



# Gráficos y Multimedia

## Sesión 3: Introducción a Unity



# Puntos a tratar

- Entorno Unity
- Assets
- Escena
- Objetos y Componentes
- Prefabs
- Scripts
- Paquetes de Assets
- Interfaz de Usuario



# Motor Unity



<http://unity3d.com>

Motor genérico para la creación de videojuegos

- Enfocado hacia el desarrollo casual

Permite un desarrollo rápido de videojuegos

- Se puede crear un videojuego sin escribir código
- Cuenta con su propia herramienta de edición integrada

Dos tipos de licencia: *Personal* (gratuita), *Plus* (35\$ al mes) y *Pro* (125\$ al mes)

Soporta gran cantidad de plataformas



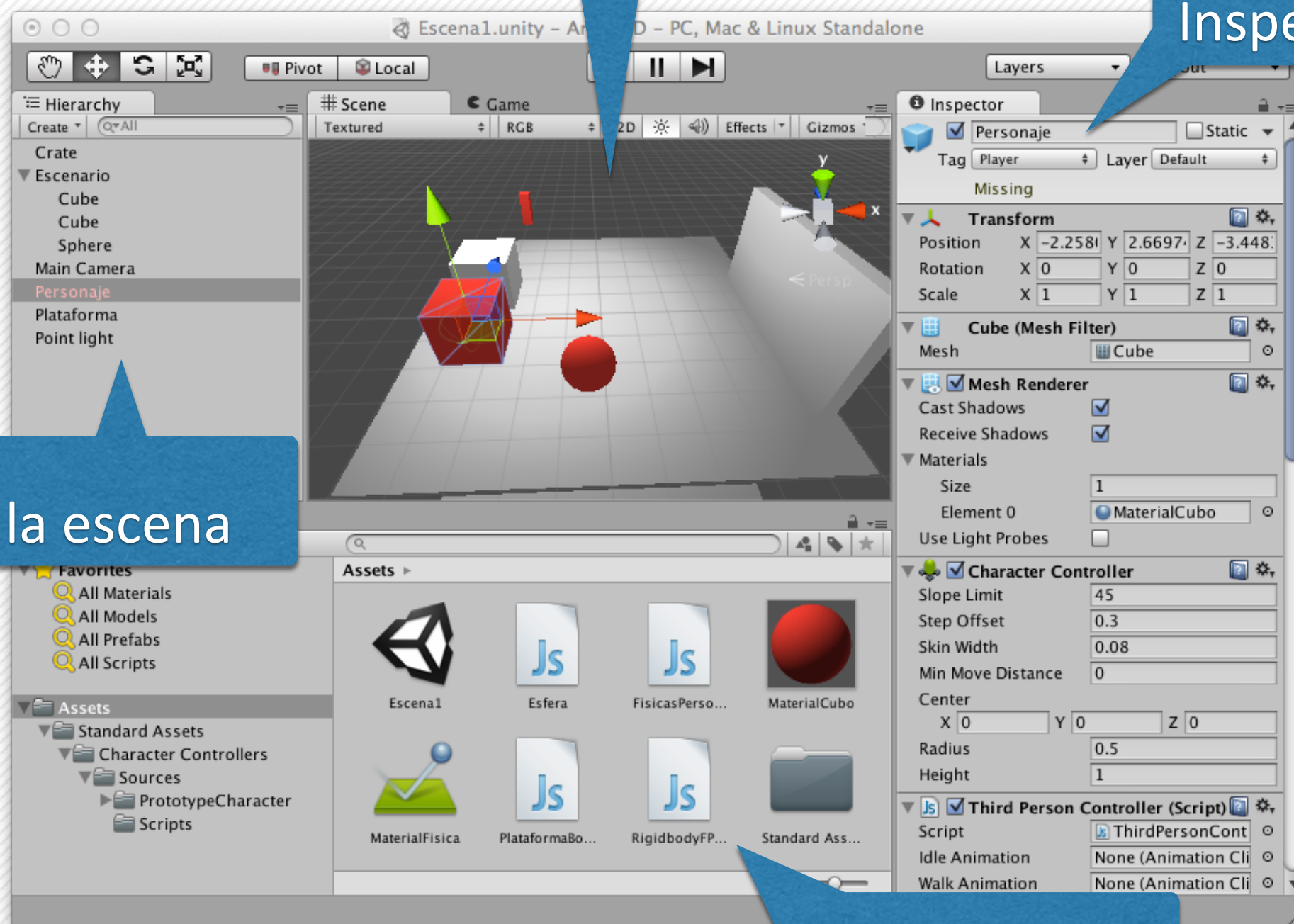


## El Editor

Escena / Juego

Inspector

Árbol de la escena



Assets





## Assets

Dentro del proyecto tenemos una serie de *Assets* (recursos)

Algunos tipos de *Assets* utilizados son

- **Escenas del juego:** Es el tipo principal de *asset*. Cada nivel del juego se guarda en un *asset* de tipo escena.
- **Mayas:** Geometría 3D que podemos añadir a la escena. Podemos importar modelos con animaciones (FBX, OBJ, 3DS, etc).
- **Materiales:** Definen el aspecto que le podemos dar a una maya (textura, color, sombreado, etc.)
- **Scripts:** Permiten personalizar el comportamiento de los objetos del juego mediante programación.
- **Clips de audio:** Músicas y efectos de sonido.

Podemos importar paquetes de *Assets* predefinidos



## Escena

Cada nivel del juego se almacena en un *asset* de tipo escena

La **escena** contiene un **árbol de objetos** (*Game Objects*)

Unity sigue una **arquitectura basada en componentes**

- Dentro de la escena tenemos únicamente *Game Objects*
- Los objetos no se definen por su tipo (todos son *Game Objects*)
- Lo que define a los objetos son los **componentes** que contienen
- Podemos añadir varios componentes a cada *Game Object*
- Los componentes definen la forma de actuar del objeto
- Ejemplos:

Una cámara es un *Game Object* con un componente *Camera*

Un *Game Object* con un componente *Light* se comportará como una fuente de luz



## Añadir objetos

Podemos crear distintos tipos de *Game Objects* predefinidos en el menú *GameObject*

