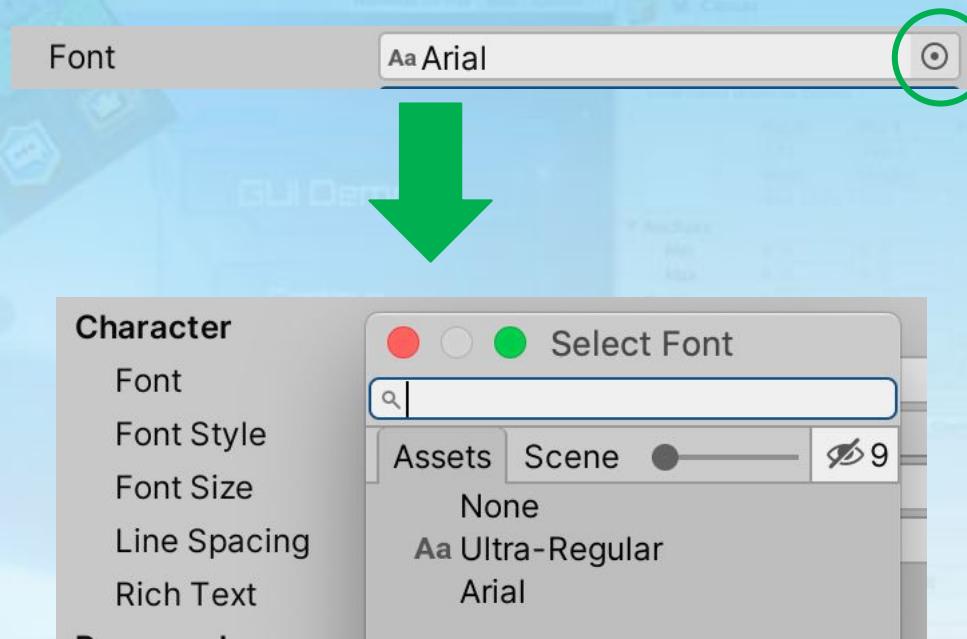
# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

## Componentes contenedores básicos.



- **Font:**

El campo Font, en el que se muestra la fuente o podemos seleccionar entre las distintas fuentes que tengamos cargadas en nuestro proyecto. Si pulsamos sobre el botón que se encuentra a la derecha de la lista desplegable, nos las mostrará.

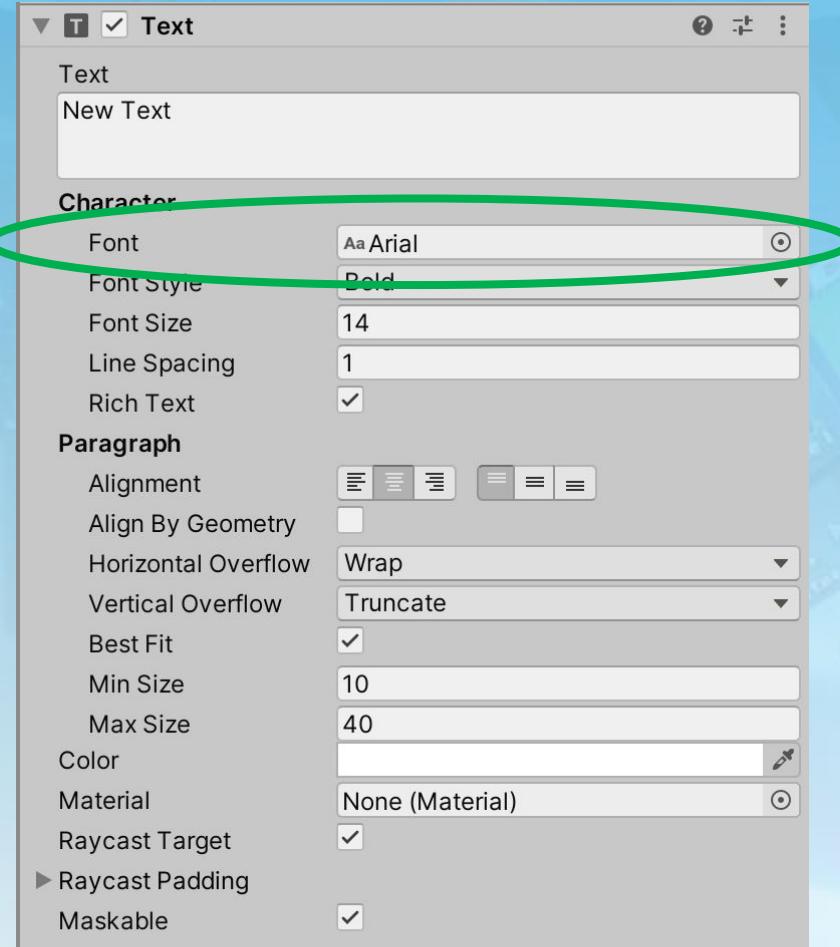


En este caso, vemos que cuando pulsamos sobre el botón, podemos elegir entre las siguientes fuentes:

- None
- Ultra-Regular
- Arial

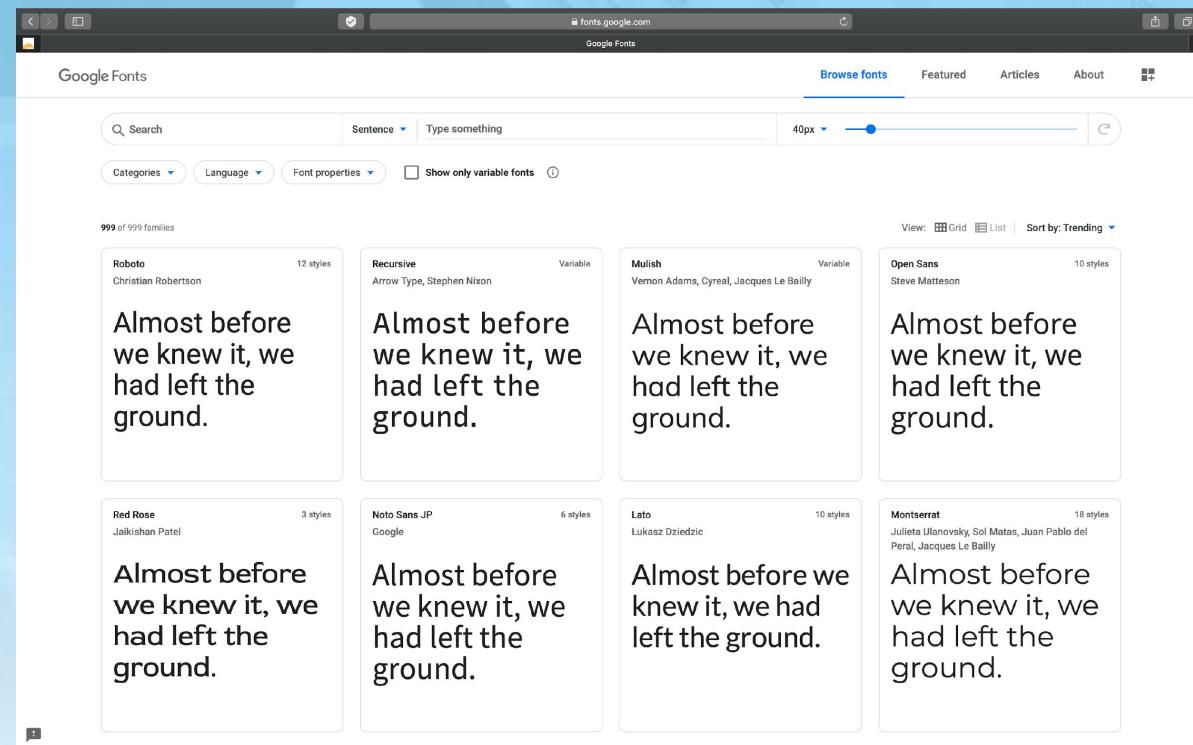
# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

Text



- Font:

Para cargar Fuentes nuevas, las podemos descargar desde varias páginas de internet. Un ejemplo es: <https://fonts.google.com>



Ver vídeo UT2\_FontRichText

Profesores: Raquel Rojo y Mario Santos.

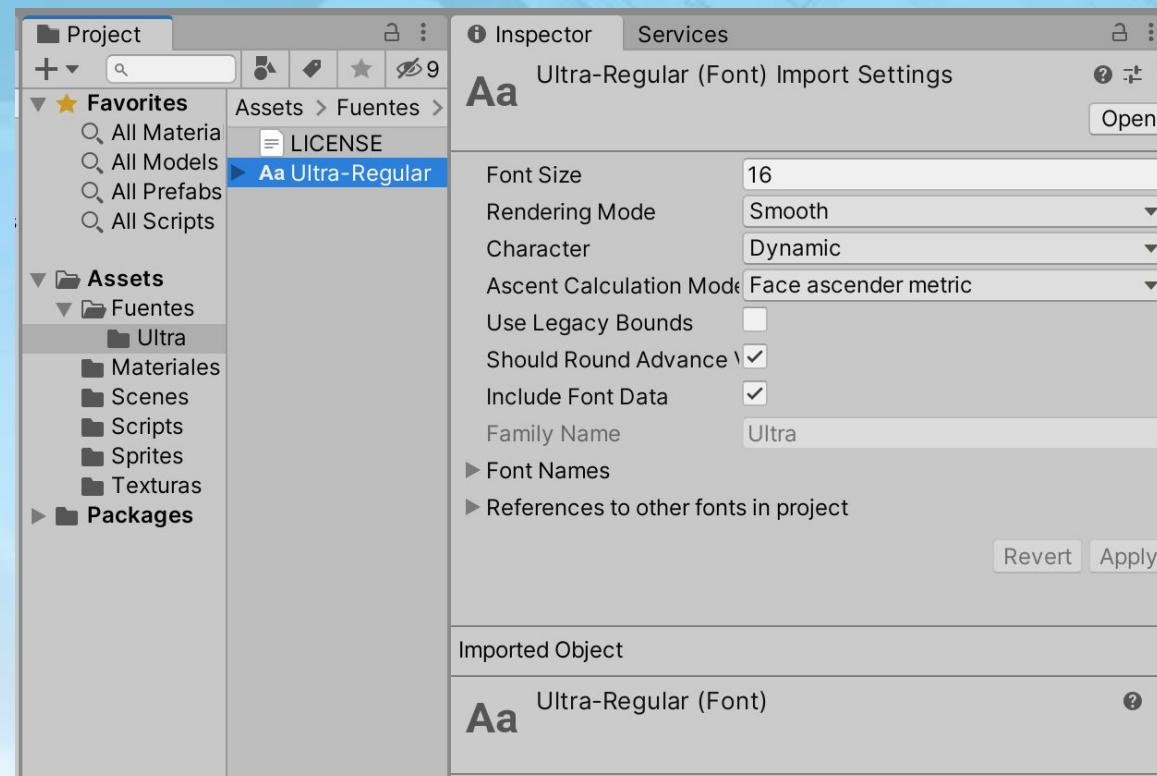
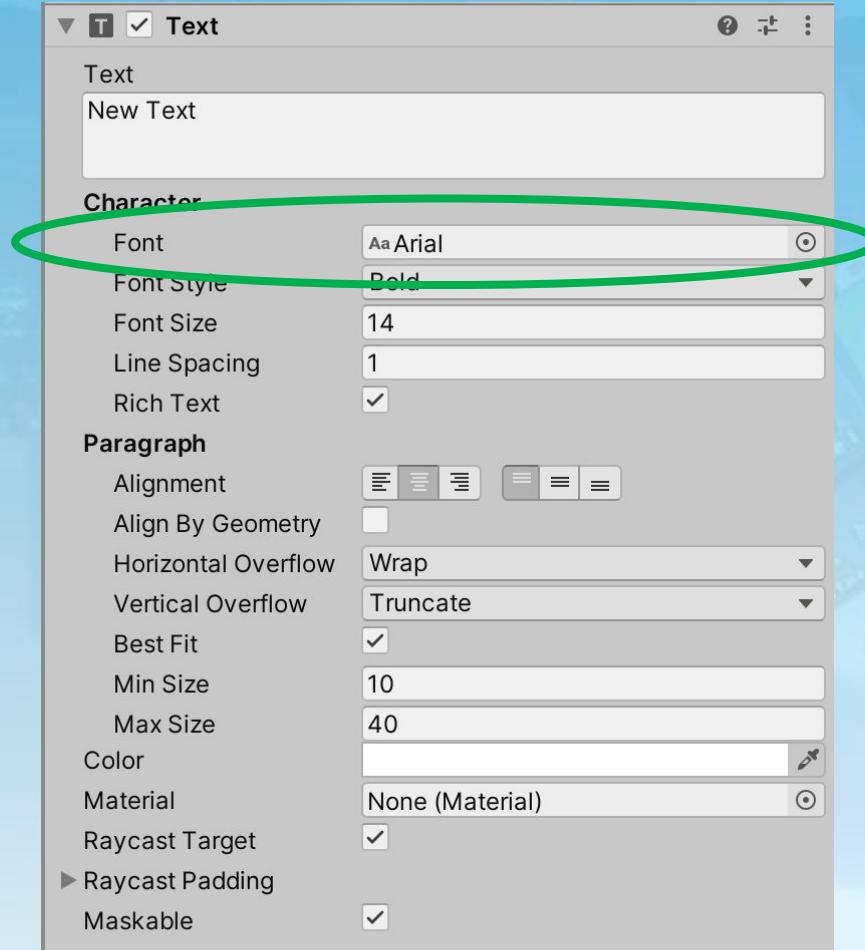
# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

## Componentes contenedores básicos.

Text

- Font:

Una vez descargada, creamos una carpeta llamada **Fuentes**, en nuestra ventana de Project, y la guardamos. En este caso, la fuente que se ha descargado se denomina Ultra.



