# Unity Editor

## Atributos

Encabezado: Sirve para poner una cabecera en el editor.

[Header(“nombre”)]

Slider: Pone un slider para cambiar el valor de una variable.

[Range(minimo, máximo)]

Multilinea: Un texto que aparezca como multilinea.

[Multiline(numero\_lineas)]

[TextArea(numero minimo, numero máximo)]

Ocultar Variable: Para ocultar una variable publica en el inspector.

[HideInInspector]

Mostrar Variable privada: Muestra la variable privada en el editor.

[SerializeField]

Espacio de Separacion: Añade un espacio de separación en el editor.

[Space(numero\_pixels)]

Texto de ayuda: Al estar con el mouse encima muestra un texto.

[Tooltip(“texto”)]

Poner que solo puede haber un componente en el objeto, se pone encima de la definición de la clase:

[DisallowMultipleComponents]

Se asegura de que el objeto donde esta nuestro script, tiene otros componentes a los que hace referencia:

[RequireComponent(typeof(componente))]

## EDITOR PERSONALZIADO

Si se quiere tener un editor personalizado para nuestro componente, lo primero de todo creamos un script que herede de Editor en lugar de MonoBehavour y le añadimos a que script modifica. Después tenemos que sobrescribir el método OnInspectorGui y debajo del base se le añade el código. La variable target tiene la información del Objeto que tiene el Script. En el ejemplo pintamos dos botones y ejecutamos unas funciones:

using UnityEditor;

[CustomEditor(typeof(ScriptName))]

public class GeneradorEditor : Editor

{

public override void OnInspectorGUI()

{

base.OnInspectorGUI();

Generador generador = (Generador)target;

if (GUILayout.Button("Generar Mapa")) {

generador.GenerarMapa();

}

if (GUILayout.Button("Limpiar Mapa")) {

generador.LimpiarMapa();

}

}

}

# Propiedades Editor Personalizado

Pintar Botón: Pinta un botón que si se pulsa devuelve True:

GUILayout.Button(“Text”)

## Tags:

Se pueden añadir submenús de la siguiente manera:

Enemigo/Esqueleto.

Si se intenta hacer el findObjectwithtag hay que referenciar al submenú también.

## Tips Menu:

Si le das en el inspector a las tres líneas, puedes cambiar al modo Debug donde salen muchos más campos en cada Componente del GameObject.

Si pulsas shift cuando mueves un objeto se moverá recto en la primera dirección que arrastres.

## Activar Funcion desde Editor:

Para activar una función desde el editor se tiene que poner encima de la función:

[ContextMenu(“Nombre Funcion”)]

Ahora cuando se de al engranaje en el componente script, saldrá el “Nombre Funcion” para que le pulsemos. Solo se puede usar para funciones sin parámetros.

## Comentarios de Metodos

Para agregar comentarios a un método para que al llamarlo desde otro lado salga descripción se pone /// y te lo añade solo para que escribas. Ejemplo:

/// <summary>

///

/// </summary>

/// <param name="ancho"></param>

/// <param name="alto"></param>

/// <param name="vacio"></param>

/// <returns></returns>