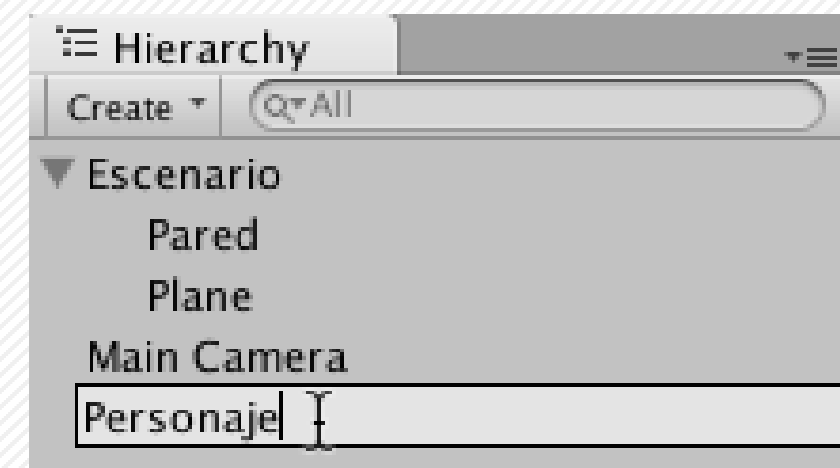
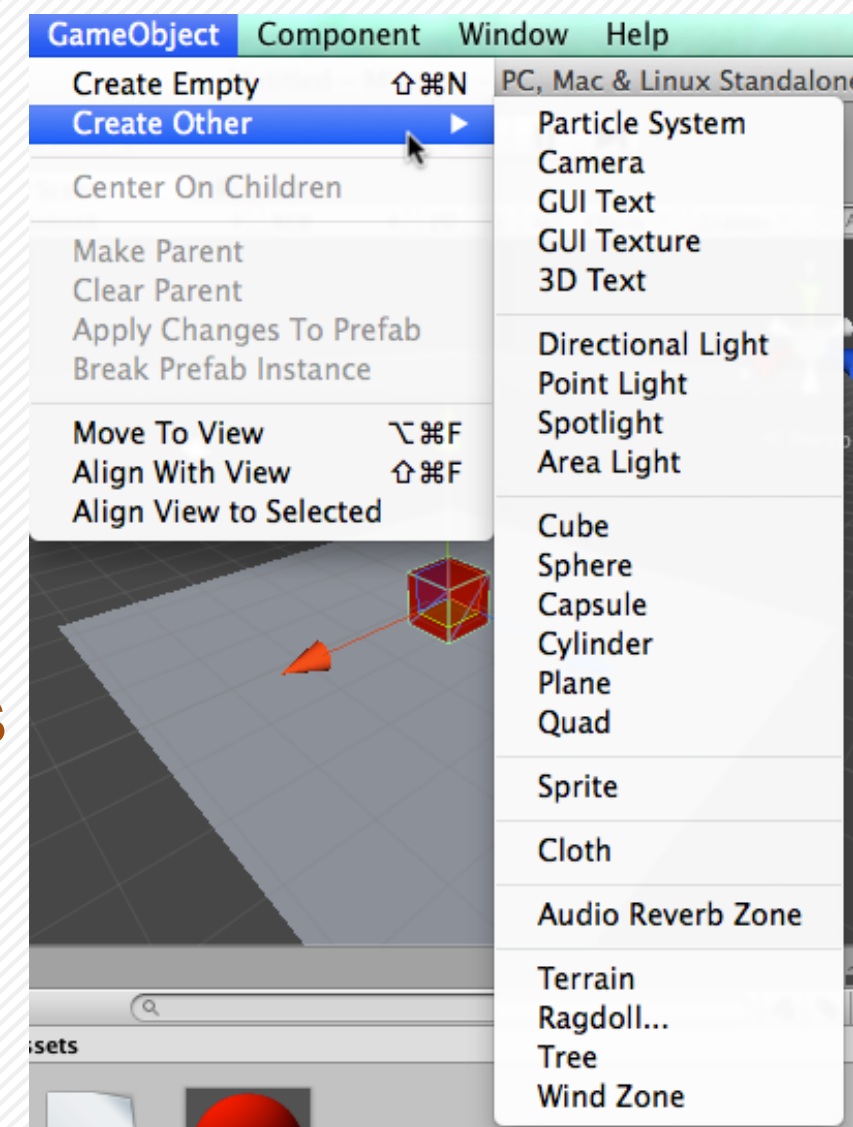
- Llevan ya una serie de componentes

Si creamos un *Empty Game Object* sólo tendremos un componente *Transform*

- Útil para agrupar objetos

Podemos agrupar objetos bajo un mismo nodo arrastrando y soltando

Podemos renombrar *Game Objects* en la vista *Hierarchy* (click sobre el nombre)

