# Trabajo Práctico Unity

**Objetivos:**

* Crear un juego simple, aprendiendo a usar el motor de juegos Unity y explorando las posibilidades que nos brinda para crear nuestros juegos o aplicaciones.

**Prerrequisitos:**

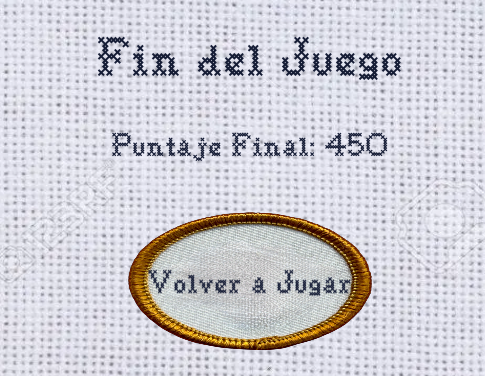
* Haber leído y analizado el material de estudio de Unity.
* Tener instalado tanto Unity Hub como la versión de editor de Unity 2021.3
* Tener instalado Visual Studio Community 2019.

**Consigna**

En esta actividad proyectual se pide realizar una aplicación para computadoras de escritorio, en principio con SO Windows, utilizando el motor de juegos Unity. Esta guía te especificará el paso a paso para realizar el proyecto. **Sin embargo, puedes cambiar ampliar y mejorar los diferentes aspectos involucrados en el proyecto.**

El mini-juego a implementar se llamará “Button Hero”, y el objetivo es atrapar los botones de costura que se irán generando en la pantalla, para ello usaremos distintas estampas que representan teclas en nuestro teclado. En caso de que algún botón atravesase toda la pantalla sin que lo atrapemos, perderemos una vida. Al agotar todas sus vidas, el jugador finalizará la partida. Cuando un botón pasa por encima de alguna estampa, el jugador debe presionar la tecla correspondiente a la estampa para “atrapar” ese botón y ganar puntos. La velocidad de los botones irá creciendo con el tiempo para ir aumentando la dificultad.

El proyecto puede realizarse de a 2 alumnos como máximo.



Los pasos a continuación son para orientar la realización de la práctica, puedes agregar, además de todo lo pedido, algunos elementos adicionales, de manera tal de lograr las características propias para la aplicación.

**Acciones a Realizar**

Lo primero que debes hacer es contar con los recursos visuales del proyecto, puedes usar los que te ofrecemos o buscar los propios.

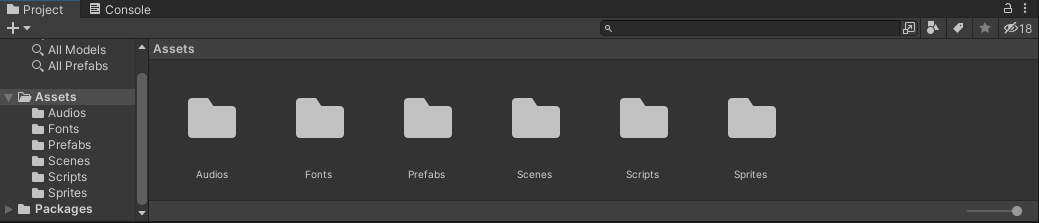
Necesitarás:

* Imagen de un Botón
* Imagen de un Círculo
* Imagen de Fondo
* Imagen de las letras ‘U’, ‘N’, ‘L’ y ‘P’.
* Fuente para los textos.
* Audios (Bien, Mal, Sonido de Fondo)

Una vez resueltos estos recursos, comenzaremos a trabajar en el entorno de desarrollo de Unity. Para ello, seguiremos los siguientes pasos (Explicados en la Clase III):

1. **Creación del Proyecto:** 
   1. Abrimos Unity Hub.
   2. Dentro de la pestaña “Proyectos”, hacemos click en “Nuevo Proyecto”.
   3. Seleccionamos la plantilla “2D”, le asignamos un nombre a nuestro proyecto y damos clic en “Crear Proyecto”.
   4. Una vez creado el proyecto, se nos abrirá nuestro editor de Unity.
2. **Estructura de archivos**
   1. Para poder tener nuestro directorio ordenado y que se nos facilite en un futuro encontrar los recursos, nos tomaremos unos minutos para organizar las carpetas de nuestro proyecto.
   2. Dentro del inspector de archivos de Unity crearemos las siguientes carpetas:
      1. Audios: Aquí guardaremos todos los audios que sonarán en nuestro juego.