[Ver vídeo UT2\\_FontRichText](#)

Profesores: Raquel Rojo y Mario Santos.

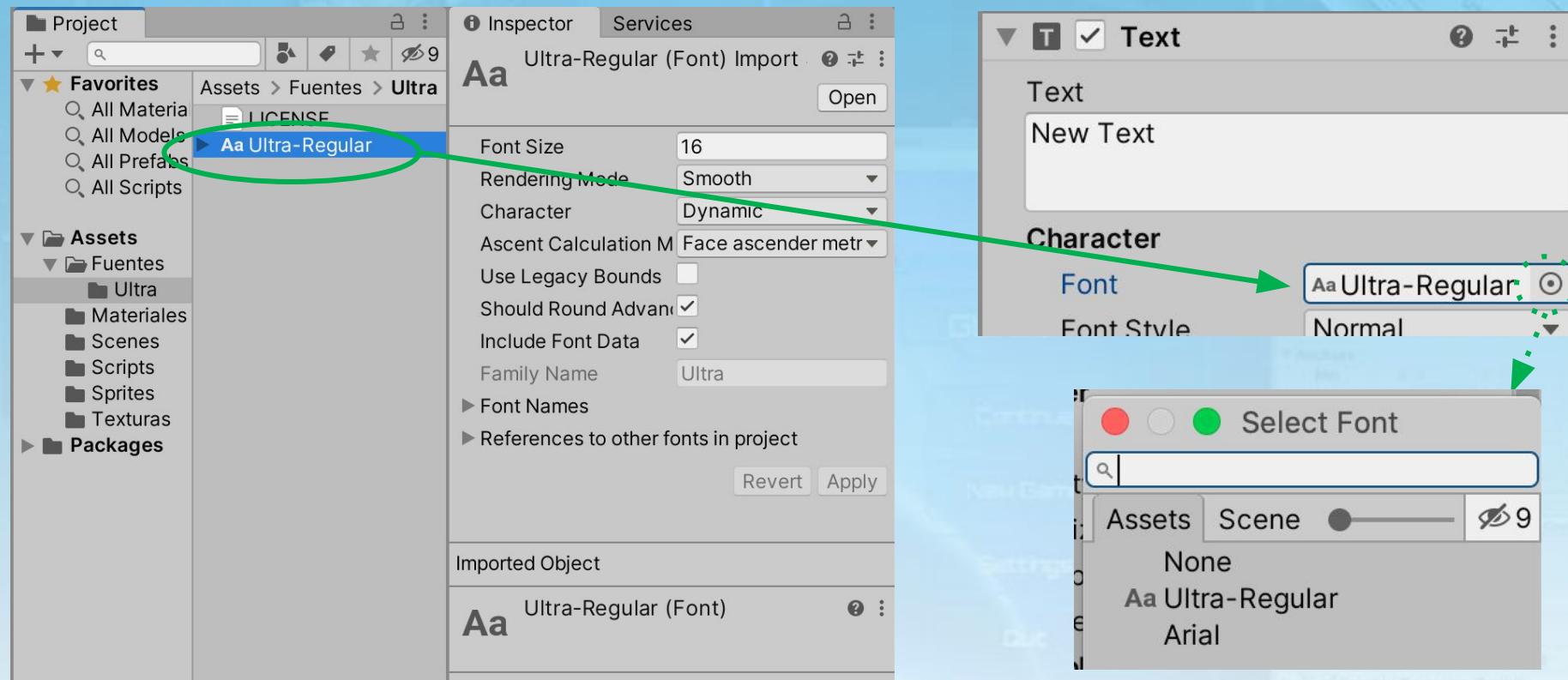
# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

## Componentes contenedores básicos.

Text

- Font:

Por último, arrastramos Aa Ultra-Regular dentro del campo Font, o pulsamos sobre el botón de la derecha.



Recordar que para ver los componentes de un GameObject en la ventana **Inspector**, ese GameObject debe estar seleccionado en la ventana **Hierarchy**. En este caso, seleccionamos el GameObject Text.

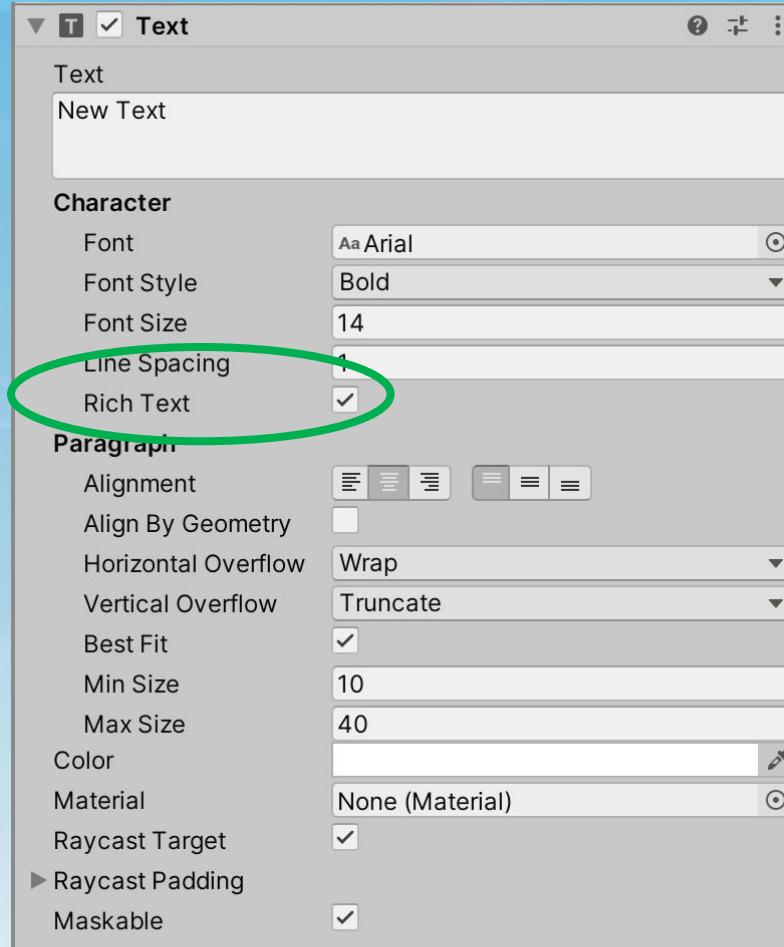


Ver vídeo UT2\_FontRichText

Profesores: Raquel Rojo y Mario Santos.

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

Text



- Rich Text:

Este parámetro nos permite trabajar con texto enriquecido, lo que da mucha potencia al GameObject de tipo Text. Nos permite trabajar con etiquetas, a modo HTML, dando formato al texto. En la documentación de Unity, podéis encontrar las etiquetas que podéis utilizar.

<https://docs.unity3d.com/2020.2/Documentation/Manual/StyledText.html>

Estas etiquetas se introducen directamente en el parámetro Text, como ejemplo ver las imágenes de la siguiente diapositiva.

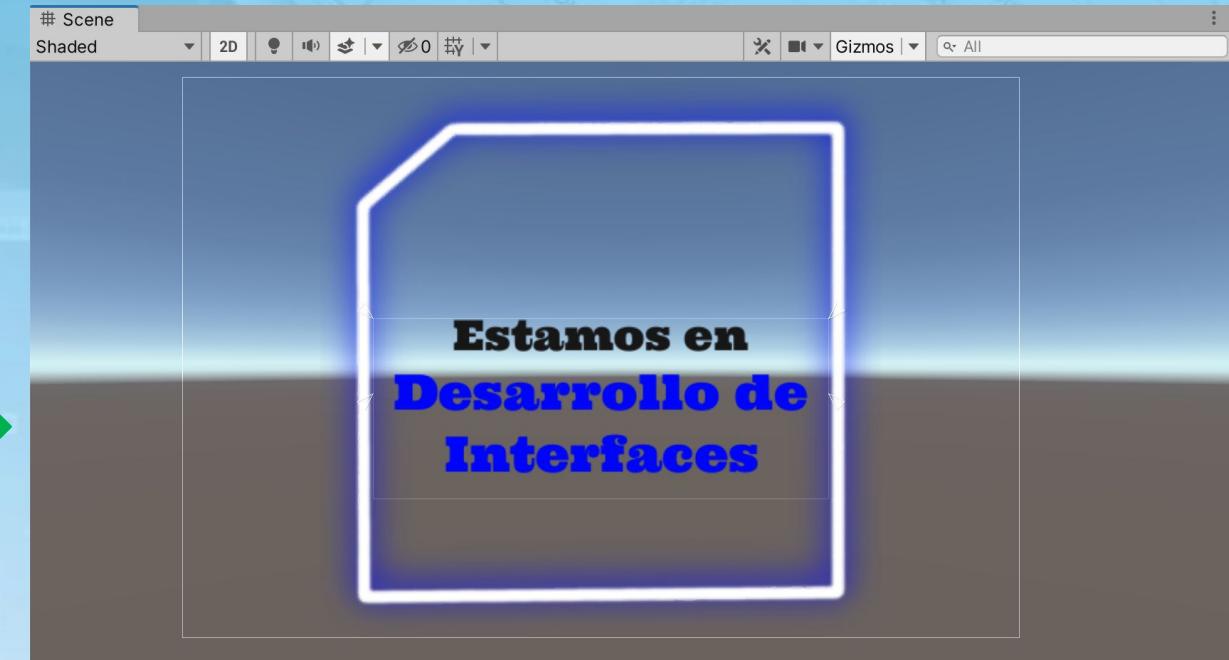
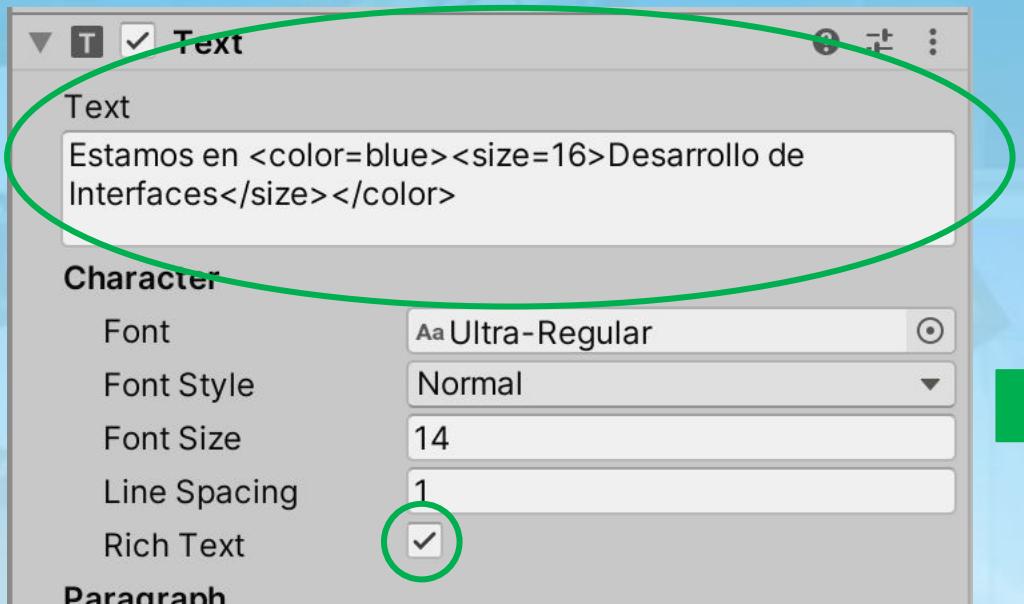


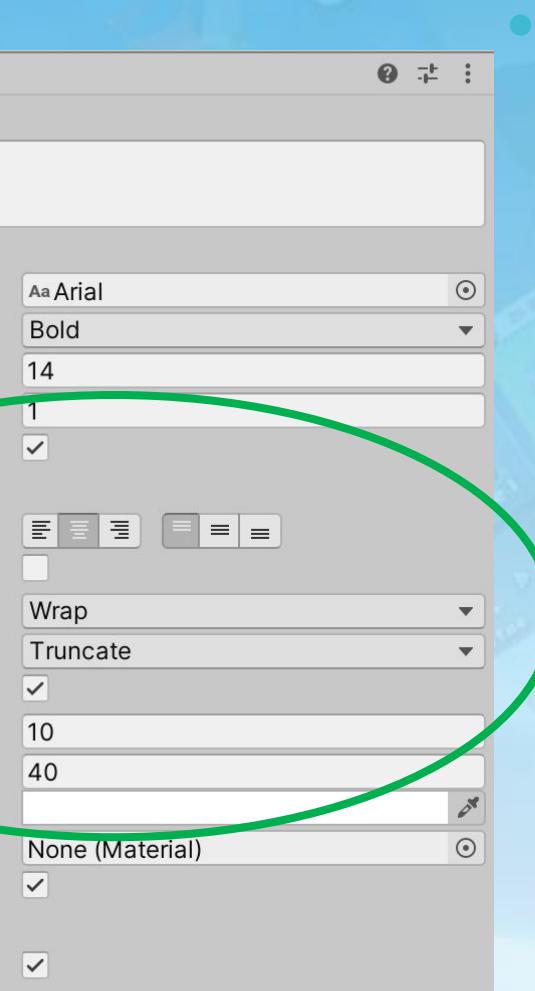
Ver vídeo UT2\_FontRichText

Profesores: Raquel Rojo y Mario Santos.