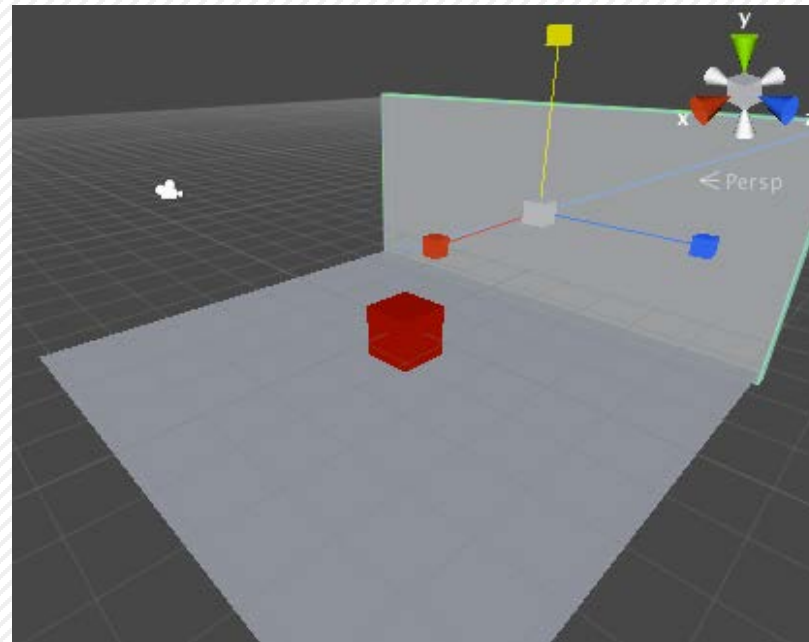




## Formas básicas

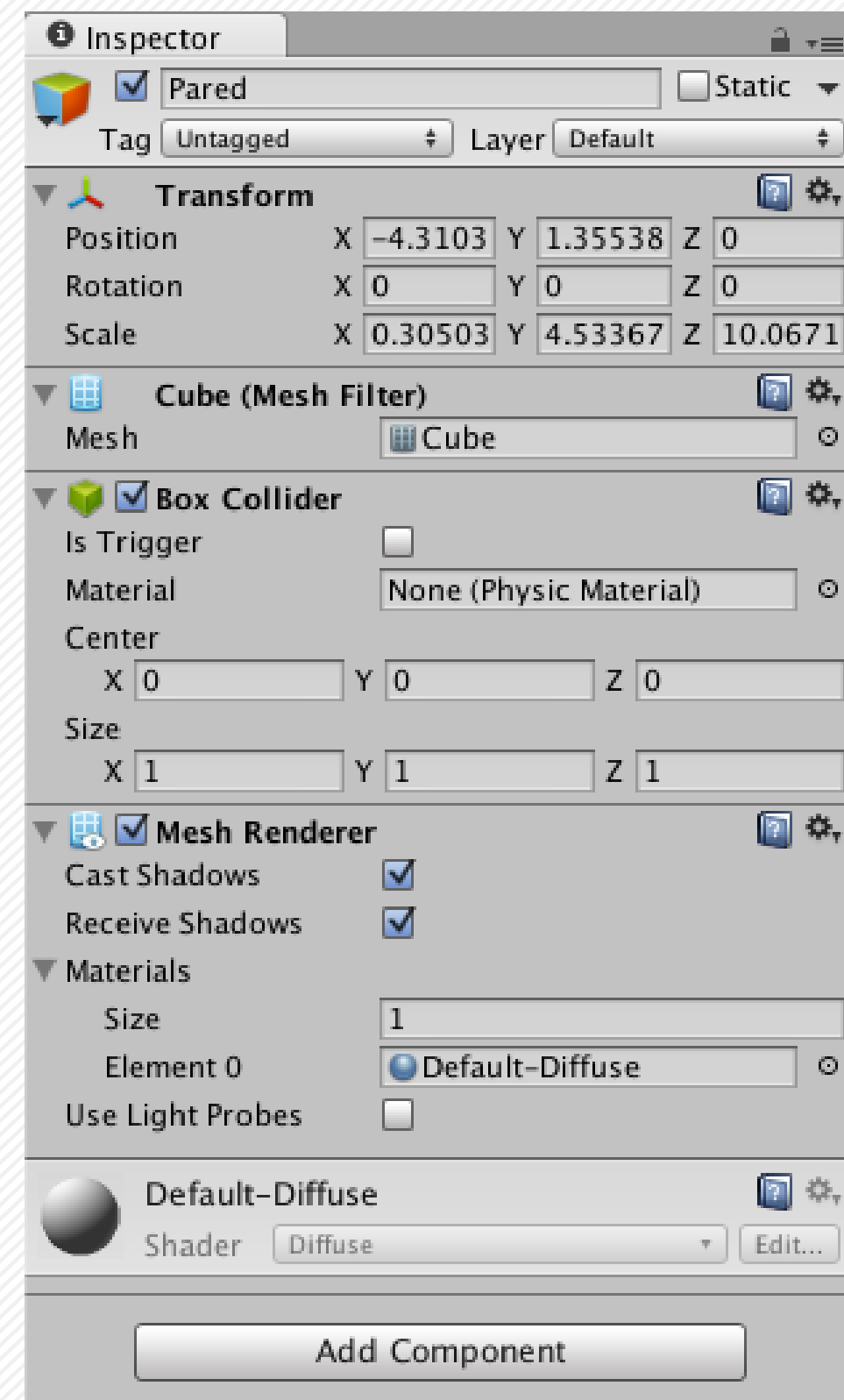
Podemos crear nodos con **formas** geométricas básicas

