

The background is a dark navy blue. In the top-left corner, there are two overlapping geometric shapes: a blue parallelogram and a light green parallelogram. In the bottom-left corner, there is a circular inset showing a detailed, grayscale image of a printed circuit board (PCB) with various electronic components. In the top-right corner, there is a faint, grayscale image of a complex, layered circuit board structure.

HLC

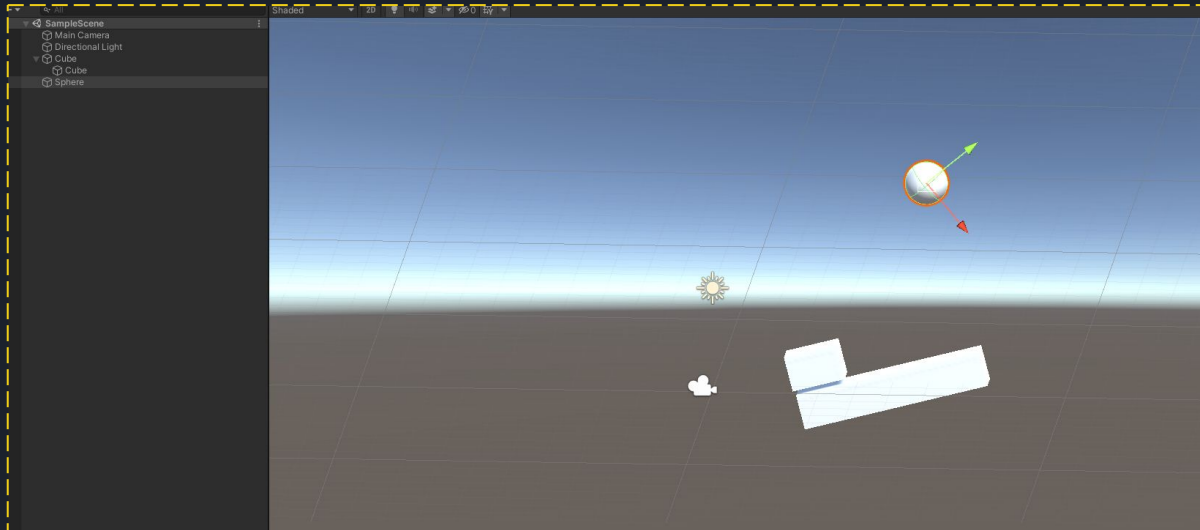
2° DAM/W

Movimiento en escenas




Con los botones de Play, Pause y Step podemos poner en marcha nuestro juego.

- Play: Ejecuta o detiene el juego
- Pause: Detiene momentaneamente el juego. Si lo activamos antes de pulsar Play, el juego comenzará pausado.
- Step: Avanza el juego frame a frame

