Unity 2020
Interfaz de Usuario (UI)

Introducción

La Interfaz de usuario

Cuadro 2

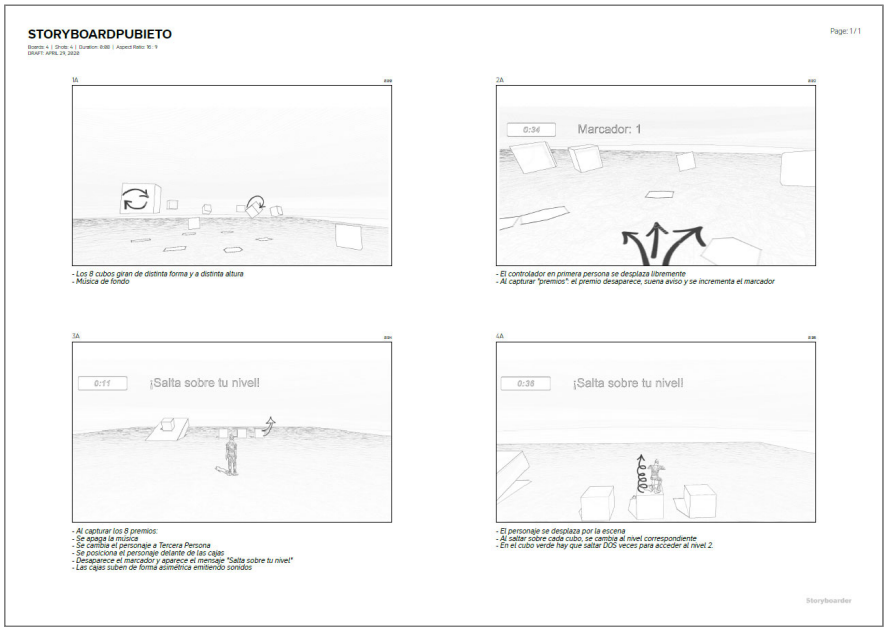
1 Marcador
2 Cronómetro

Menú de opciones

1 Escena de Menú
2 En las escenas

0 Introducción

- Para completar la escena inicial, añadiremos los elementos que faltan de la interfaz de usuario
- Finalizaremos el tema definiendo un menú para todas las escenas que permita al usuario cambiar de escena o finalizar la aplicación



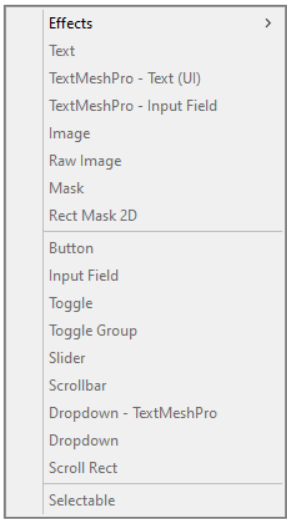
La interfaz de usuario
Elementos de la UI

- La interfaz de usuario (UI) se puede confeccionar combinando tres tipos de elementos (https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/index.html)

a) Elementos pasivos
   - Text / TextMeshPro (textos con texturas)
   - Image / Raw image (imágenes con más funcionalidades)

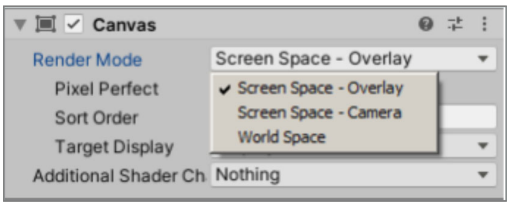
b) Elementos interactivos
   - Button / Button TextMeshPro
   - Toggle
   - Slider
   - Scrollbar

c) Elementos agrupadores
   - Canvas (elemento base de la interfaz)
   - Panel
   - Scroll view
   - Event System



La interfaz de usuario
Canvas

- Es el lienzo donde se sitúan los elementos del UI
- Canvas trabaja en varios modos:



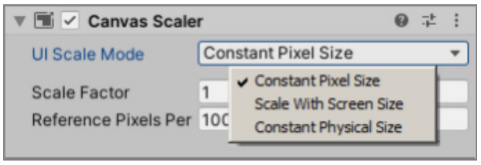
a) Screen space - Overlay
   - Ocupa pantalla entera
   - No se puede modificar su *Transform*

b) Screen space – Camera
   - Canvas separados para aplicaciones con varias cámaras
   - Para aplicaciones con pantalla partida

c) World space
   - Se puede cambiar su Transform
   - Permite canvas en 3D: flotar sobre el escenario, realidad virtual (se desplaza según tu punto de vista)