- Planos
- Cubos
- Esferas
- Cilindros



Estos objetos incorporan una serie de **componentes**

- *Transform* (posición y orientación)
- *Renderer* (material)
- *Mesh* (malla geométrica)
- *Collider* (geometría de colisión)







## Cámara

### Objeto con componente *Camera*

- Permite configurar los parámetros de la cámara
- Podemos definir varias cámaras en la escena
- Podemos añadir un componente *Camera* a nuestro personaje (FPS)

### Al seleccionar una cámara vemos el *preview* Componentes comunes en la cámara

#### GUI Layer

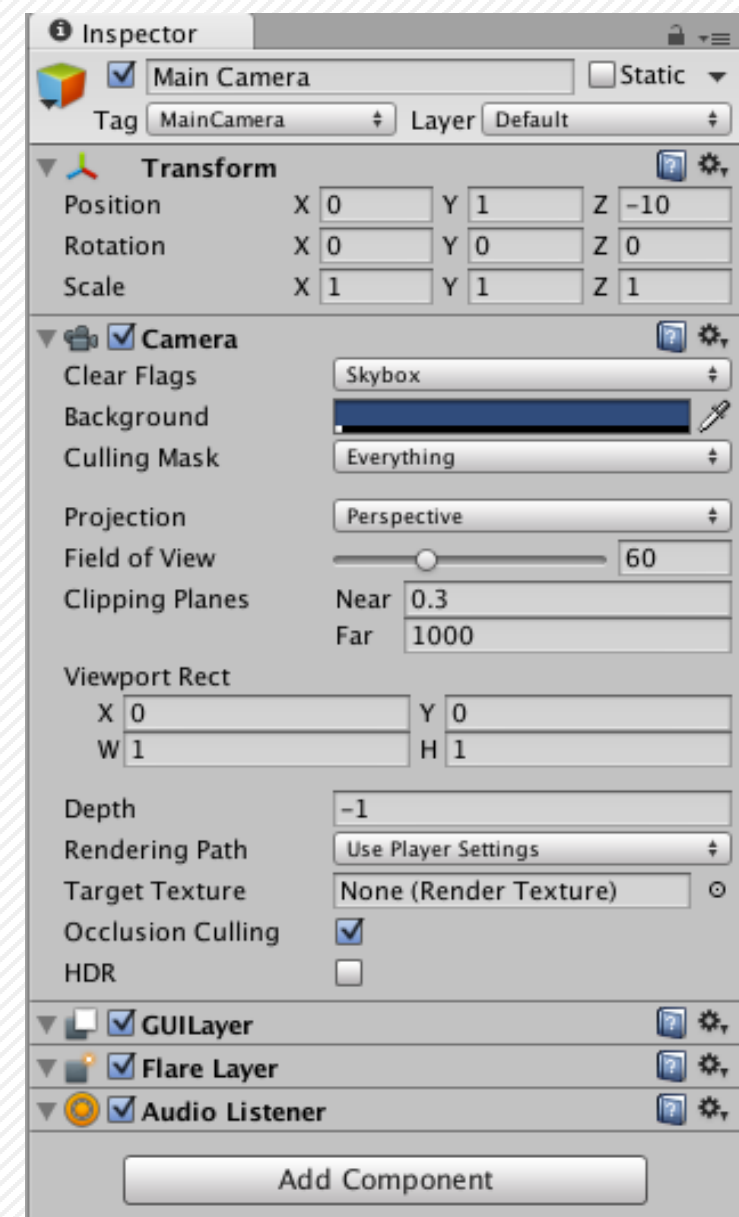
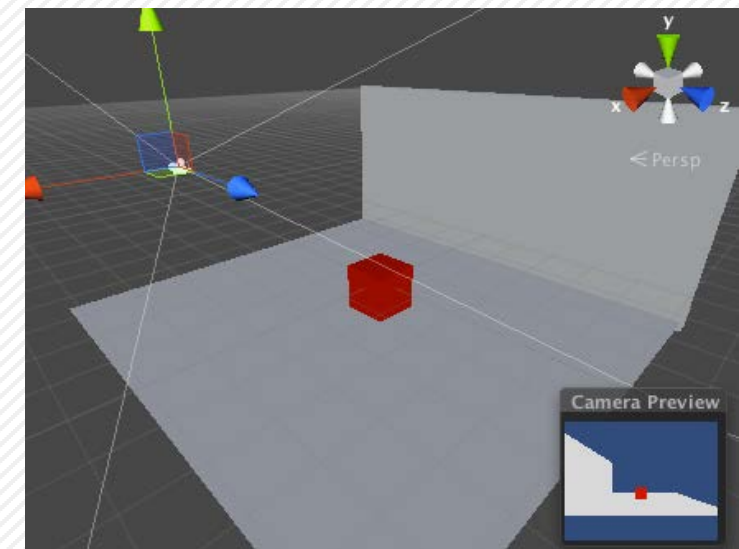
La interfaz se dibuja sobre la vista de la cámara