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

- Rich Text:





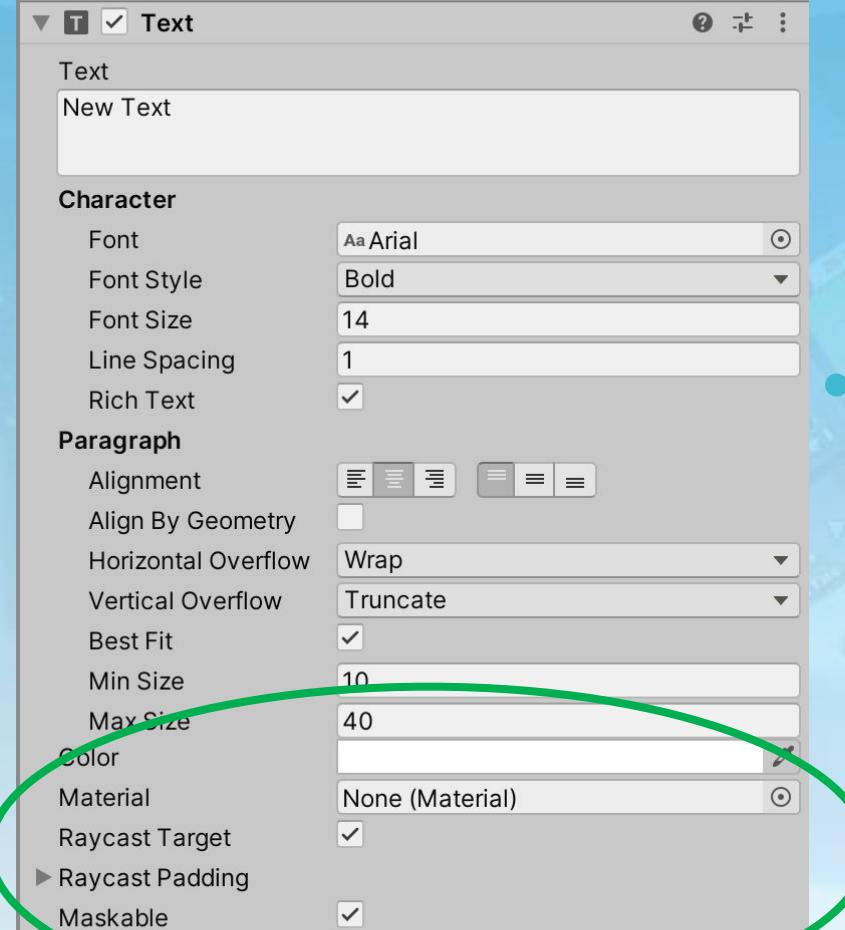
- **Paragraph:**

El siguiente grupo de parámetros que nos encontramos son los que afectan a los párrafos.

- **Alignment:** Alinea el texto.
- **Align By Geometry:** Alinea el texto teniendo en cuenta el parámetro Alignment y el tamaño del campo Text.
- **Horizontal/Vertical Overflow:** Indica si queremos que el texto se corte automáticamente o rebose los límites de la geometría.
- **Best Fit:** Unity calcula el tamaño más optimo para ajustar el párrafo a la geometría.
- **Min Size Vs Max Size:** Podemos dar una cota inferior y superior que nunca se rebasará a la hora de agrandar o minimizar el texto automáticamente.

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

Text



- **Color y Material:**

Estos dos parámetros los hemos visto ya en la parte C de la UT2, cuando hemos estado trabajando con Paneles.

La forma de funcionar es la misma, el parámetro Color, nos permite modificar el color de la fuente y el parámetro Material, nos permite crear un Material para ponérselo a la fuente.

- **Raycast Target, RayCastPadding y Maskable:**

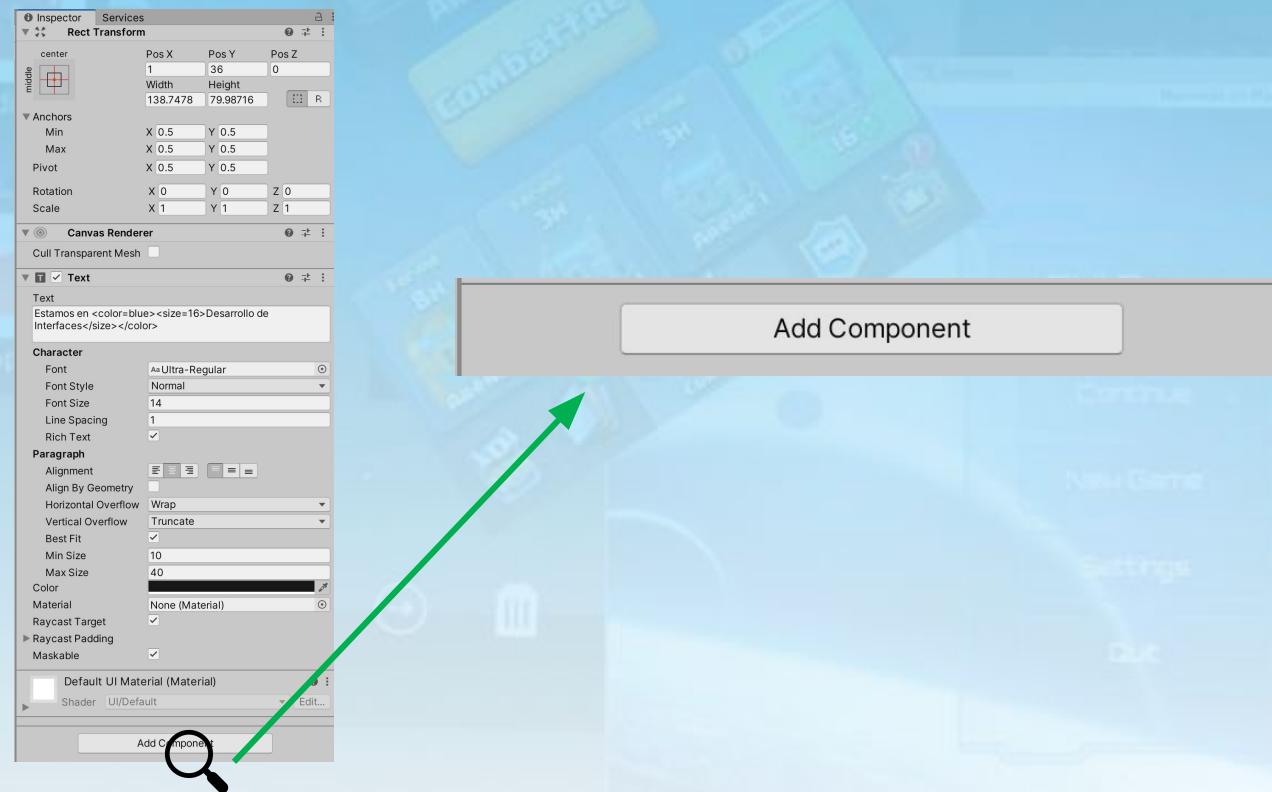
Estos parámetros serán estudiados en la U.T. en la que veamos las VR.

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

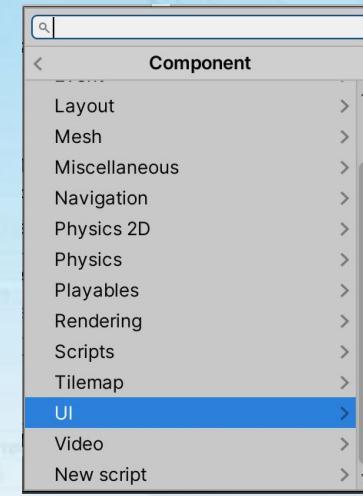
## Componentes contenedores básicos.

Al componente o GameObject Text, se le puede añadir, desde la ventana Inspector, otros componentes para aportarle efectos al texto como por ejemplo una línea que remarque el contorno del texto, una sombra o la combinación de ambas.

Para añadir estos efectos seguiremos los siguientes pasos:



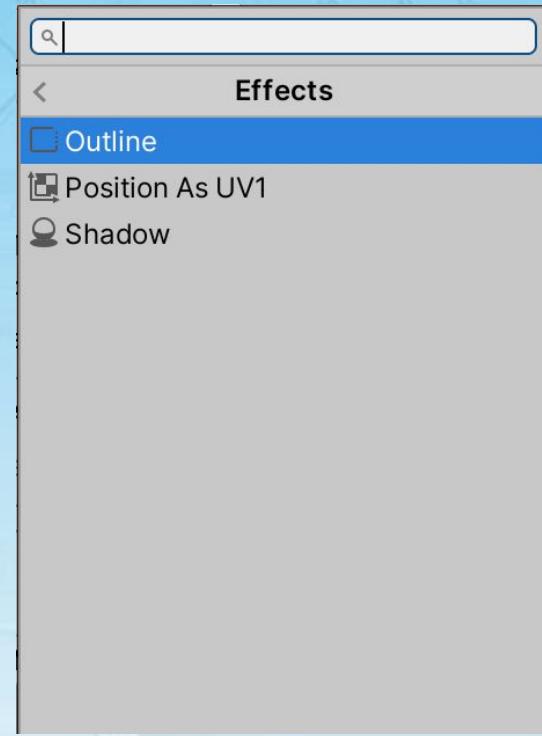
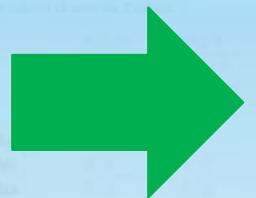
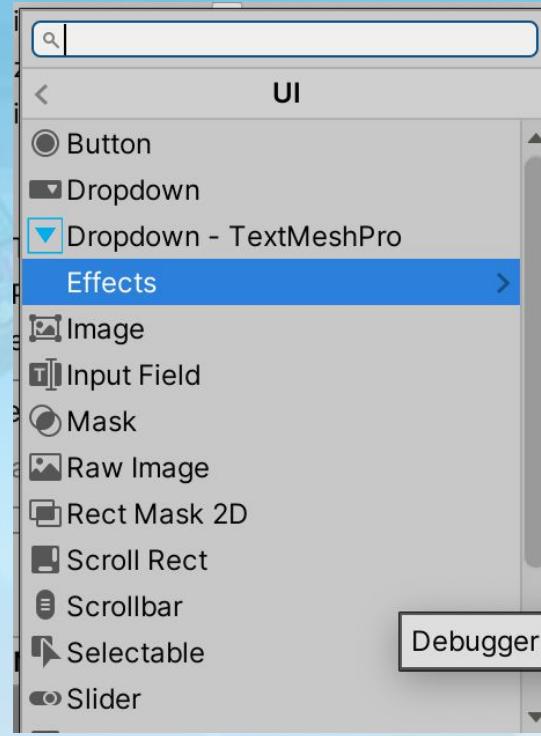
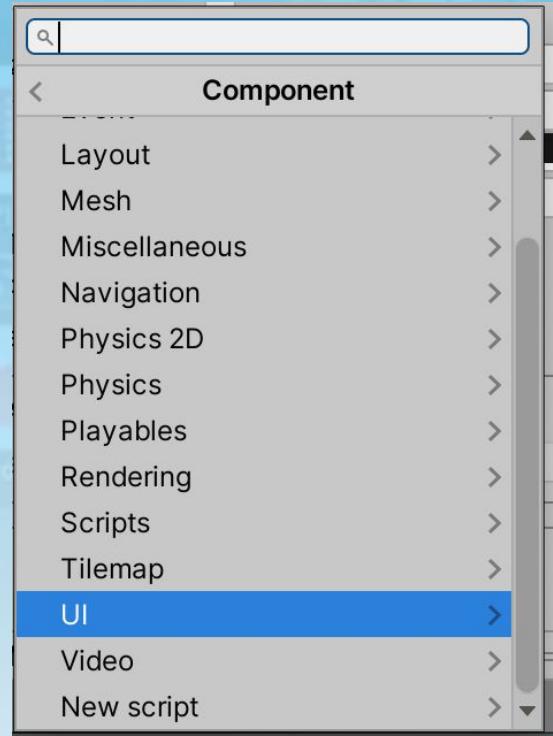
Pulsamos sobre el botón Add Component y nos mostrará las siguientes opciones:



# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

Text

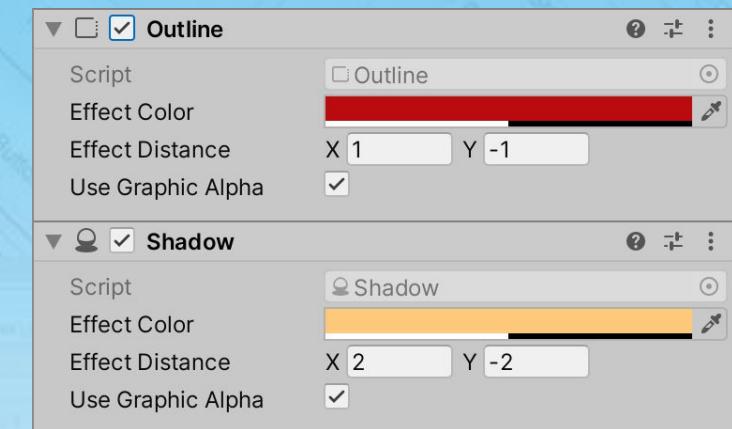
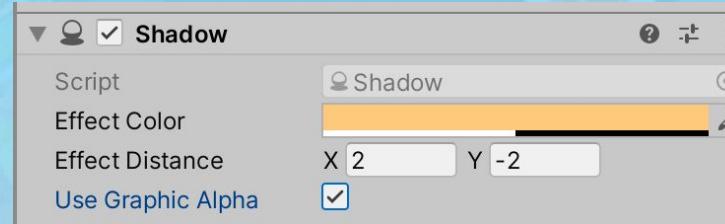
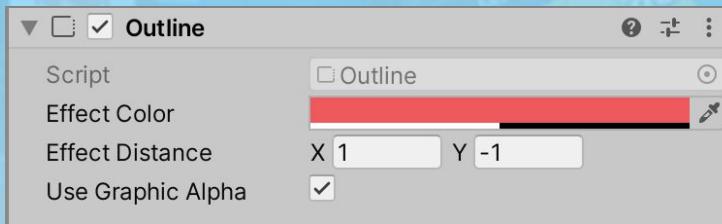
Una vez nos muestre todos los componentes entraremos en UI y encontraremos distintos componentes pertenecientes a la UI de Unity. Debemos entrar en Effects, dentro, encontraremos lo que buscamos.



# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

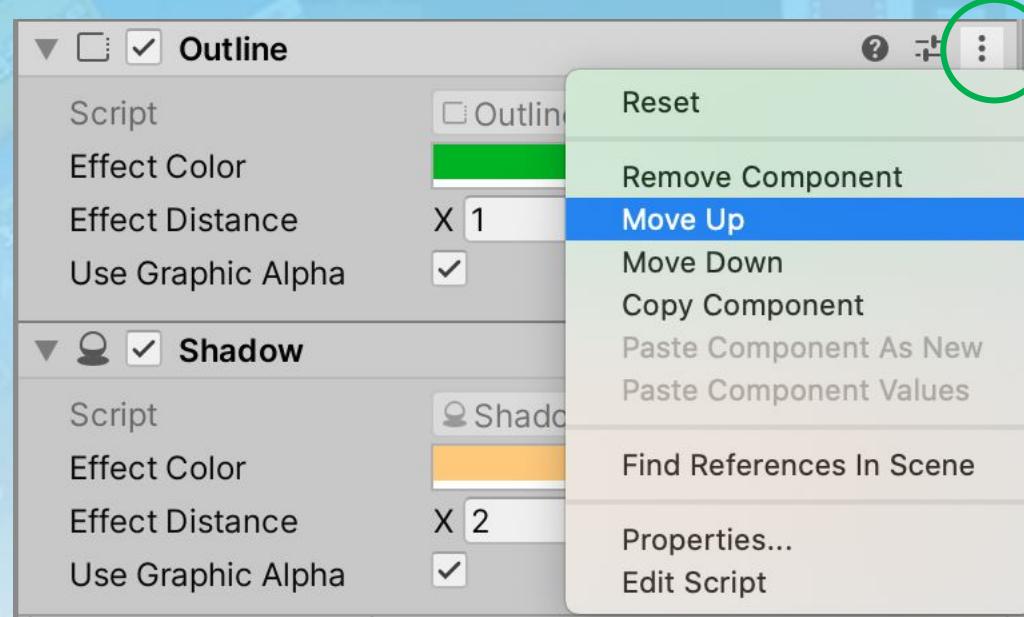
Una vez añadido el componente, por ejemplo el OutLine, se verá en la ventana Inspector y podremos modificar sus parámetros.

El resultado se verá en la ventana Scene y Game.



## UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

Es importante reseñar que el orden en el que añadimos los componentes es importante (igual que las capas en programas como el Gimp o el Photoshop). Para cambiar el orden de los componentes se pulsa sobre los tres puntos que hay en la esquina superior derecha y se selecciona la opción deseada.



# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Button

- Este componente de la GUI es de los más utilizados.
- Siempre deben aparecer dentro de un **Canvas** y pueden estar o no, dentro de un **Panel**.
- Podremos ejecutar funciones públicas que hayamos creado en los scripts.
- Podemos trabajar con los botones estándar o personalizados.
- El componente o GameObject Button es el que se muestra en la imagen.

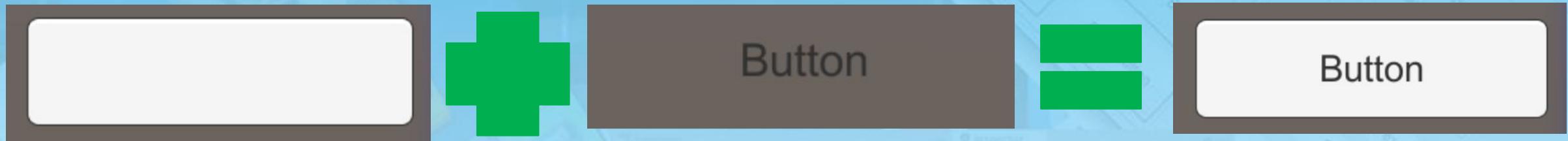


Este es un Button estándar. Veámoslo un poco más en profundidad.

## Button

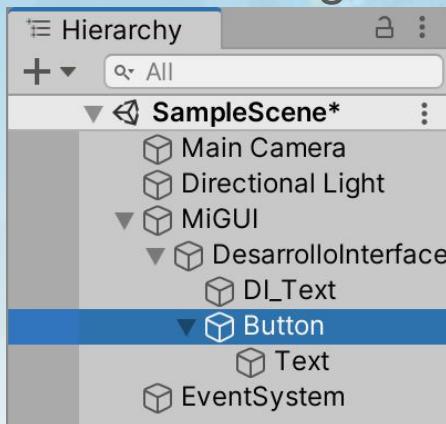
UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:  
Componentes contenedores básicos.

A simple vista y sin preocuparnos de otros aspectos, podemos observar que el botón se compone de una imagen de un rectángulo gris y un texto encima.



Esto nos está indicando que por un lado vamos a trabajar con un componente **Image** y por otro con un componente **Text** y que la **unión** de los dos, crea el **GameObject Button**.

Si observamos la ventana Hierarchy, Unity ha creado formalmente lo que hemos representado gráficamente. Ha cargado el GameObject Button y le ha emparentado otro GameObject Text.

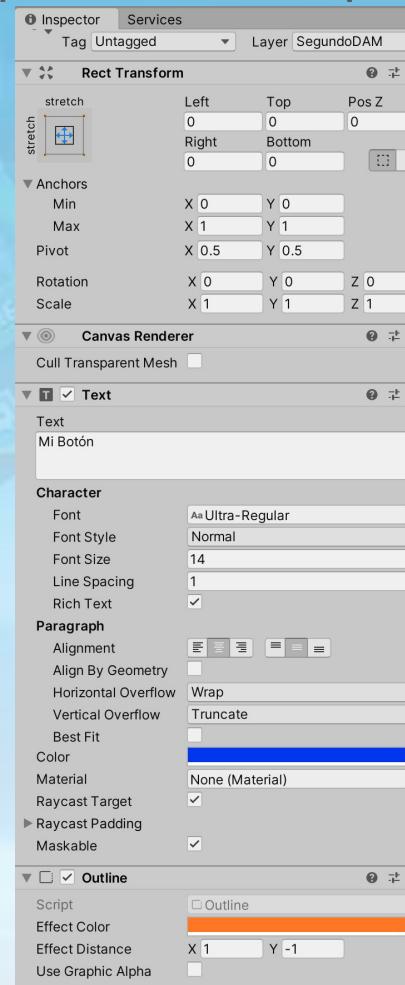
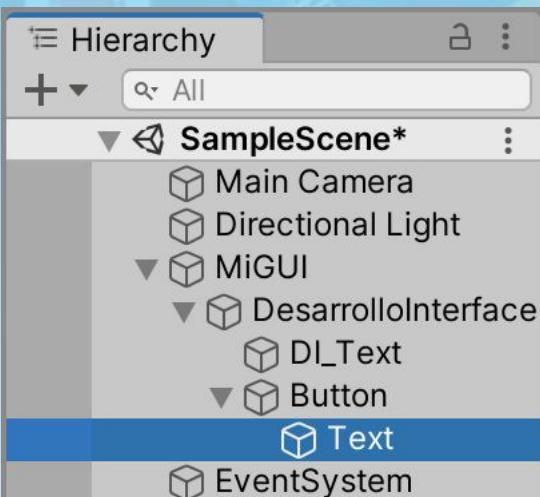


El componente **Image** se verá en una UT posterior.

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Button

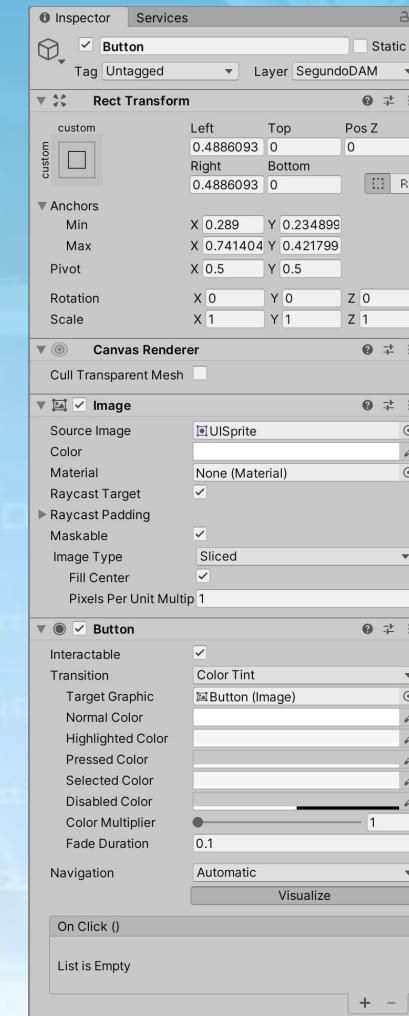
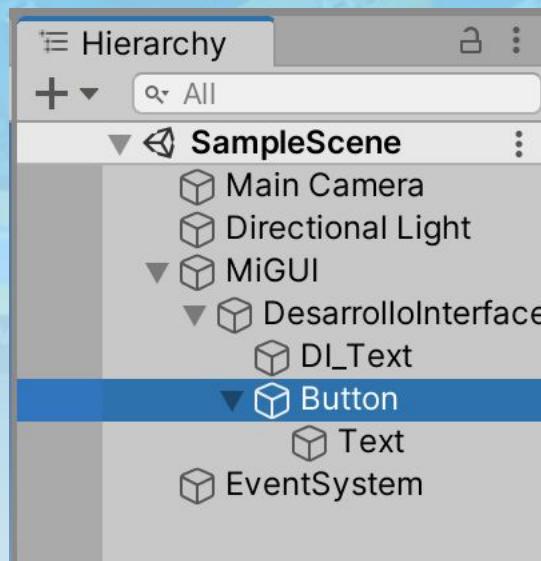
Al tener emparentado un GameObject de tipo Text, todo lo que hemos visto en estas transparencias para los campos de tipo Text lo podemos aplicar al botón para personalizarlo. Por ejemplo:



# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Button

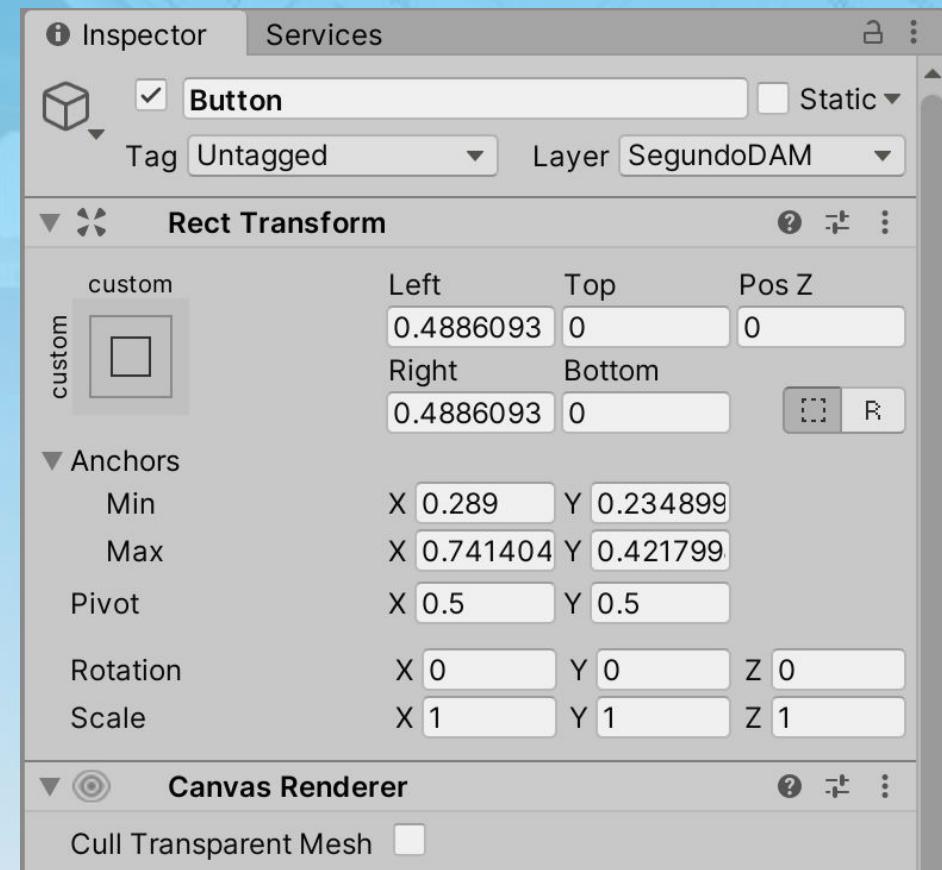
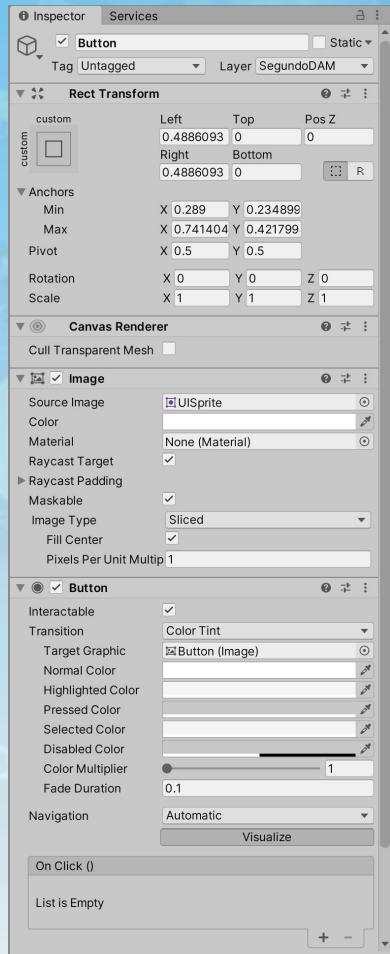
Hemos visto el GameObject hijo Text, veamos ahora, las propiedades del GameObject padre Button en la ventana Inspector:



# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

Button

Los primeros componentes de este GameObject tipo Button, Rect Transform y Canvas Rederer, ya los hemos trabajado con anterioridad en esta U.T.