La interfaz de usuario
Escalado del Canvas

- Se pueden establecer varios criterios para escalar los elementos del Canvas



a) Constant pixel size

- Obsoleto
- Según la resolución cambia mucho el tamaño del objeto

b) Scale with screen size

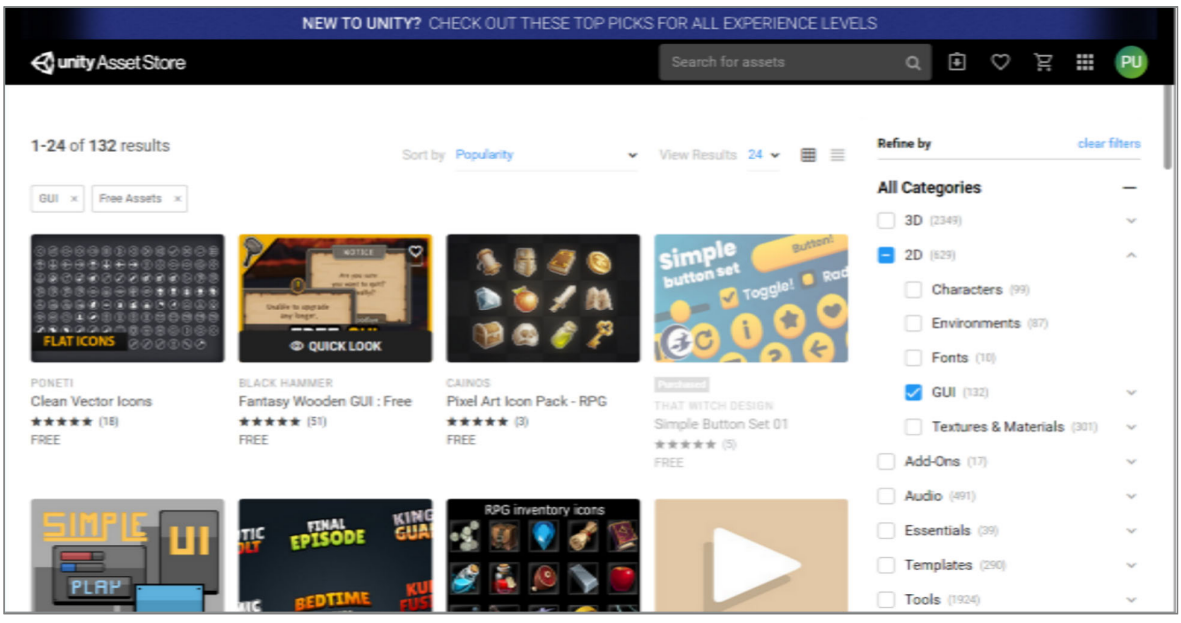
- Recomendable
- Debe coincidir con el tamaño de la pantalla de referencia (por ejemplo 1920 x 1080)
- Recomendación:
  - Screen Match Mode: *Match Width or Height*
  - *Match*: 0.5

c) Constant physical size

- Peligroso en pantallas pequeñas

La interfaz de usuario
Importamos iconos

- Para hacer los ejercicios, vamos a importar una serie de iconos gratis de la Asset Store de Unity. También en https://www.flaticon.com/



La interfaz de usuario

Posicionar elementos en el Canvas

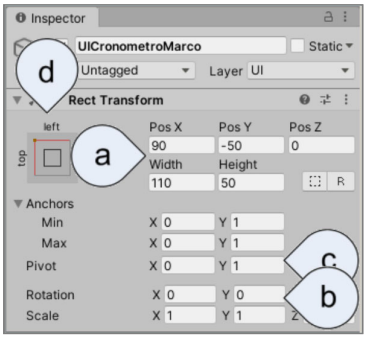
- Para posicionar los elementos de la interfaz en el *Canvas*, hay que tener varias cuestiones en cuenta:

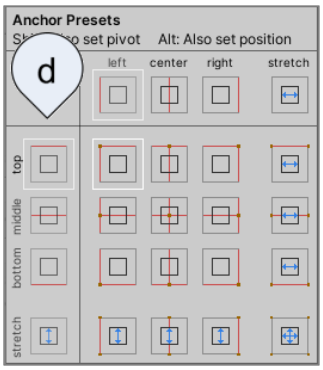
a) Situación y dimensiones
   - La situación del *Pivot* respecto del *Anchor*
   - Dimensiones del elementos

b) Es mejor no modificar los valores de *Rotation y Scale*

c) *Pivot:* es el punto de referencia del elementos
   - X=0, Y=0 ⟶ Esquina inferior izda.
   - X=0.5, Y=0.5 ⟶ Punto medio/centro
   - X=1, Y=1 ⟶ Esquina superior dcha.

d) *Anchor (punto de anclaje del elemento):* punto con el que intenta mantener las distancias indicadas, si varía el tamaño de la pantalla
   - Pulsando las teclas *Shift* y/o *Alt* se cambia el *Pivot* y/o la posición del elemento





La interfaz de usuario

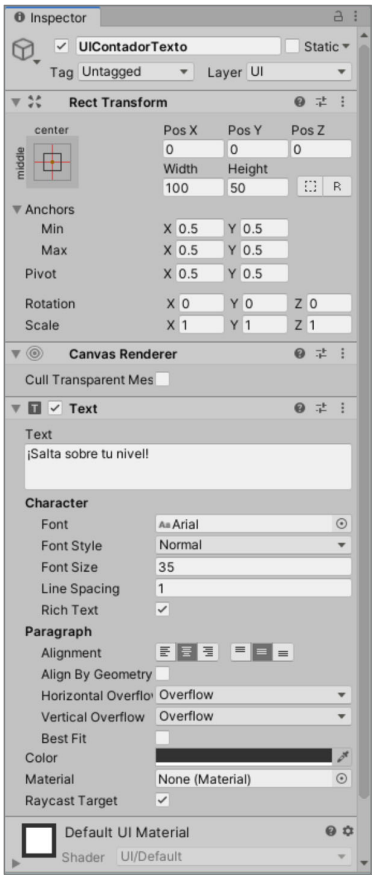
UI - Text

- Objeto que representa un texto en la interfaz de usuario
- Puede ir ligado a otros elementos, como a botones
- En cuanto a los datos del Inspector:

a) El *Rect. Transform* es igual que para el resto de elementos

b) Se pueden definir las características típicas de un texto en cualquier programa
   - Es interesante comprobar los valores de *Horizontal* y *Vertical Overflow*, ya que pueden afectar a la representación final del texto si ocupa más espacio del rectángulo fijado

- Se pueden descargar fuentes gratuitas desde: https://www.1001freefonts.com/

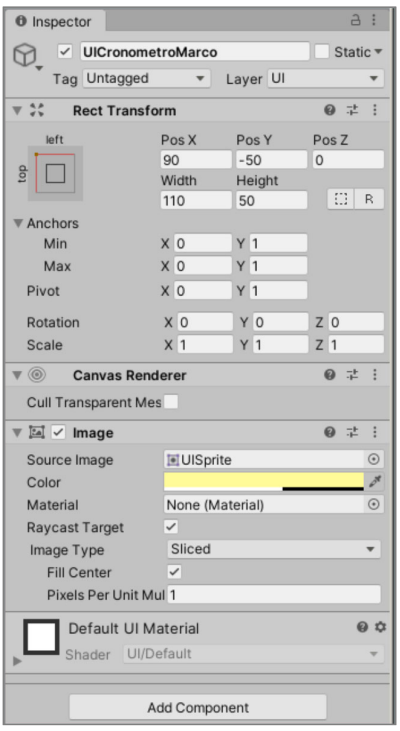


La interfaz de usuario
UI - Image

- Objeto que representa una imagen en la interfaz de usuario
- En cuanto a los datos del Inspector:

a) El *Rect. Transform* es igual que para el resto de elementos

b) Se pueden definir las características típicas de una imagen en cualquier programa
   - El fichero de imagen se indica en el control *Source Image*
   - Puede ser interesante convertir las imágenes (sobre todo si son importadas) a escala de gris, para darle un color de fondo con la opción *Color*



La interfaz de usuario
UI - Button

- Elemento interactivo. Al pulsarlo, se ejecuta el procedimiento indicado en el componente *OnClic()*