#### Flare Layer

Se muestra un destello sobre la imagen

#### Audio Listener

Escucha las fuentes de audio de la escena y reproduce el sonido a través de los altavoces, sólo debe haber uno en la escena





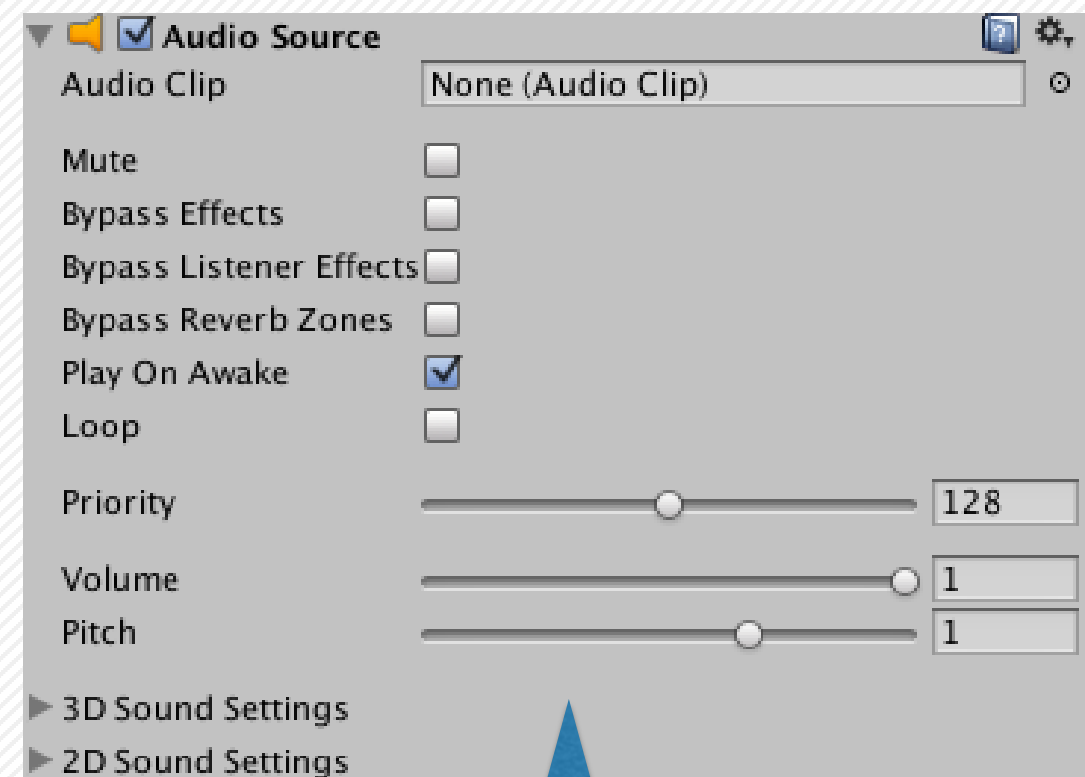
## Audio

Para reproducir audio necesitaremos un *Game Object* con un componente **Audio Source**

- Podemos añadirlo a la **cámara** para oírlo siempre igual
- Podemos añadirlo a **objetos del escenario** para que el volumen cambie según la distancia