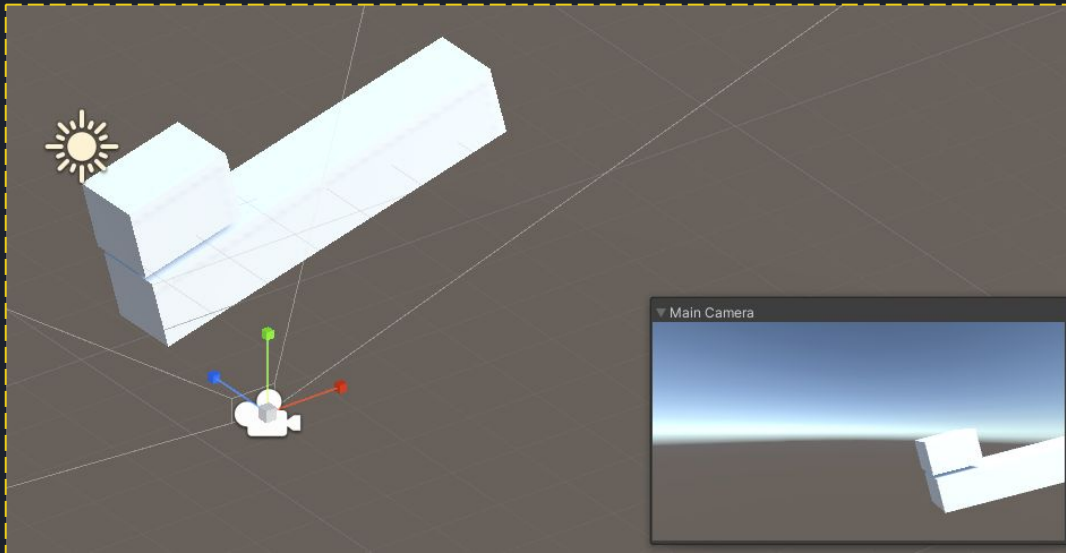


Probemos a lanzar una esfera gravitacionalmente a otro objeto usando:
Add component: Rigidbody

Creación de nueva cámara

Cuando damos a Play o accedemos a la vista Game,  , podremos elegir desde dónde ver el juego. Es decir, verlo como si fuese la grabación de una cámara real en el plano.

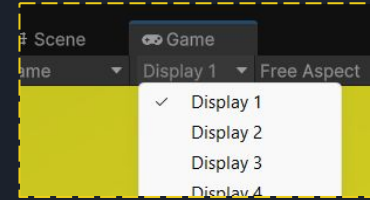
Para ver la previsualización de lo que “está viendo” una de nuestras cámaras, simplemente tendremos que seleccionarla en el panel Hierarchy. Si no aparece: click derecho en Scene > **Overly menu** > **Cameras**



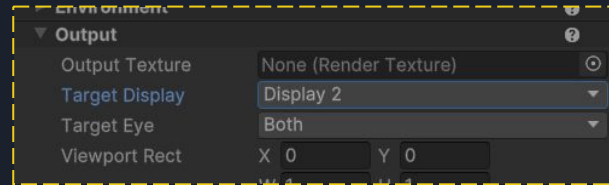
Añadamos otra cámara desde el panel de Hierarchy tal y como añadimos un objeto 3d

Creación de nueva cámara

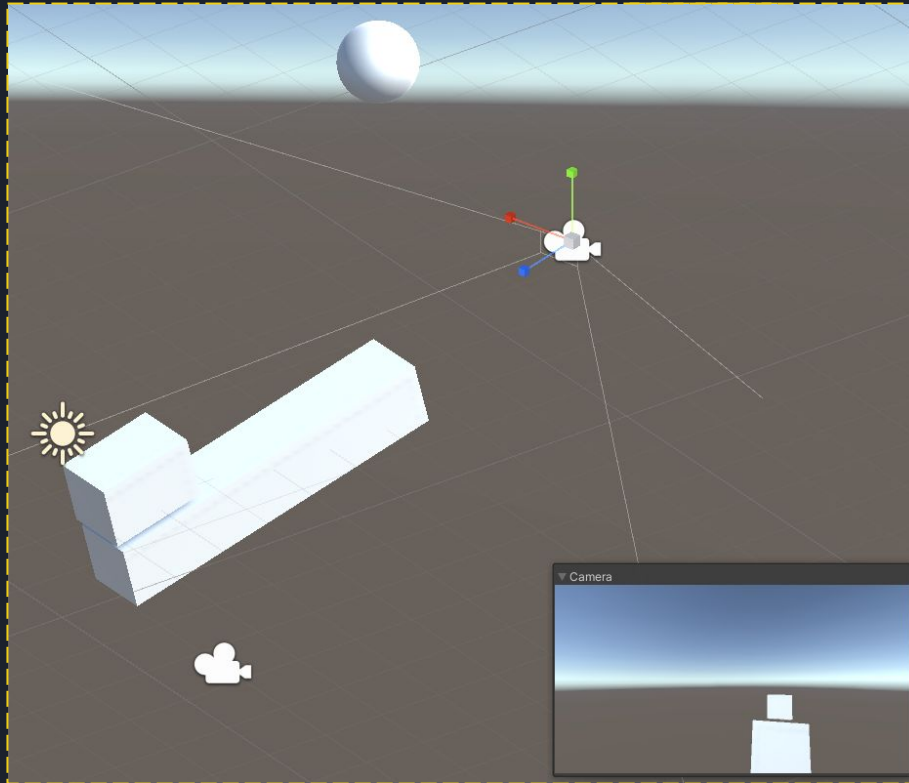
Para ver una cámara u otra desde la vista Game, cambiamos en Display:



Desde el panel Inspector de cada cámara, en Output, seleccionamos en Target Display la que queramos.