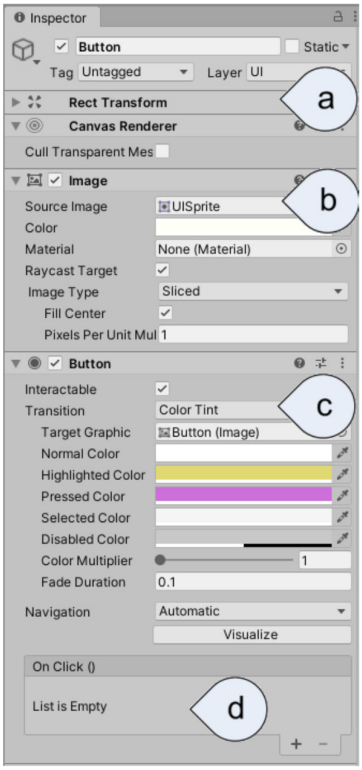
a) *Situación y dimensiones*
   - Igual al resto de elementos

b) Se puede indicar una imagen para el botón, con el color indicado

c) Se puede indicar el modo de interacción, según el estado del botón (inactivo, pulsado, ratón encima):
   - Color Tint: Cambia el color del botón
   - Sprite swap: cambia la imagen

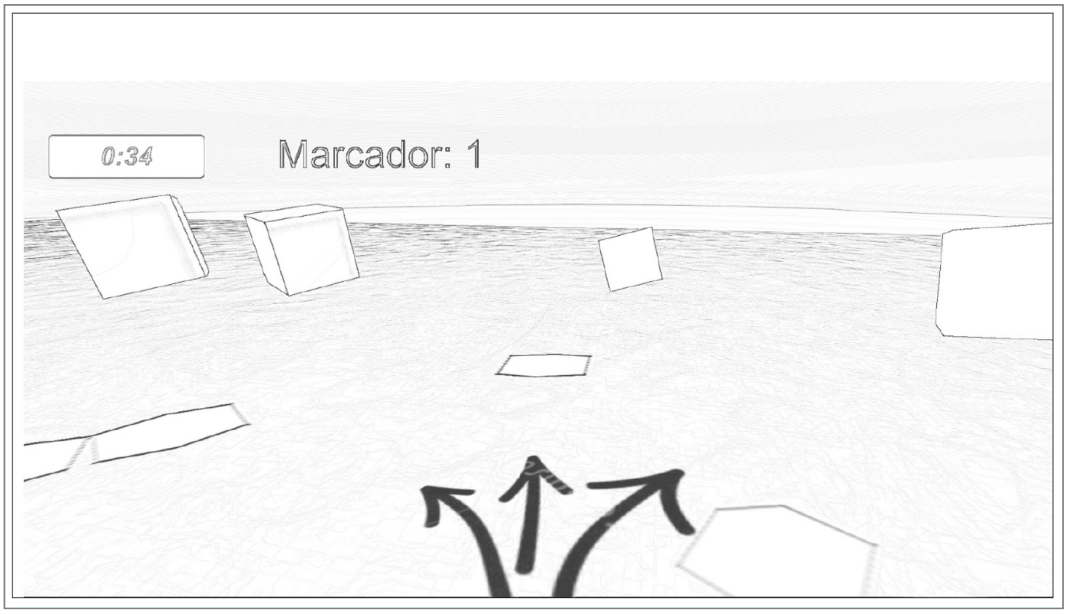
d) *OnClic():* Procedimiento(s) a ejecutar al pulsar el botón:
   - Hay que añadir el GameObject que contiene el Script.
   - Del Script, se selecciona la función deseada