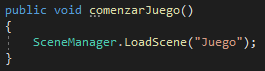
* Correcto.mp3
* Fallo.mp3
* GEazy x Bebe Rexha Me Myself I.mp3
* Mariah Carey All i Want For Christmast is You.mp3
  + 1. Fonts: Aquí guardaremos todas las fuentes que usarán los textos de nuestro juego.
* GoodbyeCrewelWorld.ttf
  + 1. Prefabs: Aquí guardaremos todos los prefabs que crearemos en este proyecto (Descritos en la sección 4).
    2. Scenes: Creada por defecto, aquí crearemos las escenas del proyecto.
    3. Scripts: Aquí guardaremos todos los scripts.
* Boton.cs
* Fin.cs
* GameManager.cs
* Generador.cs
* Menu.cs
* Pulsador.cs
  + 1. Sprites: Aquí guardaremos todas las imágenes que usaremos.
* Boton.png
* Circulo.png
* Fondo.png
* Letra L.png
* Letra N.png
* Letra P.png
* Letra U.png
  1. Luego, nuestro directorio debería de haber quedado de la siguiente manera:



1. **Diseño del menú:**
   1. Ahora trabajarás en crear la siguiente escena:
   2. Dentro de la carpeta Scenes, crearemos una nueva Escena, la cual llamaremos “Menú”.
   3. Dentro de la jerarquía de objetos, crearemos un “Canvas” para colocar todos los elementos del menú, el cual llamaremos “interfaz”.
   4. Dentro del canvas crearemos un “Text” el cual llamaremos “Título”.
      1. Dentro del inspector del Objeto, nos enfocarémos en el componente “Text”
      2. Arrastraremos la fuente “GoodbyeCrewelWorld.ttf”
      3. Aumentamos el tamaño de la fuente a 50
      4. Dentro del texto escribimos “Button Hero”
      5. Luego, usaremos la herramienta de “mover” para desplazar al logo en la esquina superior de la pantalla
      6. Por último, usaremos la herramienta de “escalar” para hacer al logo tan grande como deseemos.
   5. Luego, crearemos un “Image” el cual llamaremos “Fondo”.
      1. Usamos la herramienta de “escalar”  para ajustar su tamaño al del calvas.
      2. Ajustamos sus anclajes para que se agrande junto al canvas:



* + 1. Arrastramos la imagen “Fondo.png” a la imagen fuente de su componente “Image”.
  1. Luego, crearemos un “Button”
     1. Dentro de su componente “Image” arrastramos la imagen “Circulo.png” a la imagen fuente.
     2. Configuraremos su componente “Text” de la misma manera que hicimos con el título (Ver d.)
     3. Dentro del inspector, agregaremos un nuevo componente de tipo Script, al cual llamaremos “Menu”.
     4. Una vez creado este script, le haremos doble click para empezar a editarlo.
     5. Borramos el setup y el update y lo único que hacemos es declarar la función “comenzarJuego()”(Que debe ser pública) para cambiar a la escena del juego, a través del SceneManager:



* + 1. Dentro del inspector del Botón, arrastramos nuestro Script a la acción “OnClick()”
    2. Dentro del menú desplegable seleccionamos que se ejecute Menu -> comenzarJuego().
    3. Nuestro componente “Botón” debería haber quedado de la siguiente manera:



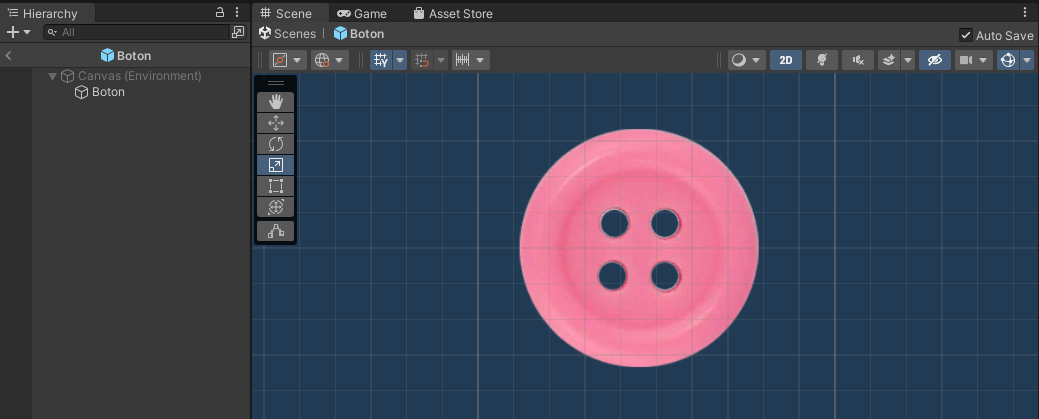
1. **Diseño de los Prefabs**

Antes de comenzar a diseñar el juego en sí, debemos identificar los grandes actores del mismo. En el caso de “Button Hero” van a existir dos grandes actores:

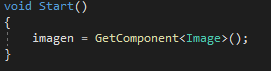
* Los botones: Se desplazan verticalmente por la pantalla y son los que debemos de atrapar.
* Los pulsadores: Que están en una posición estática en la pantalla y cada uno responde a una tecla distinta.

Cestos objetos los vamos a utilizar varias veces dentro de nuestro juego, nos conviene primero diseñarlos de manera **aislada** como **prefabs** y luego integrarlos a nuestra escena. A continuación se describe como diseñar cada uno:

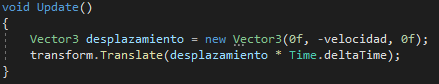
**Prefab Botón:**

****

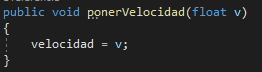
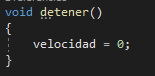
1. Dentro de la carpeta “Prefabs” creamos un nuevo Prefab al cual llamaremos “Boton”.
2. Dentro del prefab, creamos una “Image” la cual llamaremos “Boton”.
3. Dentro de su componente “Image” arrastramos la imagen “Boton.png” a la imagen fuente.
4. A nuestro GameObject “Boton” le agregamos un componente “Circle Collider 2D” para que pueda interactuar con los pulsadores, y ajustamos su radio a 41.
5. Luego le agregamos un componente “RigidBody 2D” para que pueda interactuar con las colisiones. Pondremos su “Body Type” a “Kinematic” para que su movimiento sea gobernado totalmente por nosotros y activamos el Checkbox de “Use Full Kinematic Contacts” para que detecte colisiones con otros colisionadores sin RigidBody (Nuestros pulsadores).
6. Por último le agregamos un Script que describa su funcionamiento el cual llamaremos “Boton”.
   1. Dentro declararemos dos variables, una para la velocidad (float) y otra para tener una referencia de su imagen (Image)
   2. Dentro del start obtendremos la referencia a su componente Image

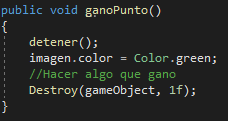
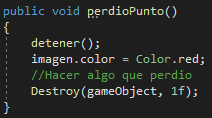


* 1. Dentro del Update hacemos que se desplace verticalmente hacia abajo, tomando en cuenta el tiempo que pasa entre frames:

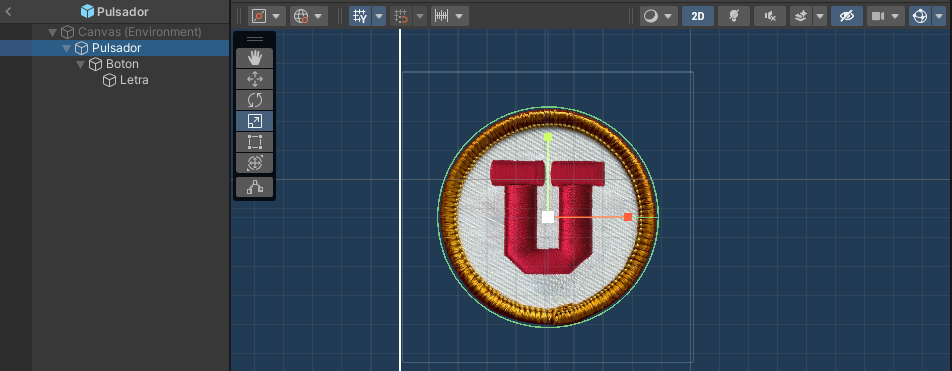


* 1. Luego declararemos algunas funciones para indicar que se ganó o perdió un punto, deteniendo el movimiento del botón y cambiando su color:

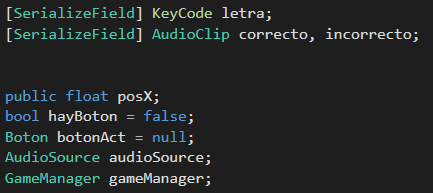




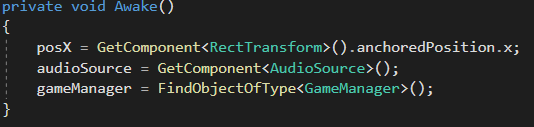
**Prefab Pulsador:**

****

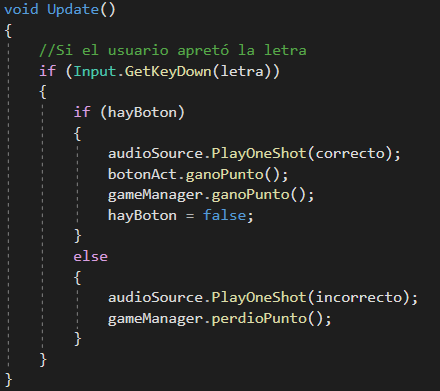
1. Dentro de la carpeta “Prefabs” creamos un nuevo Prefab al cual llamaremos “Pulsador”.
2. Dentro del prefab, creamos una “Image” la cual llamaremos “Boton”.
3. Dentro del inspector, le asignamos a nuestro Boton el tag “Boton” para que se diferencie de otros objetos.
4. Dentro de su componente “Image” arrastramos la imagen “Circulo.png” a la imagen fuente.
5. Dentro de “Boton” creamos otro GameObject “Image” llamado “Letra”
6. Dentro de su componente “Image” arrastraremos la imagen de alguna letra a su imagen fuente.
7. A nuestro GameObject “Boton” le agregamos un componente “Circle Collider 2D” para que pueda interactuar con los pulsadores, y ajustamos su radio a 38.
8. También le agregamos un componente “Audio Source” para que pueda reproducir archivos de audio.
9. Por último le agregamos un Script que describa su funcionamiento el cual llamaremos “Pulsador”.
   1. Dentro declararemos dos variables, una la tecla (KeyCode), para el audio correcto y audio incorrecto (AudioClip), una para su posición en X (float), una para ver si tiene un boton encima (boolean), una referencia al Boton que tenga encima (Boton), una referencia a su Audio Manager (AudioManager) y una referencia al GameManager (Script encargado del juego en general):



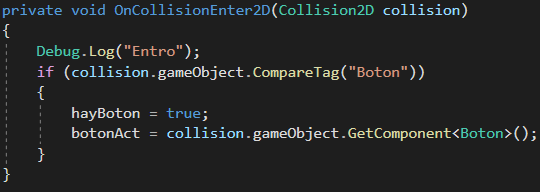
* 1. Dentro del Awake(Para que se ejecute antes que los Start) obtendremos las referencias a componentes



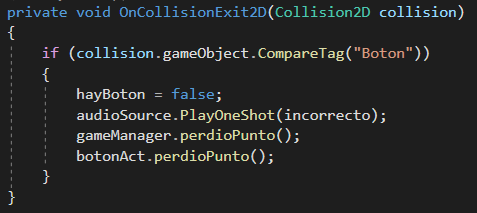
* 1. Dentro del Update nos fijamos si se apretó la tecla, en tal caso nos fijamos si tenemos algun boton encima para decidir si gana puntos o si pierde vidas :



* 1. Luego, usando el sistema de colisiones de Unity nos fijamos si entramos en contacto con algún botón y en tal caso nos traemos su instancia para usarla dentro del Script:



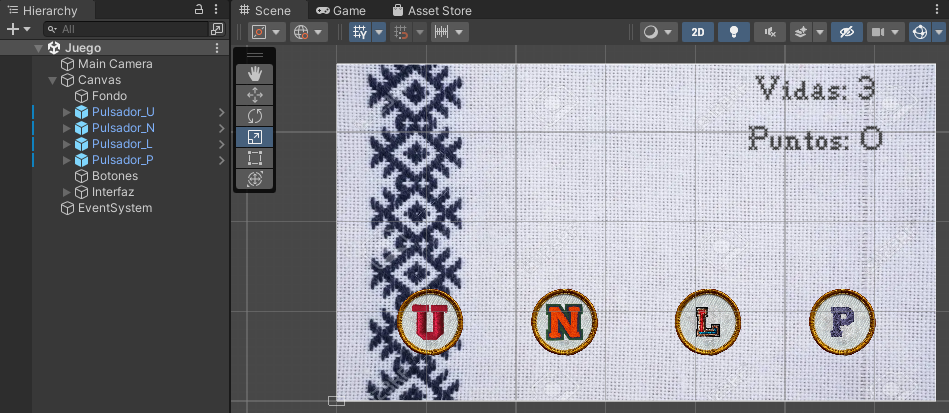
* 1. Hacemos lo mismo para cuando se nos pasa de largo algún botón:



* 1. Por último asignamos las referencias a los audios a través del inspector, arrastrando cada audio desde la carpeta “Audios” a su variable correspondiente:



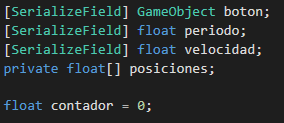
1. **Diseño del juego**
2. Creamos una nueva Escena dentro de nuestra carpeta “Scenes” llamada “Juego”



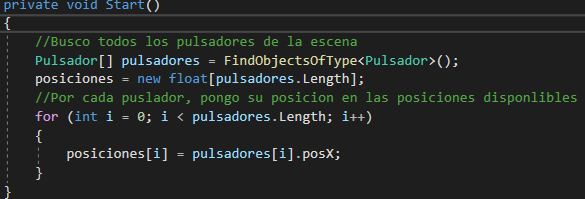
1. Dentro creamos un Canvas con el mismo fondo del Paso 3.
2. Arrastramos desde la carpeta “Prefab”, 4 “Pulsador” hacia nuestra escena. Luego les cambiaremos el nombre para reconocernos.
3. Por cada pulsador, cambiamos la imagen de su letra por la letra que lo identifica y cambiamos la tecla del Script “Pulsador” por la tecla que lo identifica.



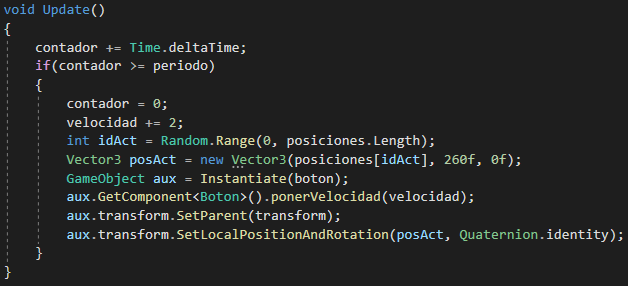
1. Luego posicionamos cada pulsador de manera que queden distribuidos en una fila.
2. Luego creamos un GameObject vacío llamado “Botones” que se encargará de ir generando los botones
3. A este GameObject le agregaremos un Script llamado “Generador”.
4. Dentro de este script declararemos variables para acceder al prefab Boton(GameObject), el periodo de generación(float), la velocidad de los botones(float), las posiciones de los pulsadores(float[]) y un contador para acumular el tiempo pasado(float).



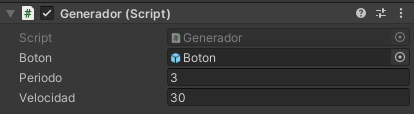
1. En el start me guardo las posiciones de los pulsadores para saber donde puedo generar los botones:



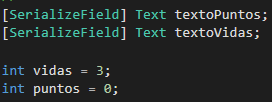
1. Dentro del update cuento el tiempo que paso y si se cumplió un periodo, se genera un Botón en una posición aleatoria dentro de las posiciones de los pulsadores, y se aumenta la velocidad.



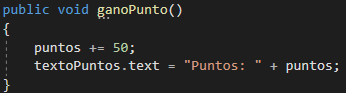
1. Luego se asignan las variables desde el inspector, arrastrando el prefab “Boton” a la variable del mismo nombre.

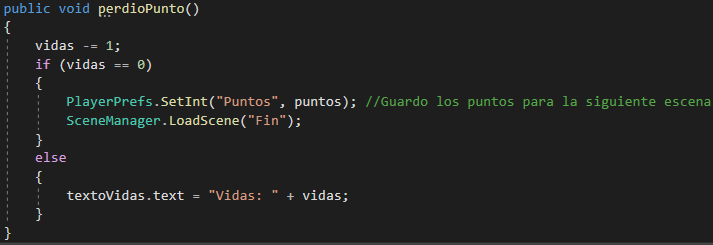


1. Por último creamos los textos que nos informan del progreso del juego. Para eso creamos un GameObject vacío dentro del Canvas llamado “interfaz”.
2. Dentro de “Interfaz” creamos dos Textos, uno para los puntos y otro para las vidas.
3. Configuramos estos textos igual que como hicimos en 3.d. pero esta vez con una fuente de 35pts.
4. Dentro de “interfaz” creamos un Script llamado “GameManager”.
5. Dentro del script tendremos dos variables para ir modificando los textos: textoPuntos y textoVidas (Text). Y dos variables internas que lleven la cuenta de estos valores (int):