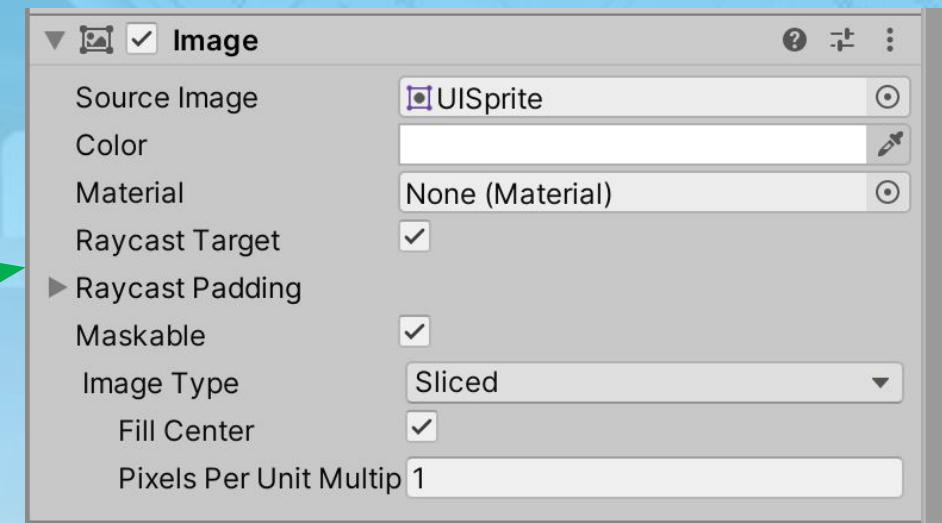
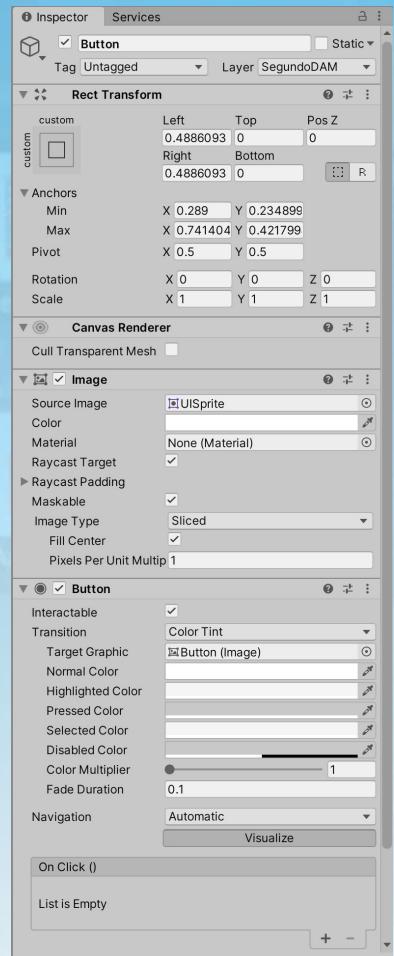


# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

## Componentes contenedores básicos.

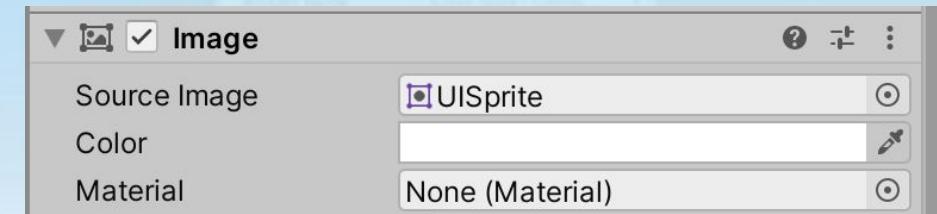
Button

Empecemos por fijarnos en el componente Image.



Los tres primeros parámetros los hemos visto con los paneles:

- Source Image
- Color
- Material

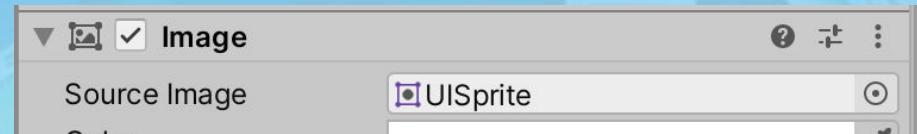


# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

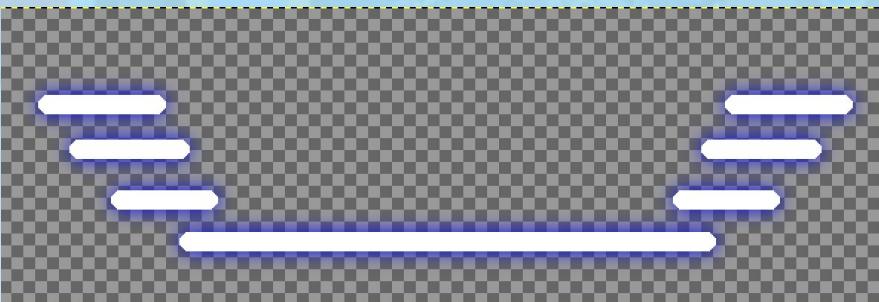
Button

## Source Image.

En este parámetro vamos a cargar el Sprite que va a determinar la forma de nuestro Button. Desde aquí, al igual que con los paneles, podemos personalizar nuestros botones.



Vamos a personalizar nuestro botón con un Sprite propio semejante al que utilizamos para el panel y que se muestra en la imagen.



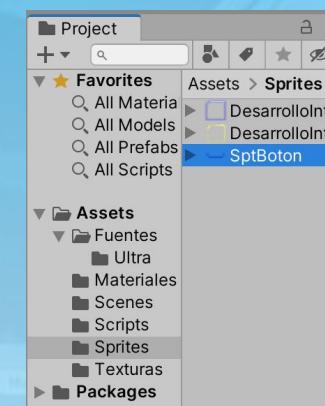
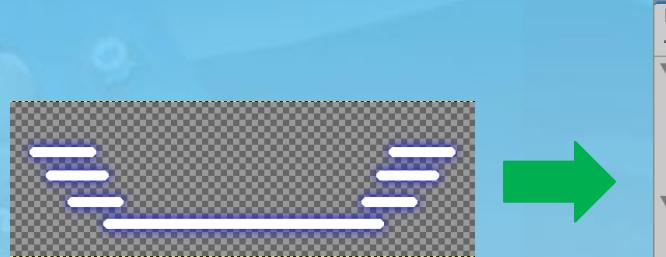
Como vimos con los paneles, se ha creado un Sprite, denominado **SptBoton**, y lo hemos guardado dentro de la carpeta **Sprites**, en la ventana Project.



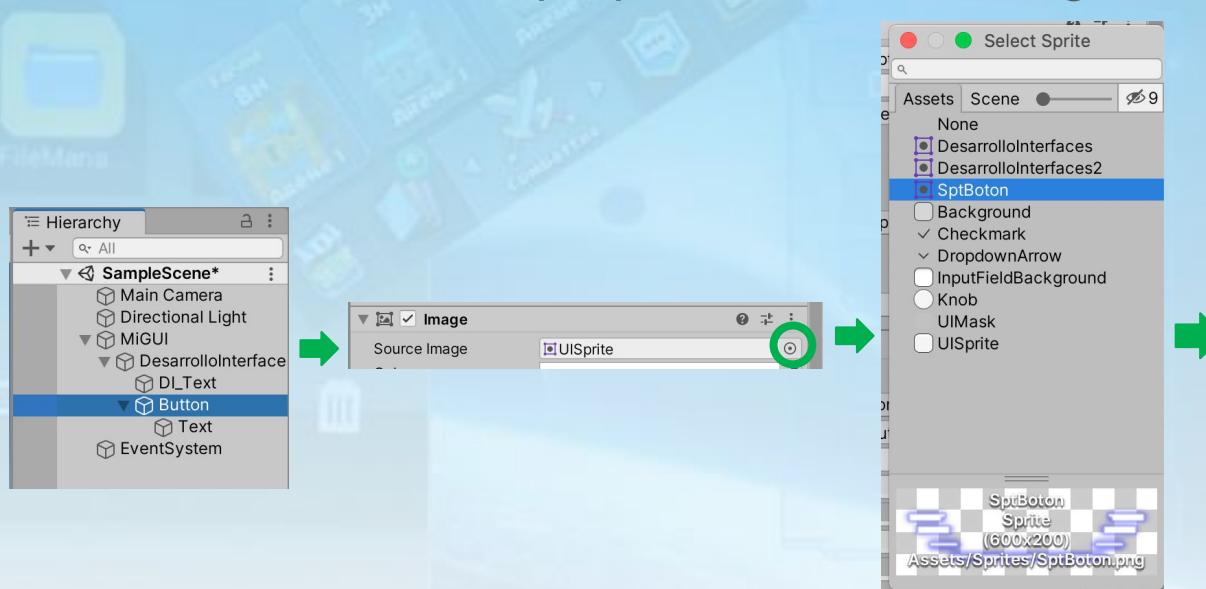
Ver el vídeo **UT2-CreacionSpritesColorMateriales**

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

**Source Image.**



Y ahora, lo seleccionamos en la propiedad Source Image de nuestro GameObject Button

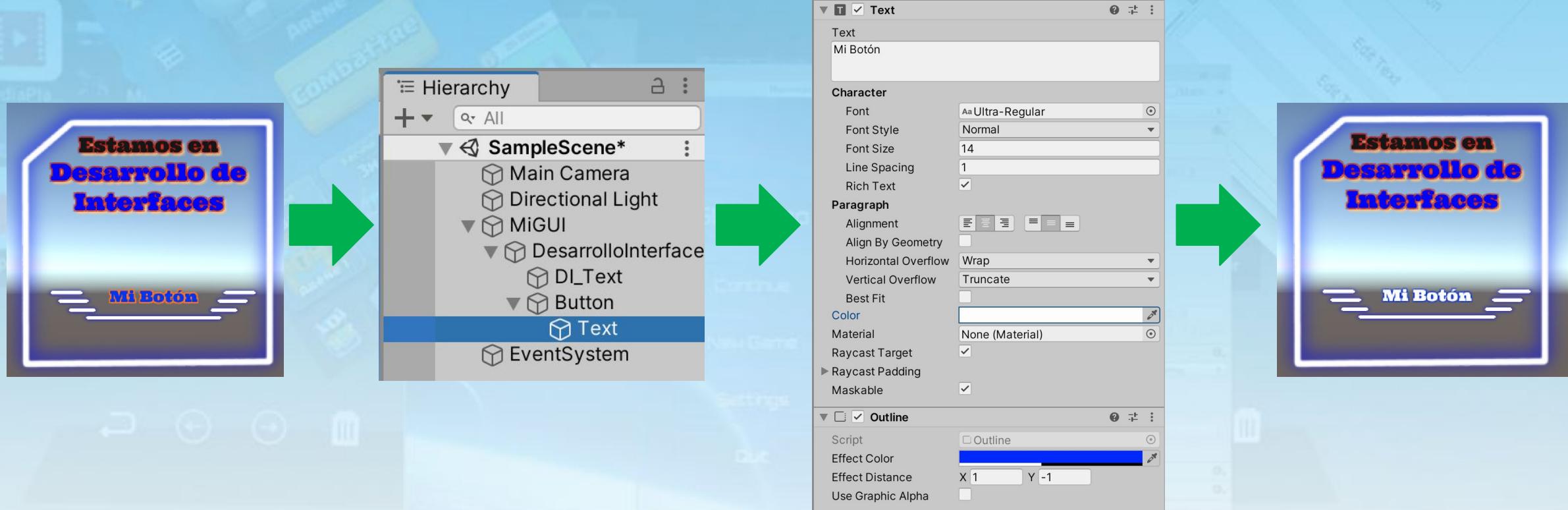


# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

Button

## Source Image.

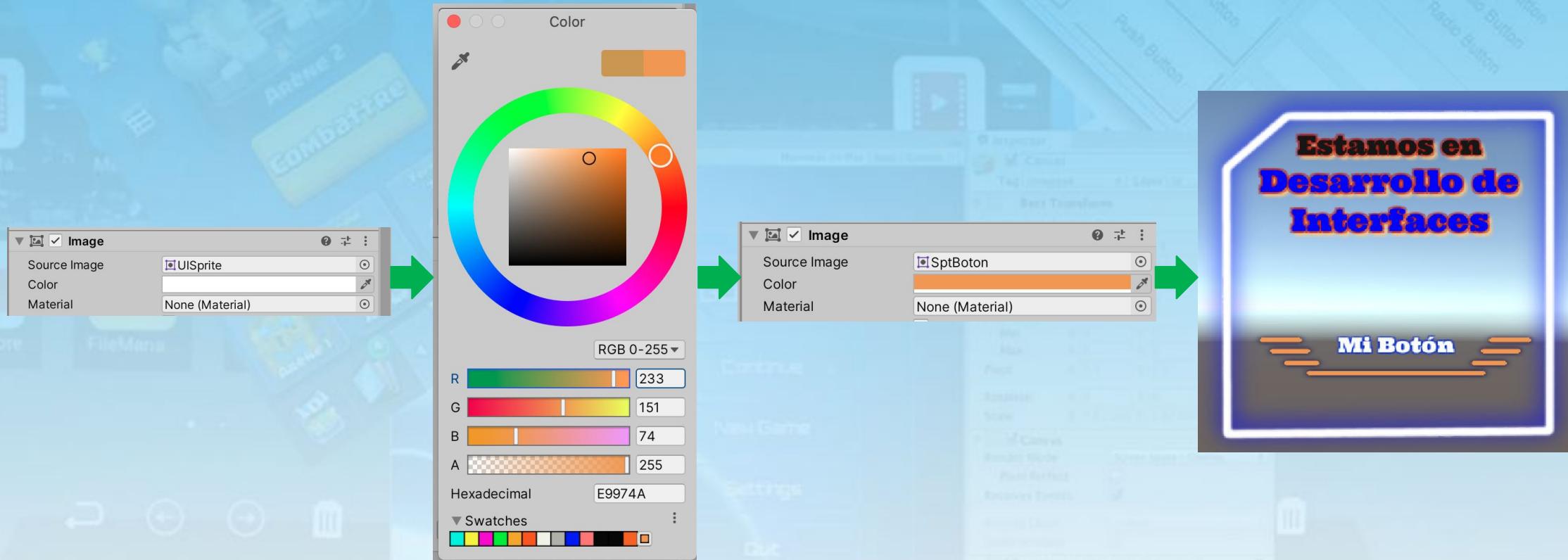
Como ya hemos dicho, podemos modificar los parámetros del GameObject Text para buscar un resultado mejor, colores similares a la imagen, colocar el texto, redimensionarlo, etc....



# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Color.

Modificando el color variará el aspecto del Sprite que determina la forma de nuestro botón.



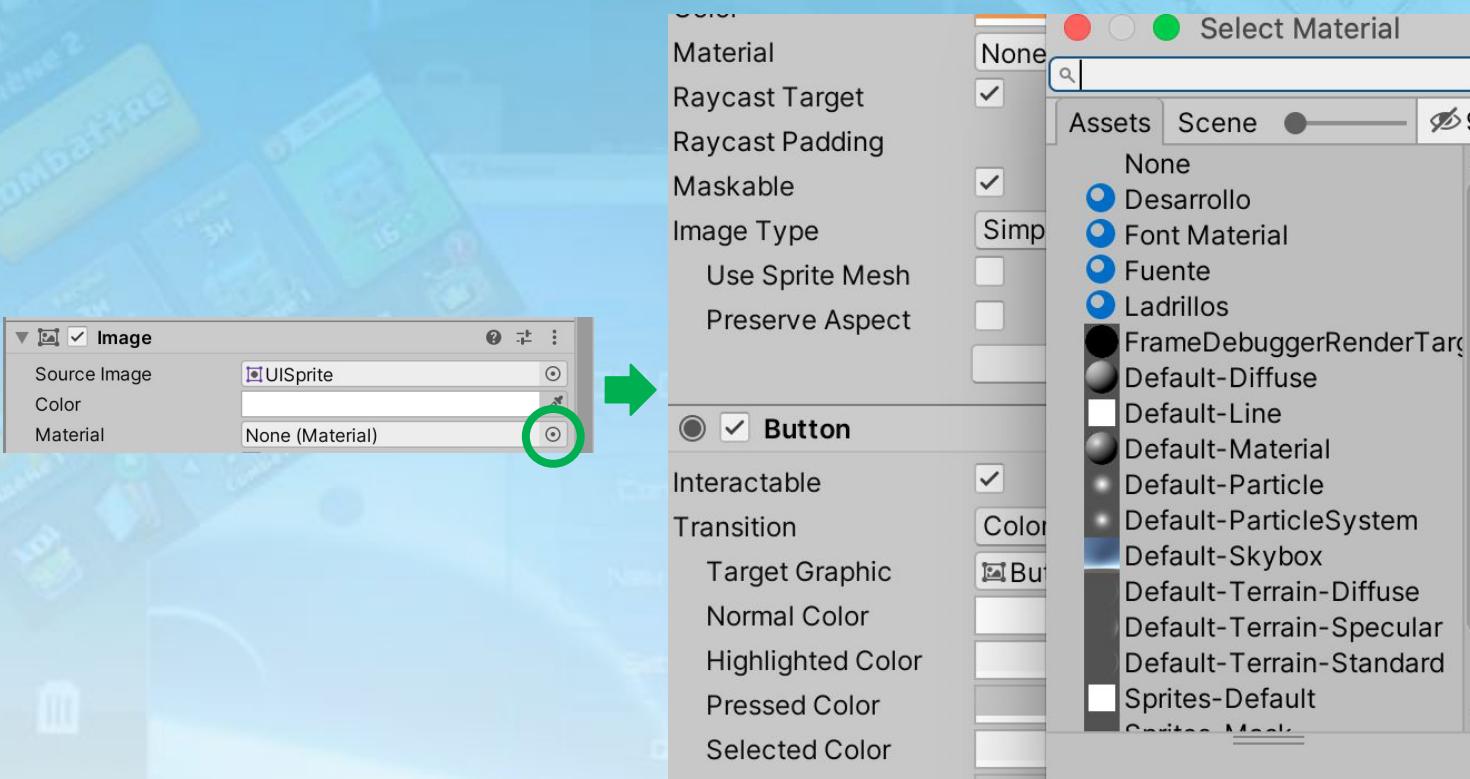
Ver el vídeo **UT2-CreacionSpritesColorMateriales**

Profesores: Raquel Rojo y Mario Santos.

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Material.

Al igual que con los Paneles, podemos crear un material específico para nuestro botón.



Ver el vídeo **UT2-CreacionSpritesColorMateriales**

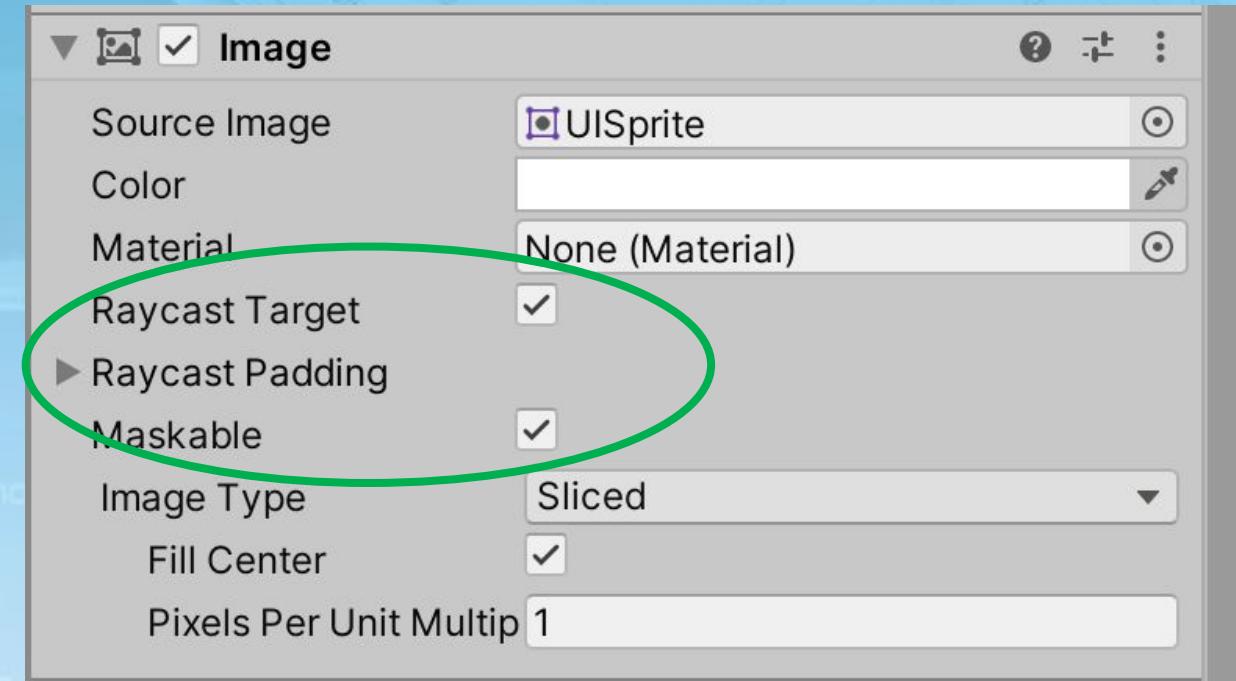
Profesores: Raquel Rojo y Mario Santos.

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

Los siguientes tres parámetros:

- RayCast Target
- Raycast Padding
- Maskable

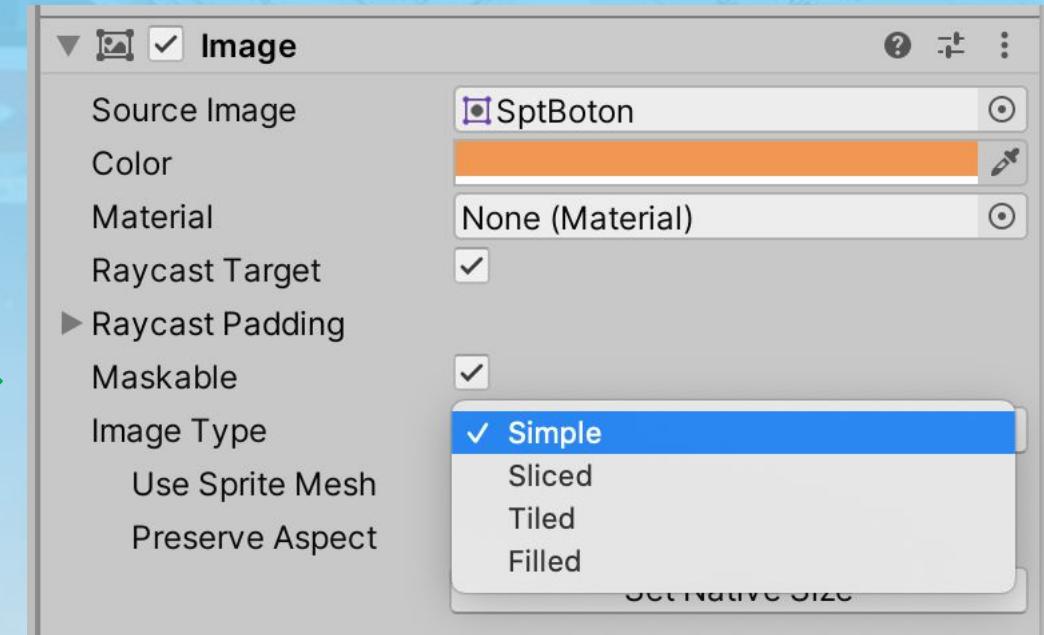
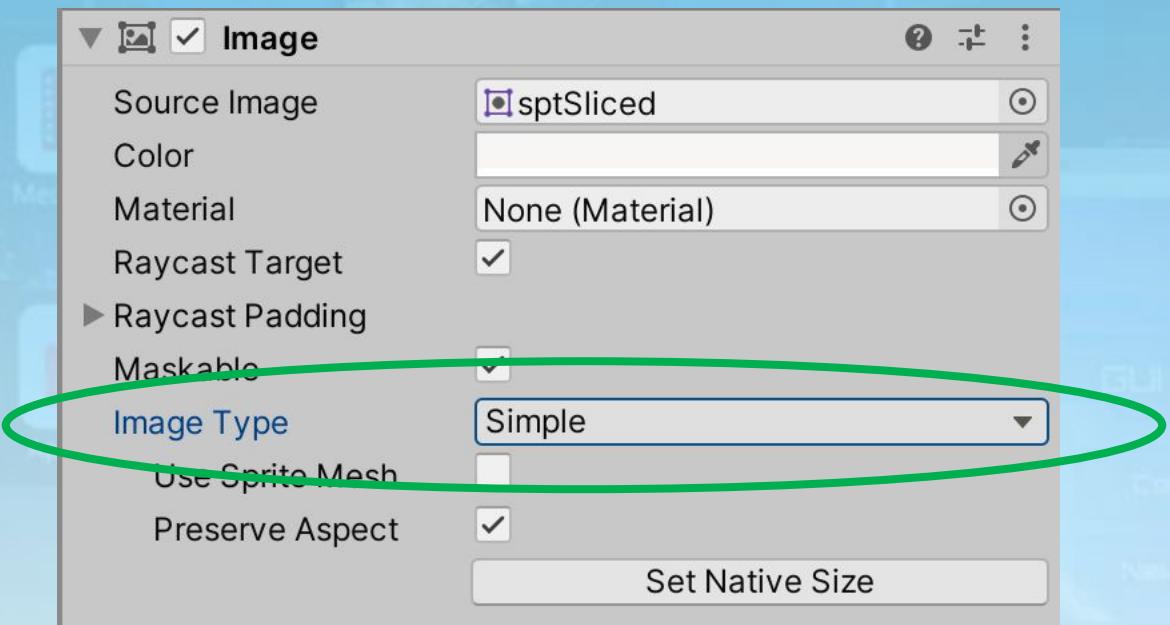
los veremos en la UT en la que trabajamos con VR.



# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Button

El parámetro de Image Type indica la forma de representar la Imagen. Si desplegamos la lista, nos encontramos con las siguientes opciones:



### • Simple

Viene como predeterminada. Con esta opción podremos escalar y modificar la imagen del Button.