- Todo botón tiene un texto como elemento hijo



Cuadro 2

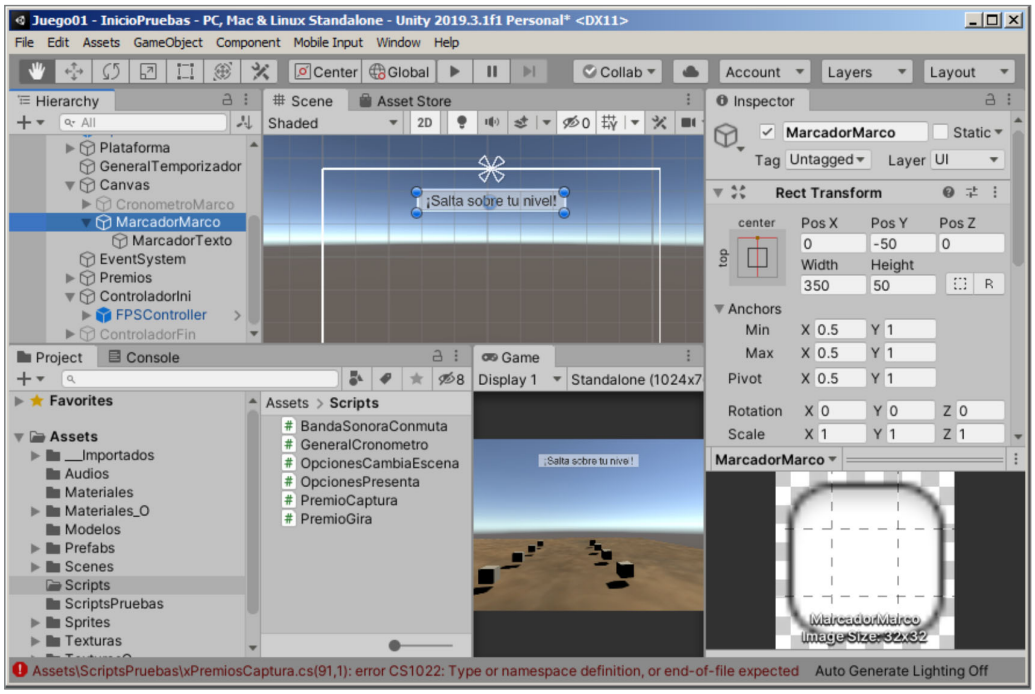
- Directrices del cuadro 2:
  1. El controlador se desplaza libremente por la escena colisionando con los premios.
  2. Al colisionar con los premios, los captura:
     - El premio desaparece, suena un aviso acústico y se incrementa el marcador
  3. Se visualiza un cronómetro
     - Cuando pase un tiempo, se indica cambiando a color rojo



Cuadro 2. Marcador

1.1 UI – Definir marcador

- Añadimos una imagen (anclada en el punto superior medio), para que sirva de Marco, y un texto para reflejar el estado del marcador



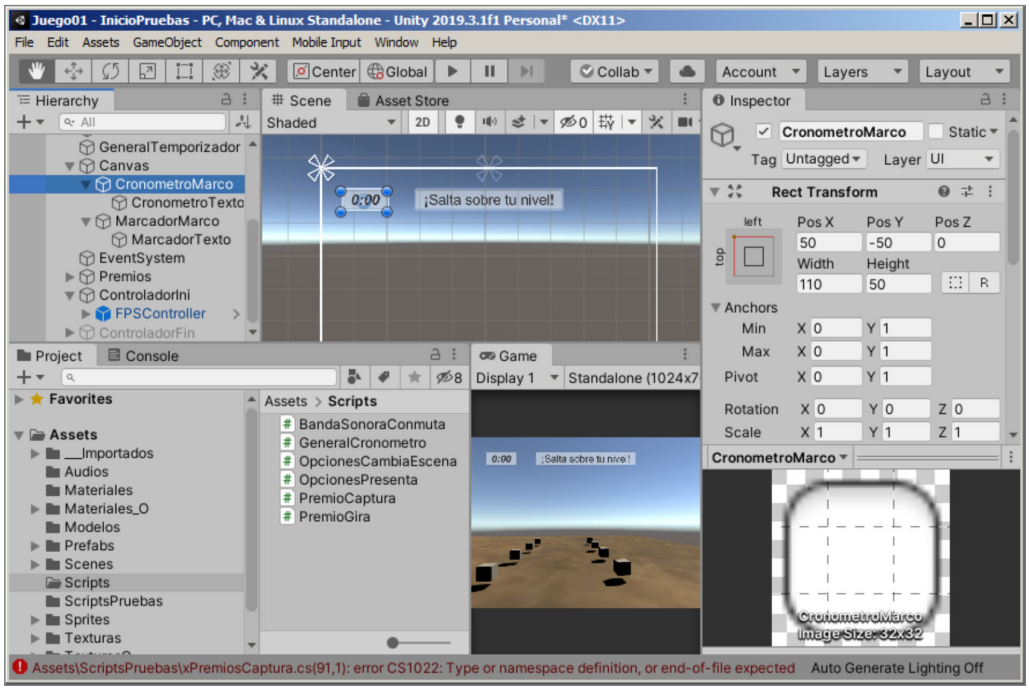
Cuadro 2. Marcador
1.2 UI – Script para marcador