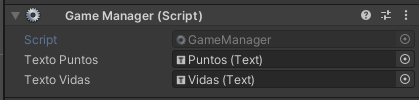
1. Y declararemos las funciones encargadas de modificar estos valores (funciones utilizadas por los pulsadores previamente)



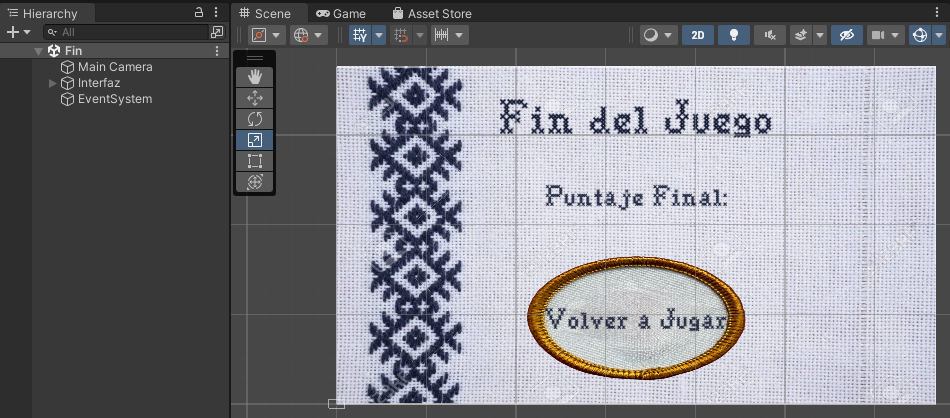


Fijémonos que si perdió un punto y se queda sin vidas, nos transportamos a la escena “Fin” guardando previamente los puntos acumulados.

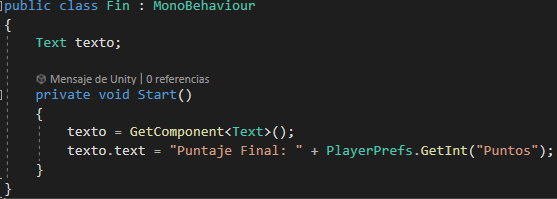
1. Por último arrastramos los textos a las variables en el inspector:



1. **Diseño pantalla de créditos**
2. Crearemos una ultima escena dentro de la carpeta “Scenes” llamada “Fin”



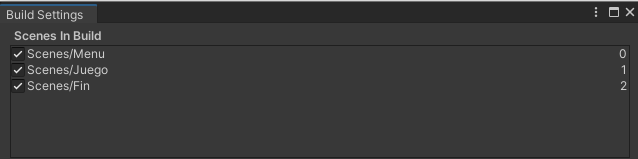
1. En esta nueva escena haremos una copia de la escena creada en el punto 3 cambiando los textos del Titulo por “Fin del Juego” y del boton por “Volver a Jugar”
2. Además, agregaremos otro texto (Copiando el titulo) que nos mostrará nuestro desempeño. Para ello a este nuevo texto le agregaremos un Script llamado “Fin”. Lo único que hará este script es modificar el texto con el puntaje obtenido la ultima jugada:



1. **Exportar**

Una vez que nuestro proyecto está finalizado, podremos exportarlo para distintas plataformas, nos enfocaremos en exportarlo a Windows:

1. En el menú de Unity seleccionamos File-> Build Settings
2. En la pestaña “Scenes in Build” arrastramos todas las escenas que creamos, asegurándonos que haya quedado el Menú como primer escena:



1. Luego hacemos click en “Build”, nos preguntará en que carpeta queremos guardarlo. Creamos una nueva carpeta llamada “Export” y le damos a “aceptar”.
2. Una vez generado, ya podemos ejecutar nuestro primer juego y compartirlo.

**Fin del Proyecto**

El proyecto debe entregarse mediante la mensajería de IDEAS, con un .zip con nombre y apellido del alumno. El .zip debe incluir la carpeta donde se desarrolló el proyecto, la carpeta del juego exportado para Windows, y un video donde cada alumno participante en el grupo explique una parte de cómo se realizó el proyecto, contando cómo se logra la funcionalidad.