[Ver vídeo UT2\\_ImageType](#)

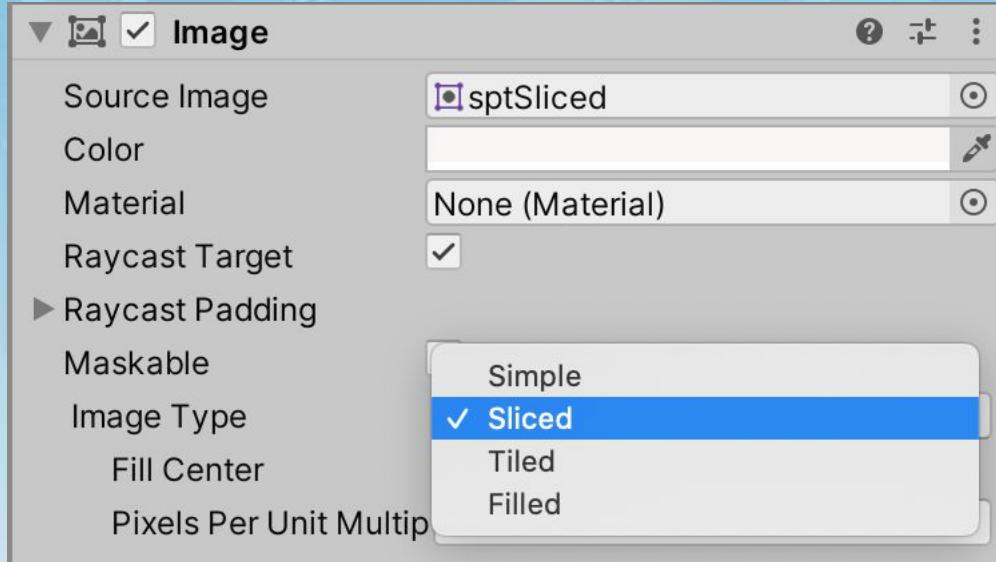
Profesores: Raquel Rojo y Mario Santos.

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Button

- **Sliced:**

Este tipo de imagen nos permite trabajar con imágenes de tipo Sprite, que teniendo algún tipo de borde, este, nunca pierde la proporción. Una imagen de este tipo sería la que se muestra a continuación.



Fijarse como el marco del botón nunca pierde la proporción, aunque la modifiquemos el tamaño



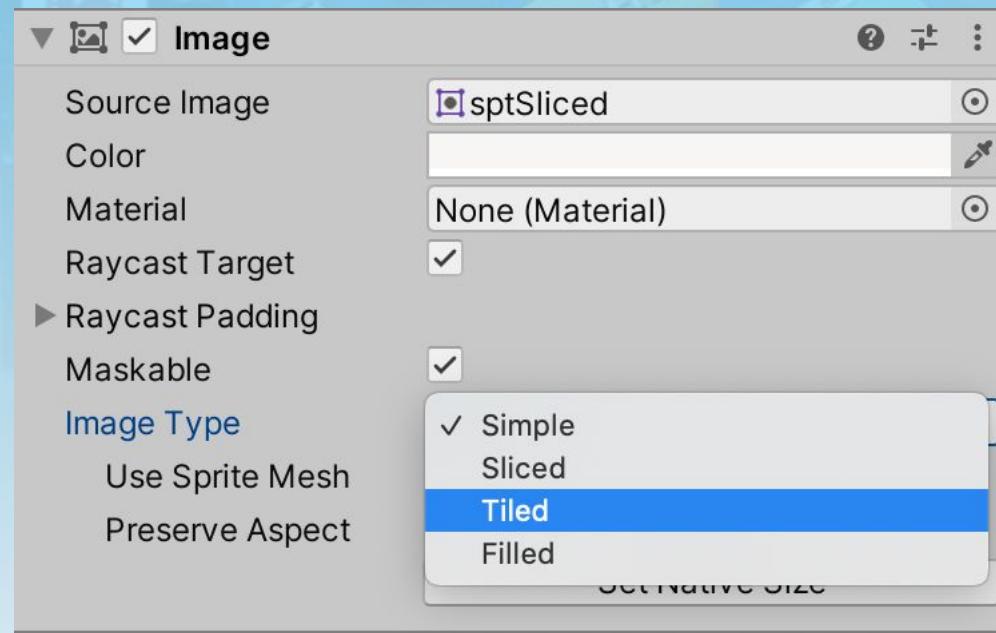
[Ver vídeo UT2\\_ImageType](#)

Profesores: Raquel Rojo y Mario Santos.

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

- **Tiled:**

Este tipo de imagen nos permite trabajar con imágenes de tipo Sprite y que son tileables, es decir, se repiten n veces hasta llenar el área del botón.



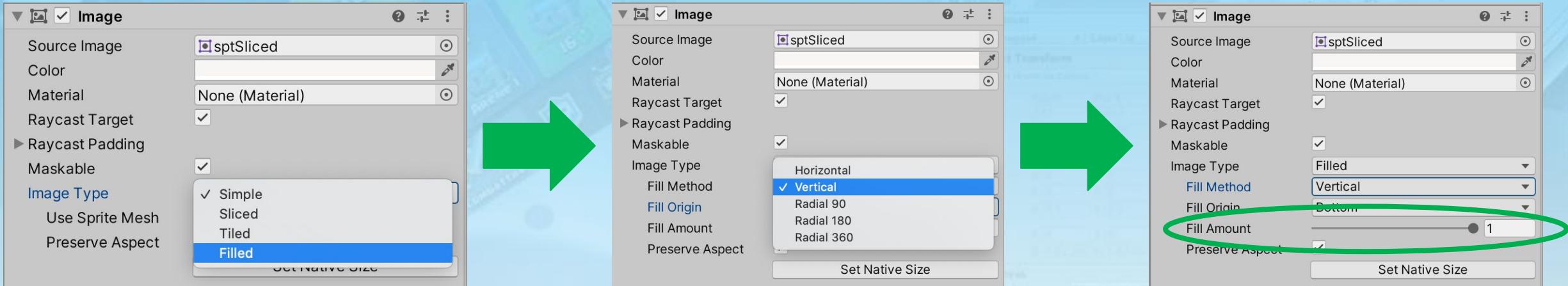
[Ver vídeo UT2\\_ImageType](#)

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Button

- **Filled:**

Este tipo de imagen nos permite crear animaciones desde una Sprite. Se pueden hacer animaciones horizontales, verticales y circulares ( $90^\circ$ ,  $180^\circ$  y  $360^\circ$ ). Para crear la animación debemos acceder al Fill Amount desde un Script. Esto se verá en la última parte de esta U.T.



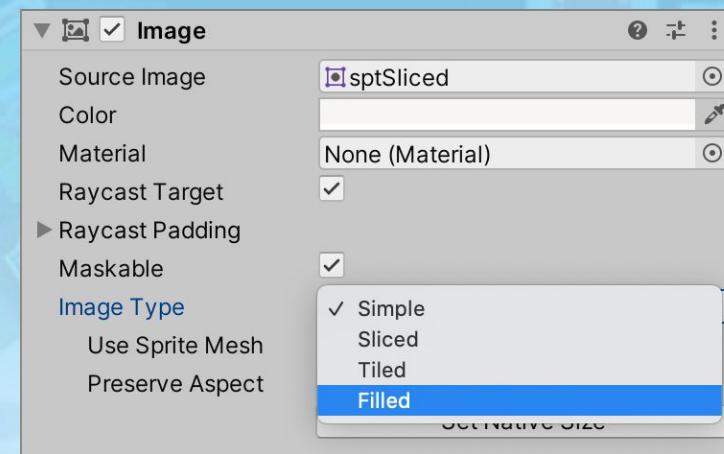
[Ver vídeo UT2\\_ImageType](#)

Profesores: Raquel Rojo y Mario Santos.

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Button

El resto de parámetros del componente **Image**, son intuitivos. Nos encontramos **Preserve Aspect** y **Set Native Size**, el primero bloquea el aspecto de la imagen para no perder las proporciones y el segundo carga la imagen al tamaño que fue creada.



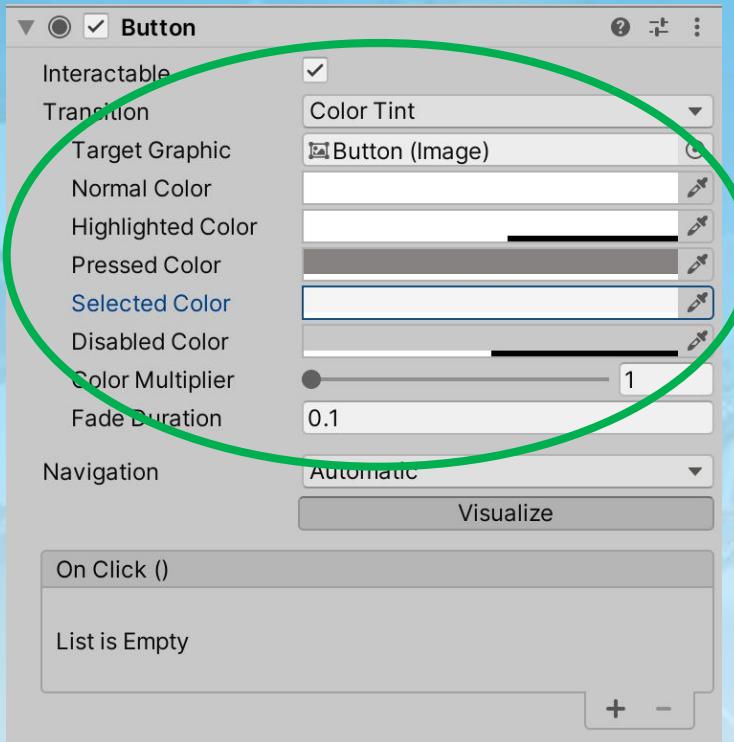
Ahora, vamos a ver los parámetros del **componente Button**.



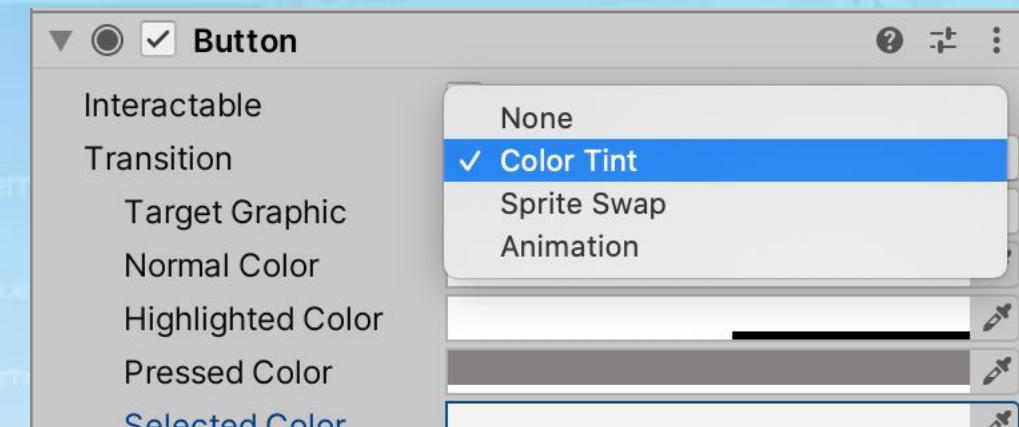
[Ver vídeo UT2\\_ImageType](#)

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Componente Button:



Los primeros parámetros son muy intuitivos y varían según el tipo de Transición seleccionado. Para ver como funcionan, ver el vídeo UT2\_TransitionButton.

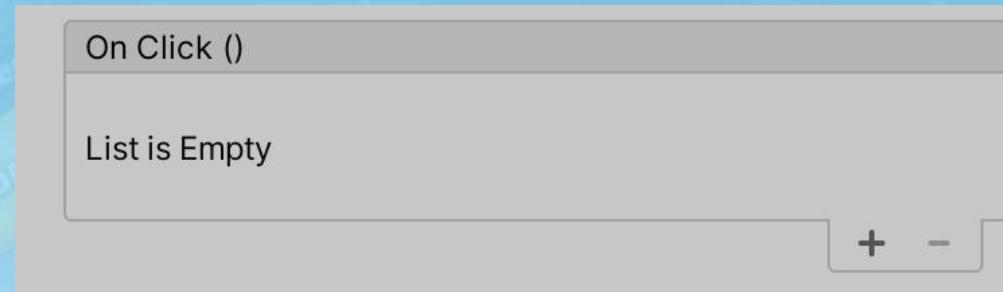


[Ver vídeo UT2\\_TransitionButton](#)

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

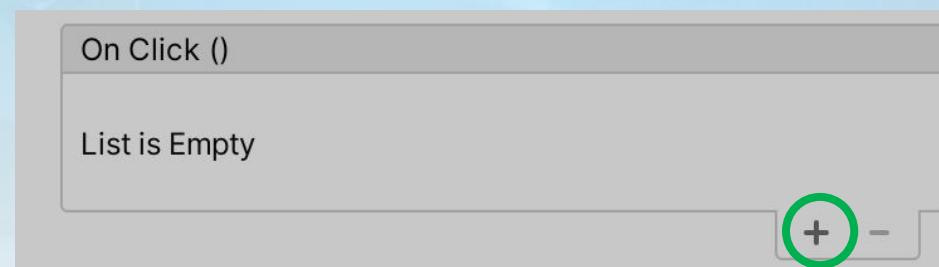
## Componente Button:

A nosotros, de este componente, quizás, el parámetro que más nos importa es el que se muestra en la imagen.



Es en este parámetro en el que vamos a especificar la función/funciones que se van a ejecutar cuando pulsemos sobre el botón (lo veremos en UT2\_ParteE).

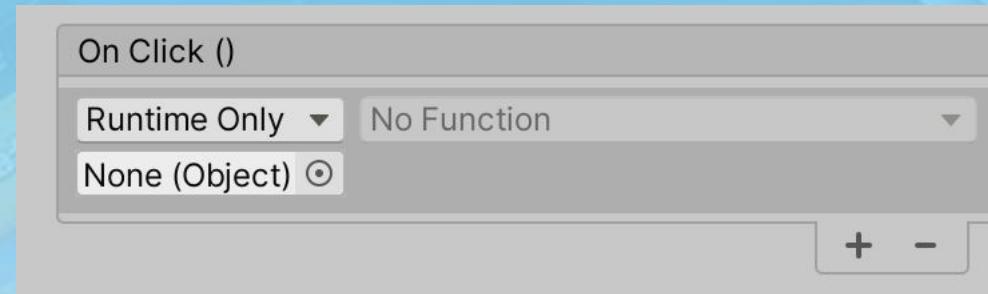
Lo primero que debemos de hacer es pulsar sobre el + que se encuentra en la esquina inferior derecha, para crear el nuevo evento.