- Añadimos un par de líneas en el script *PremioCaptura.cs*

```
using UnityEngine.UI; //  Para usar el UI

public class xPremiosCaptura : MonoBehaviour
{   public AudioClip pickSound;  // Audio CLip en el que guardamos el sonido a emitir cuando se captura premio
    int iMarcador;  //  Variable para el marcador
    string msgMarcador;  //  Cadena en la que ir completando el mensaje que aparecerá en el marcador

    [SerializeField]
    private Text TextoMarcador;

    public GameObject ControlIni, ControlFin;  //  Variables con los dos controladores, para activar/desactivarlos de forma más cómoda

    // Start is called before the first frame update
    void Start() {
        iMarcador = 0;    //  Inicializamos marcador a cero
        // GameObject.Find("MarcadorTexto").GetComponent<Text>().text = "Marcador: 0"; //  Presentamos mensaje en la interfaz
        TextoMarcador.text = "Marcador: 0"; //  Presentamos mensaje en la interfaz
    }

    private void OnControllerColliderHit(ControllerColliderHit loColisionado) {
        string etiqueta = loColisionado.collider.gameObject.tag;  // Guardamos en 'etiqueta' el tag del objeto que ha colisionado

        if (etiqueta == "Premio")  //  Actuación en caso de colisión con un Premio {
            Debug.Log("colisión con " + etiqueta); //  Aviso para debug. Se puede desactivar
            Destroy(loColisionado.collider.gameObject);
            gameObject.GetComponent<AudioSource>().clip = pickSound;  //  Leemos el fichero indicado en el componente
            gameObject.GetComponent<AudioSource>().volume = 1.0f;       //  Fijamos el volumen (de 0 a 1)
            gameObject.GetComponent<AudioSource>().Play();              //  Activamos el sonido
            iMarcador += 1;             // Incrementamos iMarcador. También se puede poner iMarcador++;
            msgMarcador = "Marcador: " + iMarcador; // Generamos mensaje a presentar en el marcador de la interfaz
            if (iMarcador >= 8) msgMarcador = "¡Salta sobre tu nivel!";

            Debug.LogWarning(msgMarcador);
            // GameObject.Find("MarcadorTexto").GetComponent<Text>().text = msgMarcador; //  Presentamos mensaje en la interfaz
            TextoMarcador.text = msgMarcador;

            /*** Respuesta al estado del marcador (Explicado en Cuadro 3. Punto 1.1) ***/
            if (iMarcador == 8)
            {   /***  Eliminamos premios que puedan quedar por capturar (Explicado en Cuadro 3. Punto 1.2)  ***/
                GameObject[] matrizPremios = GameObject.FindGameObjectsWithTag("Premio");
```

Cuadro 2. Cronómetro
2.1 UI – Definir cronómetro

- Añadimos una imagen (anclada en la esquina superior izquierda), para que sirva de Marco, y un texto para reflejar el estado del cronómetro



Cuadro 2. Cronómetro
2.2 UI – Script para cronómetro

- Añadimos un par de líneas en el script *GeneralCronometro.cs*

```
5   using UnityEngine.UI; //  Para usar el UI
6
7   public class xGeneralCronómetro : MonoBehaviour
8   {   [SerializeField]
9       Text UICronometro;
10
11      float tiempo, tiempoAnt;
12      int minutos, segundos;
13      string sTiempo, sSegundos;
14
15      // Start is called before the first frame update
16      void Start()
17      {   tiempo = tiempoAnt = 0.0f;
18          GameObject.Find("CronometroTexto").GetComponent<Text>().text = "0:00";
19          UICronometro.text = "0:00";
20      }
21
22      // Update is called once per frame
23      void Update() {
24          tiempo = tiempo + Time.deltaTime; // Al tiempo acumulado, le sumamos el tiempo que ha pasado desde la última frame.
25          if (tiempo > (tiempoAnt + 1))
26          {   tiempoAnt = tiempo;
27              minutos = (int)tiempo / 60;
28              //segundos = (int)tiempo - (minutos * 60);
29              segundos = (int)tiempo % 60; // más rápido
30
31              sSegundos = "" + segundos;
32              if (segundos < 10) { sSegundos = "0" + segundos; }
33
34              sTiempo = minutos + ":"  + sSegundos;
35
36              if (tiempo > 10) //  Avisamos si lleva más de x segundos
37              {   // GameObject.Find("CronometroTexto").GetComponent<Text>().color = Color.red; //  Ponemos el tiempo en rojo
38                  UICronometro.color = Color.red;
39              }
40
41              // GameObject.Find("UICronometroTexto").GetComponent<Text>().text = sTiempo;
42              UICronometro.text = sTiempo; ;
43          }
44      }
```