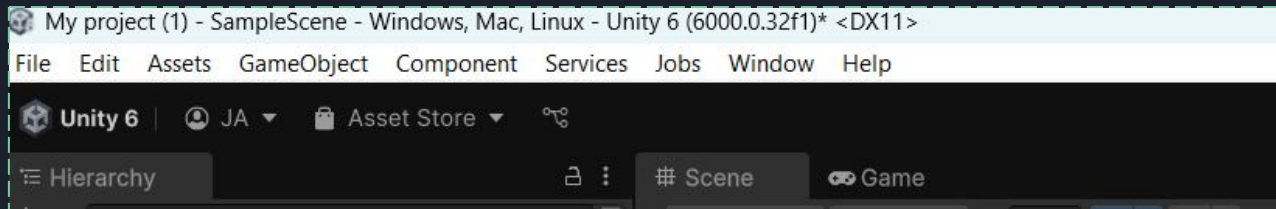
Creación de nueva cámara



Una cámara la podemos mover por el plano del mismo modo que lo hacemos con el resto de objetos.

Para alinear la cámara con la vista de escena, lo cual nos puede facilitar un poco dicha tarea, podemos hacer `GameObject > Align With View`.

Vista de proyecto

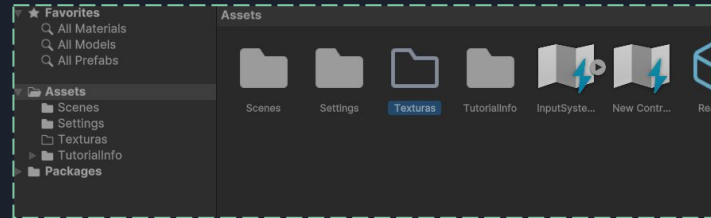
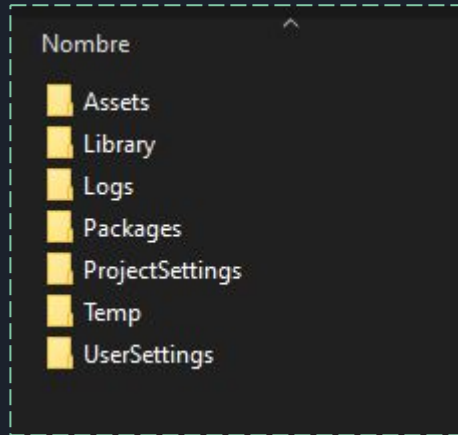


Los elementos que podemos encontrar en el menú Project es:

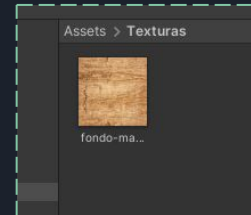
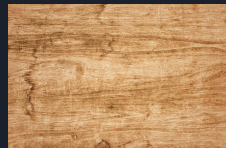
- **File:** Gestión de proyectos y escenas. Destaca la opción de Build Profiles
- **Edit:** Clásicas opciones de copiar, pegar, duplicar y eliminar. Además incluye también las opciones de proyecto y generales de Unity.
- **Assets:** Representa un directorio real de nuestro explorador de archivos del sistema operativo.
- **GameObject:** Por una parte contiene las mismas opciones que el menú Create de la vista de jerarquía y por otro ofrece distintos modos de modificar los objetos de la escena cambiando su alineación, padre, etc.
- **Component:** Mismo menú que encontraremos en el botón de Add Component cuando trabajemos con objetos en el inspector.
- **Window:** Aquí encontramos los distintos paneles y ventanas de Unity clasificados en distintas categorías
- **Help:** Enlaces a la documentación, ayuda y reporte de errores.