Añadiremos el **clip de audio** a reproducir como *asset* del proyecto

- Especificaremos dicho *Asset* en el componente *Audio Source*



Las fuentes de audio serán “escuchadas” desde el Audio Listener de la escena. Esto será lo que oiremos a través de los altavoces

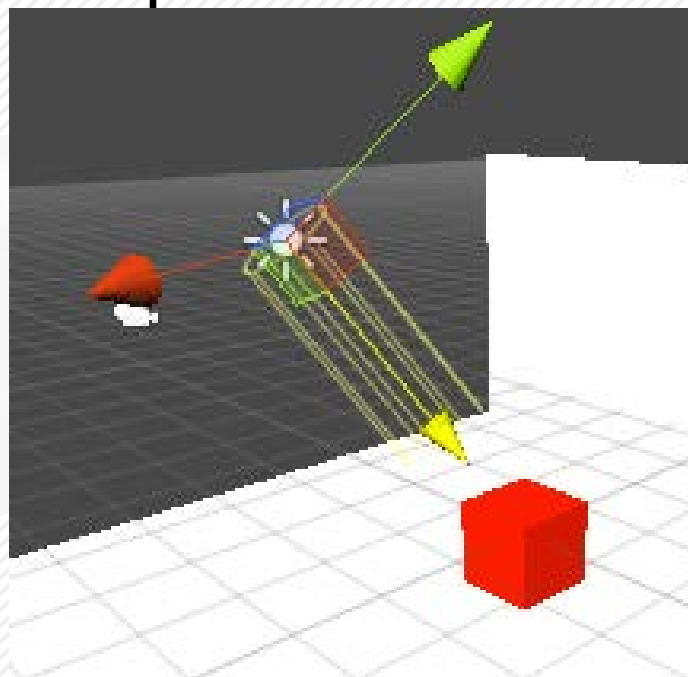


## Luces

### Objeto con componente *Light*

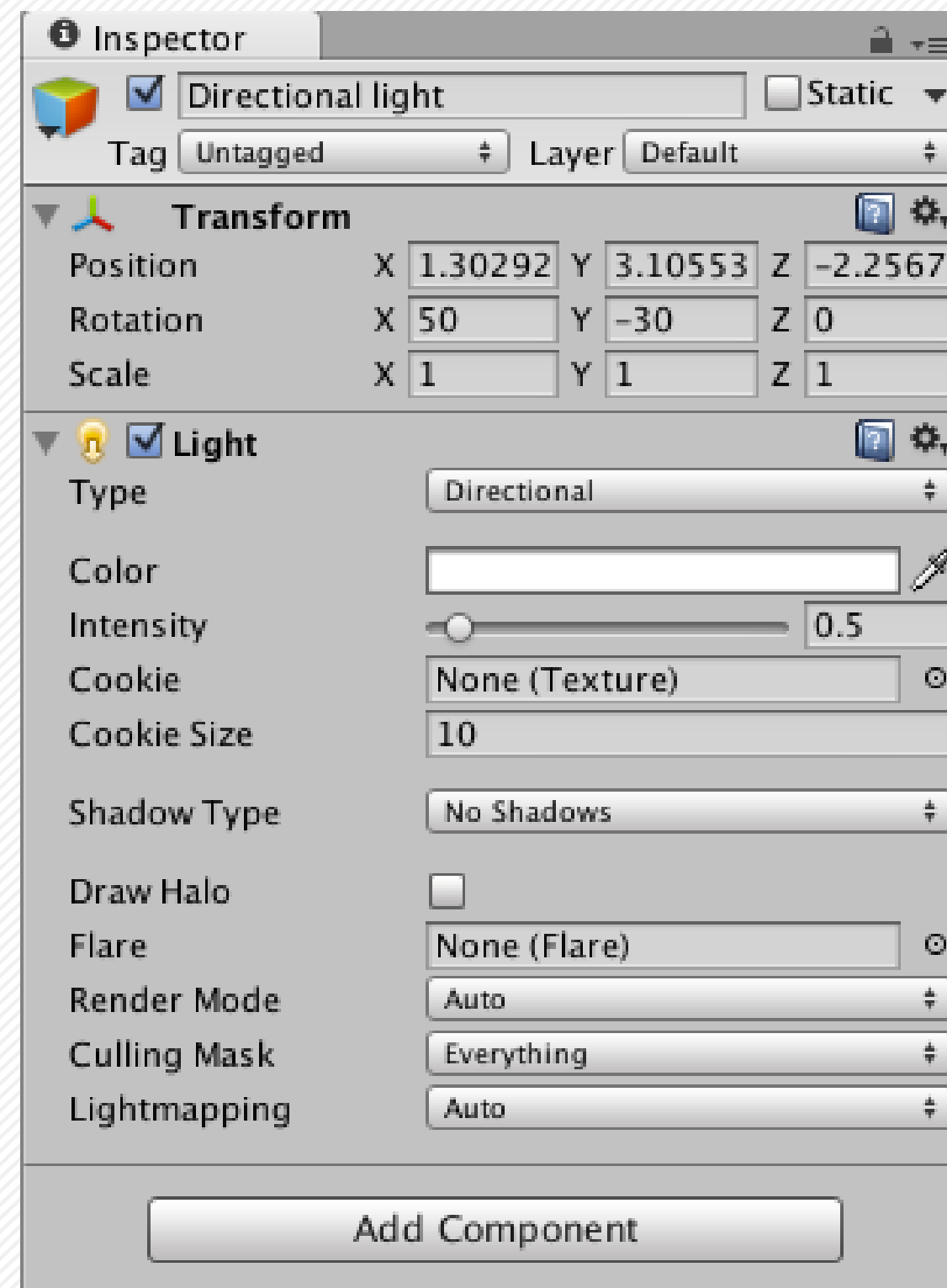
- Encontramos distintos tipos de luces predefinidas