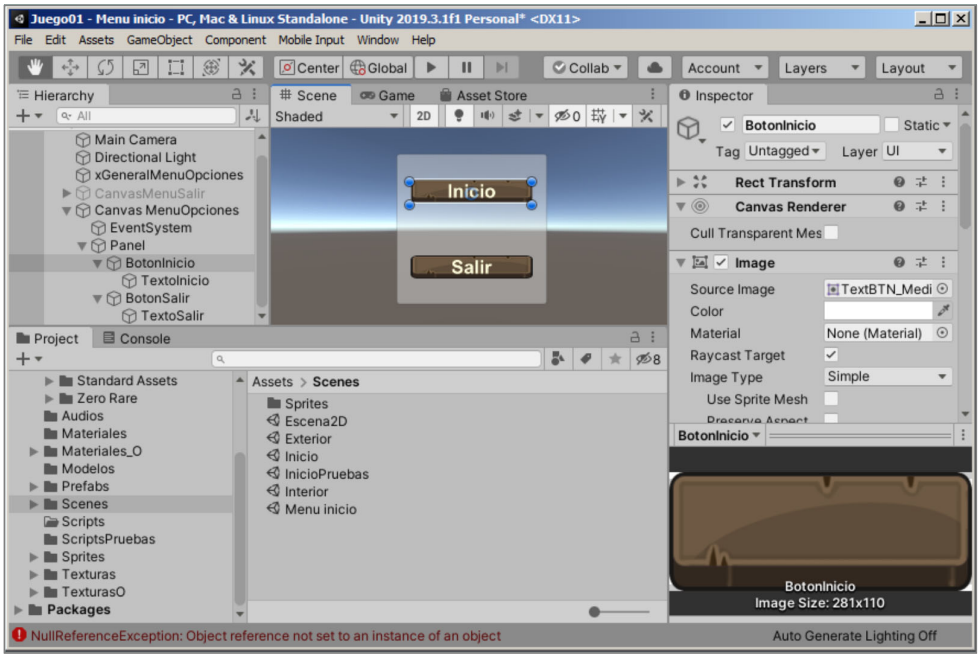

Menú Opciones

- Para completar la interfaz de usuario, vamos a crear un menú de opciones
  1. Se activa pulsando la tecla <*Esc*> en cada una de las escenas
  2. Aparece un menú que permite cambiar a la escena de inicio o salir de la aplicación

Opciones. Escena de Menú

1.1 Crear menú en una escena nueva

- Creamos una escena nueva (Menu Opciones) en la que creamos:
  - Un *canvas* con un *panel* en el centro
  - Dos botones, uno en el que pone Inicio y otro Salir



Opciones. Escena de Menú

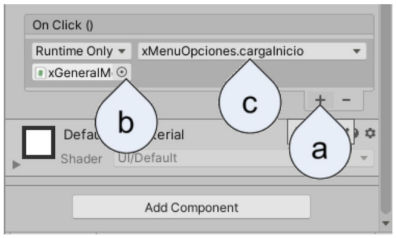
1.2 Script para cambio de escena

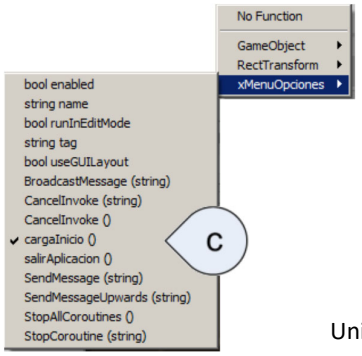
- Creamos el script (MenuOpciones) añadido a un *Game Object* (GeneralMenuOpciones) para gestionar el menú
  - Contiene dos funciones publicas que se llamarán desde los botones
  - Añadimos la opción de pulsar la inicial por si no funcionan los botones