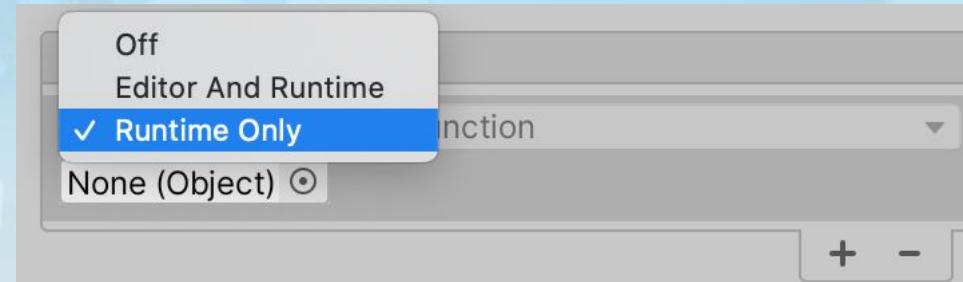
# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Componente Button:

Una vez pulsado nos vamos a encontrar



En el primer parámetro podemos decir que se ejecute la función solo cuando estemos en Ejecución, cuando estemos en ejecución o Edición, o nunca.



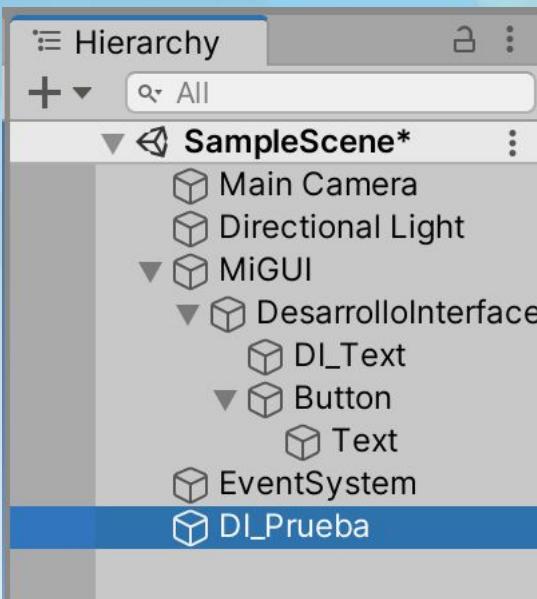
# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

## Componentes contenedores básicos.

Button

### Componente Button:

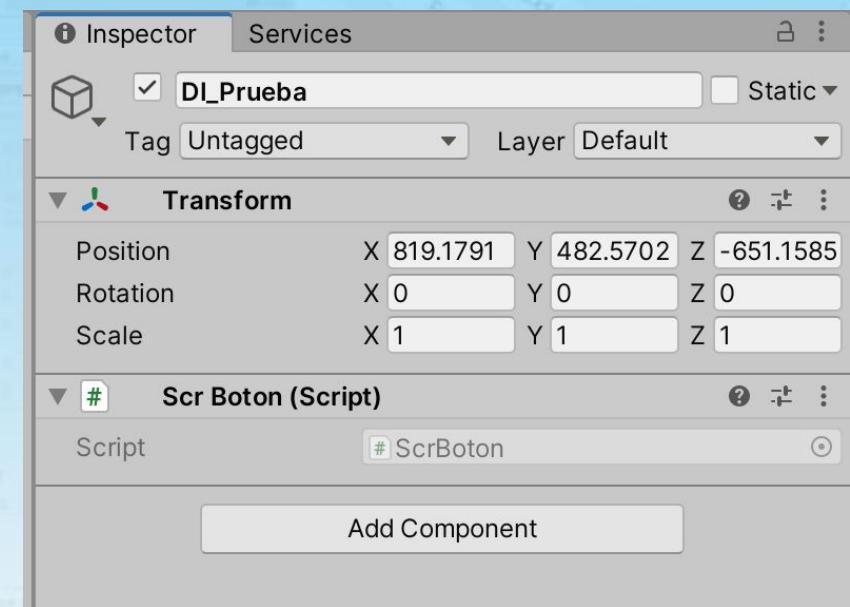
Para el segundo parámetro hay que tener en cuenta lo que ya se dijo en la UT1, "Un script siempre va asociado a un GameObject". Bien, pues en este segundo parámetro, se debe de cargar ese GameObject. Imaginemos que tenemos un GameObject en la ventana Hierarchy que se llama DI\_Prueba. Dentro de este GameObject hay un script denominado ScrBoton, que tiene una función que va a visualizar un mensaje por consola ("Hola, mundo!!!").



```

1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  public class ScrBoton : MonoBehaviour
6  {
7      public void MiMensaje()
8      {
9          Debug.Log("Hello, World!!!");
10     }
11 }
12

```



# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

Button

## Componente Button:

Ahora cargamos el GameObject en el segundo parámetro del evento OnClick, arrastrando desde la ventana de Hierarchy sobre el evento Onclick().



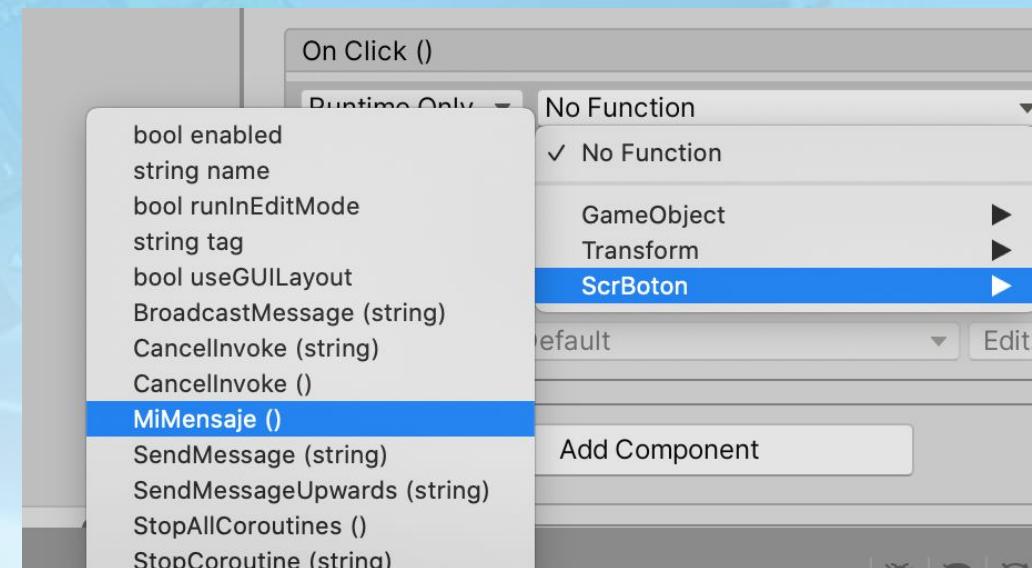
## Button

**Componente Button:**

En ese momento, ya se ha cargado el Script.

Ahora sólo nos queda elegir la función que queremos que se ejecute. Esto lo vamos a indicar en el tercer parámetro.

Al pulsar sobre el nombre del script, nos aparecerán todos los métodos públicos.



Lo veremos con más profundidad en UT2\_ParteE

Profesores: Raquel Rojo y Mario Santos.

## Prefabs

Por último, en esta parte de la U.T.2. nos queda hacernos una pregunta, si creamos un Panel, un Text o un botón personalizado,

**¿tendríamos que hacer todos los pasos para todos los Panels, Texts o Buttons?**

La respuesta es que **NO**, una vez creado un componente con sus parámetros, lo podemos guardar y utilizarlo todas las veces que deseemos, a modo de plantilla.

Incluso, desde el propio código, podríamos ir creando **instancias** de este componente personalizado.

Este componente personalizado, se llama **PREFAB** y la forma de trabajar con ellos es la que se muestra en la siguiente diapositiva.

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales:

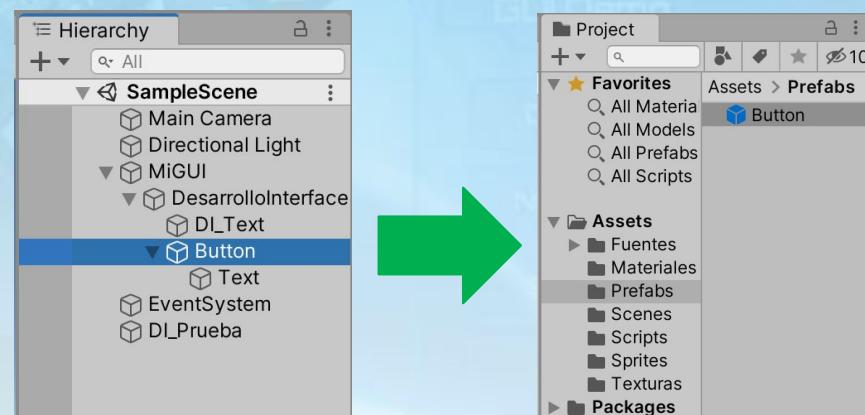
## Componentes contenedores básicos.

### Prefabs

Lo primero que vamos a hacer, es crear una carpeta, en nuestra ventana Project, en la que guardaremos todos nuestros prefabs. Esta carpeta no es obligatoria, pero sigue la filosofía de este curso respetando el buen hacer y el orden en los proyectos. A esta carpeta la vamos a denominar Prefabs.



Hecho esto, vamos a ir a nuestra ventana Hierarchy y vamos a seleccionar el GameObject que queremos convertir a Prefab. Una vez seleccionado, dejaremos el botón izquierdo del ratón pulsado y lo arrastramos dentro de nuestra carpeta Prefab.

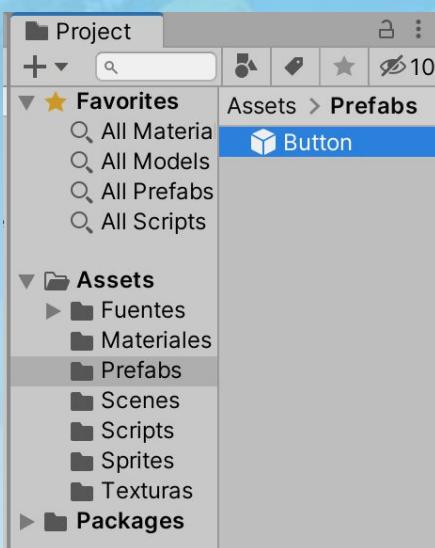


Fijarse como ha cambiado el icono representativo del botón, ahora es un cubo azul, esto nos indica que es un prefab.

# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

## Prefs

Fijarse que en la ventana Hierarchy aparece también con el icono del cubo azul, indicando que es un prefab. Ahora lo único que habría que hacer es arrastrar nuestro prefab desde la carpeta Prefs hasta nuestra ventana Scene, de esta manera podremos añadir al proyecto todos los botones personalizados, que necesitemos.



Arrastrar

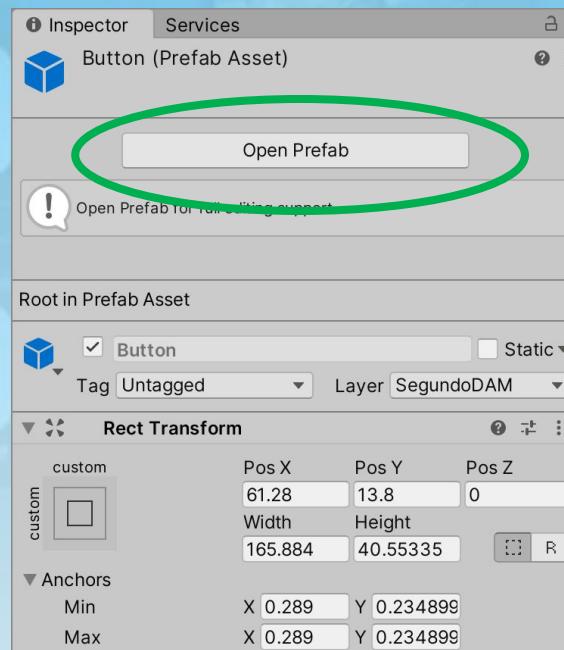


# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

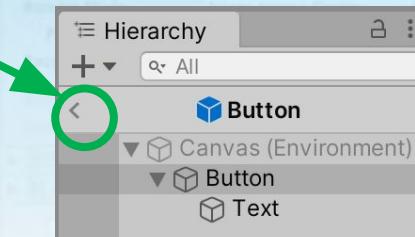
## Prefabs

Debemos observar varias cosas:

Si seleccionamos el prefab, vemos que nos aparece la siguiente información en la ventana Inspector.



- Si pulsamos sobre el botón “Open Prefab”, abrimos las propiedades del prefab y hay que tener claro que cualquier cambio que hagamos va a repercutir sobre todas las instancias del mismo.
- Si por el contrario, se hace un cambio en una instancia, ese cambio, solo le afecta a la instancia, pero no al resto.
- Cuando abrimos un prefab, la ventana Hierarchy cambia y nos muestra exclusivamente ese prefab, para volver al modo de edición, pulsar sobre este botón.



# UT2.- Introducción a la generación de interfaces gráficas de usuario con editores visuales: Componentes contenedores básicos.

Prefabs



En la siguiente parte de la U.T.2 veremos como crear scripts para poder trabajar con nuestros paneles, botones y cajas de texto.