Vista de proyecto

En cuanto a la relación de los componentes Asset con los ficheros reales en nuestro sistema, simplemente tendremos que hacer click derecho en Assets > Show in explorer



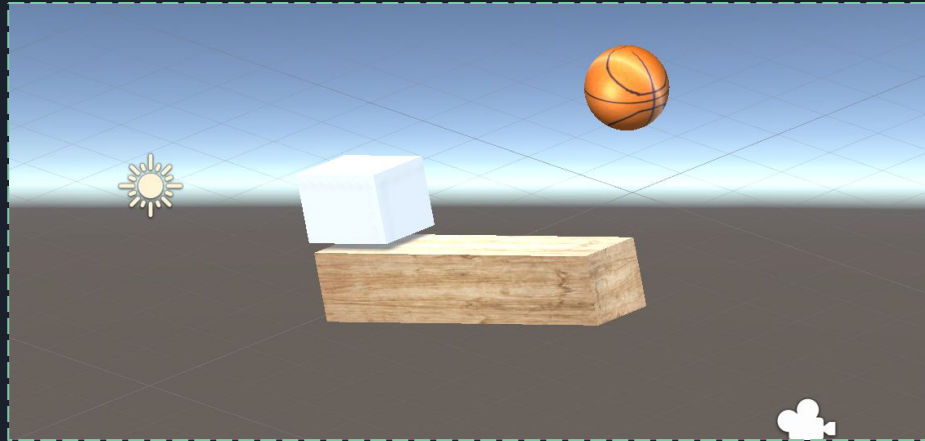
Podremos crear carpetas para organizar nuestros componentes. Por ejemplo, en la carpeta de texturas podemos añadir una imagen que nos sirva de textura:





Vista de proyecto

Si arrastramos la textura a uno de los objetos de nuestra escena, se aplicará.





Vista de proyecto

Ciclo de vida de Unity

Ciclo de vida de Unity:

Unity llama automáticamente a ciertos métodos de un script que hereda de `MonoBehaviour`. Algunos de los más comunes son:

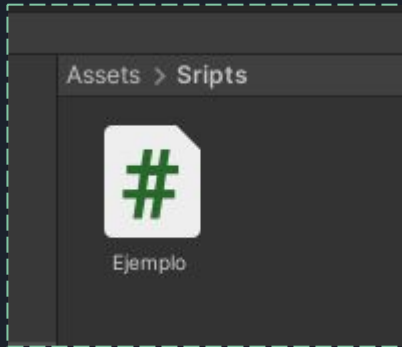
- `Awake()`: Se ejecuta cuando el objeto que contiene el script se inicializa.
- `Start()`: Se ejecuta una vez antes del primer frame, ideal para configuraciones iniciales.
- `Update()`: Se llama una vez por frame, útil para lógica que depende del tiempo.
- `FixedUpdate()`: Se llama en intervalos fijos, usado para física.
- `OnDestroy()`: Se ejecuta cuando el objeto es destruido.

Hola Mundo en la consola

Desde la consola podemos mostrar los mensajes de información, advertencia y error tanto de nuestro código como de Unity.

Los script los podremos crear o modificar mediante el IDE **VS Code** en el lenguaje **C#**.

Crearemos una carpeta llamada Scripts dentro de Assets. Y dentro de este, un script C#.



Este archivo lo abrimos en Visual Studio Code, o en algún IDE similar.

```
1 using UnityEngine;
2
3 public class Ejemplo : MonoBehaviour
4 {
5     void Start(){
6         Debug.Log("Mensaje de información");
7         Debug.LogWarning("Mensaje de advertencia");
8         Debug.LogError("Error");
9     }
10 }
```

Hola Mundo en la consola

Dicho script lo podremos arrastrar a uno de nuestros objetos. Tras eso, nos vamos a la vista de Consola y damos a Play.