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4
5   using UnityEngine.SceneManagement;
6
7   public class MenuOpciones : MonoBehaviour
8   {
9       // Start is called before the first frame update
10      void Start()
11      {
12
13      }
14
15      // Update is called once per frame
16      void Update()
17      {
18          if (Input.GetKeyDown(KeyCode.I)) CargaInicio();
19
20      }
21
22      public void CargaInicio ()
23      {
24          SceneManager.LoadScene("Inicio");  //  Cargamos la escena de Inicio
25      }
26
27      public void SalirAplicacion ()
28      {
29          Application.Quit(); //  Salimos de Unity (solo válido en la versión comilada)
30      }
31  }
32
```

Opciones. Escena de Menú
1.3 Asignar funciones a cada botón

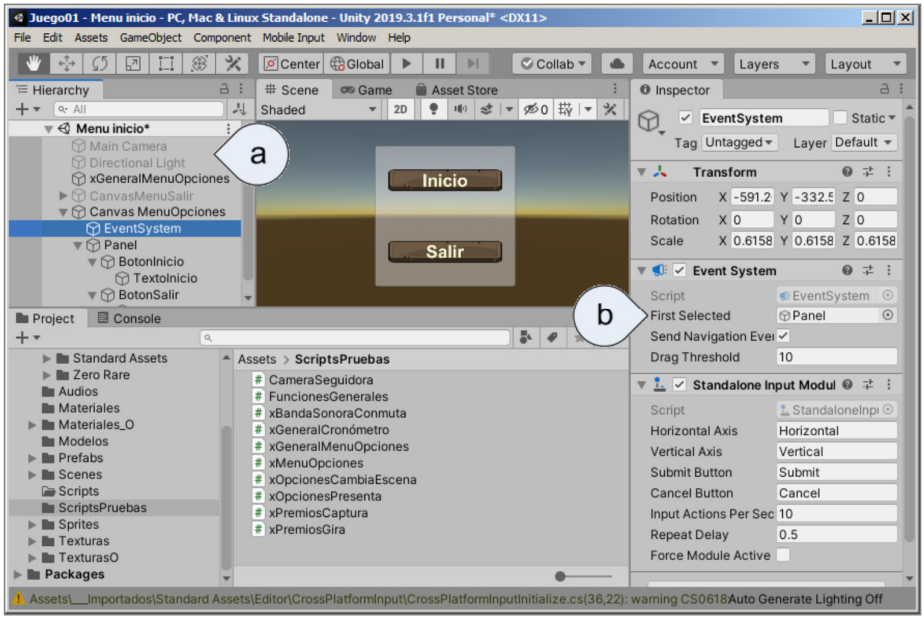
- Para asignar la función a ejecutar al pinchar cada botón, hay que ir a la parte inferior del Inspector de cada botón, en el recuadro *On Clic()*, y:
  a) Pinchar sobre el **+**, para añadir función
  b) Arrastrar el *GameObject* que contiene el script (*GeneralMenuOpciones*) al control *Object*
  c) Elegir la función del desplegable *Function*





Opciones. Escena de Menú
1.4 Finalizar la escena

- Para terminar la escena (hay que tener en cuenta que la vamos a cargar, de forma parcial, dentro de otras
  a) Desactivar la cámara y la luz
  b) Indicar que el *Panel* es el elementos *First Selected* del *EventSystem*



Opciones. En las escenas

2.1 Script llamando al Menú de Opciones

- En cada escena, crear un *Game Object* (*FuncionesGenerales*) que incluye el script *FuncionesGenerales.cs*
  a) Pulsando la tecla <esc> presenta el menú de opciones (carga la escena *Menu Opciones* de forma parcial

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

using UnityEngine.SceneManagement;

public class FuncionesGenerales : MonoBehaviour
{
    private bool siActivo;
    // Start is called before the first frame update
    void Start()
    {
        siActivo = false;  //  Inicialmente, el menú no e´stá activo
    }

    // Update is called once per frame
    void Update()
    {
        if (Input.GetKeyDown(KeyCode.Escape))
        {
            siActivo = !siActivo;  // Conmutamos el estado del menú

            if (siActivo) SceneManager.LoadScene("Menu Opciones", LoadSceneMode.Additive);  //  Cargamos la escena con el menú de forma aditiva a la actual
            else SceneManager.UnloadSceneAsync("Menu Opciones");  //  Descargamos la escena con el menú opciones

        }
    }
}

```