Directional Light  
Point Light  
Spotlight  
Area Light



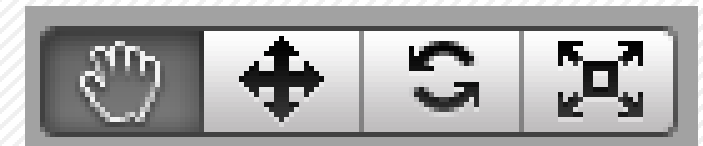
### Podemos configurar diferentes parámetros

- *Color*
- *Intensidad*
- *Distancia*




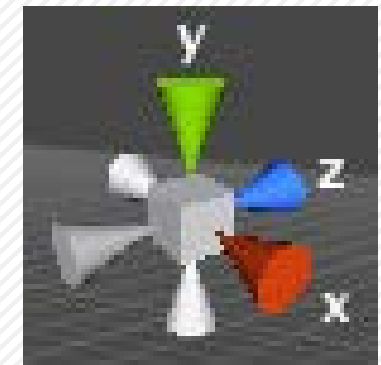


# Navegar por la escena



Podemos utilizar diferentes combinaciones de teclado para navegar por la escena 3D en el editor

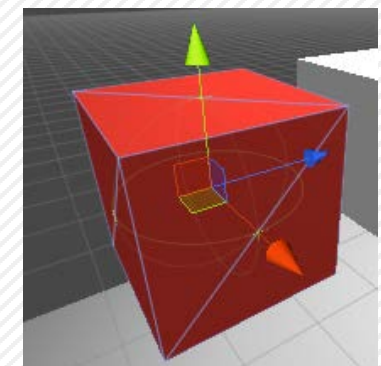
- Seleccionamos el icono 
- **Desplazamiento lateral:** *Click* y arrastrar
- **Zoom:** *Ctrl* + *Click* y arrastrar / Rueda del ratón
- **Rotación:** *Alt* + *Click* y arrastrar. Podemos usar el *gizmo*



## Iconos



- **Seleccionar un objeto:** *Click* sobre el objeto
  - Podemos ver sus propiedades en el inspector
- **Mover, rotar o escalar el objeto:** *Click* sobre el objeto y arrastrar
  - Podemos aplicar la transformación sólo sobre un eje







## Prefabs

Los *prefabs* son un tipo de *Asset*

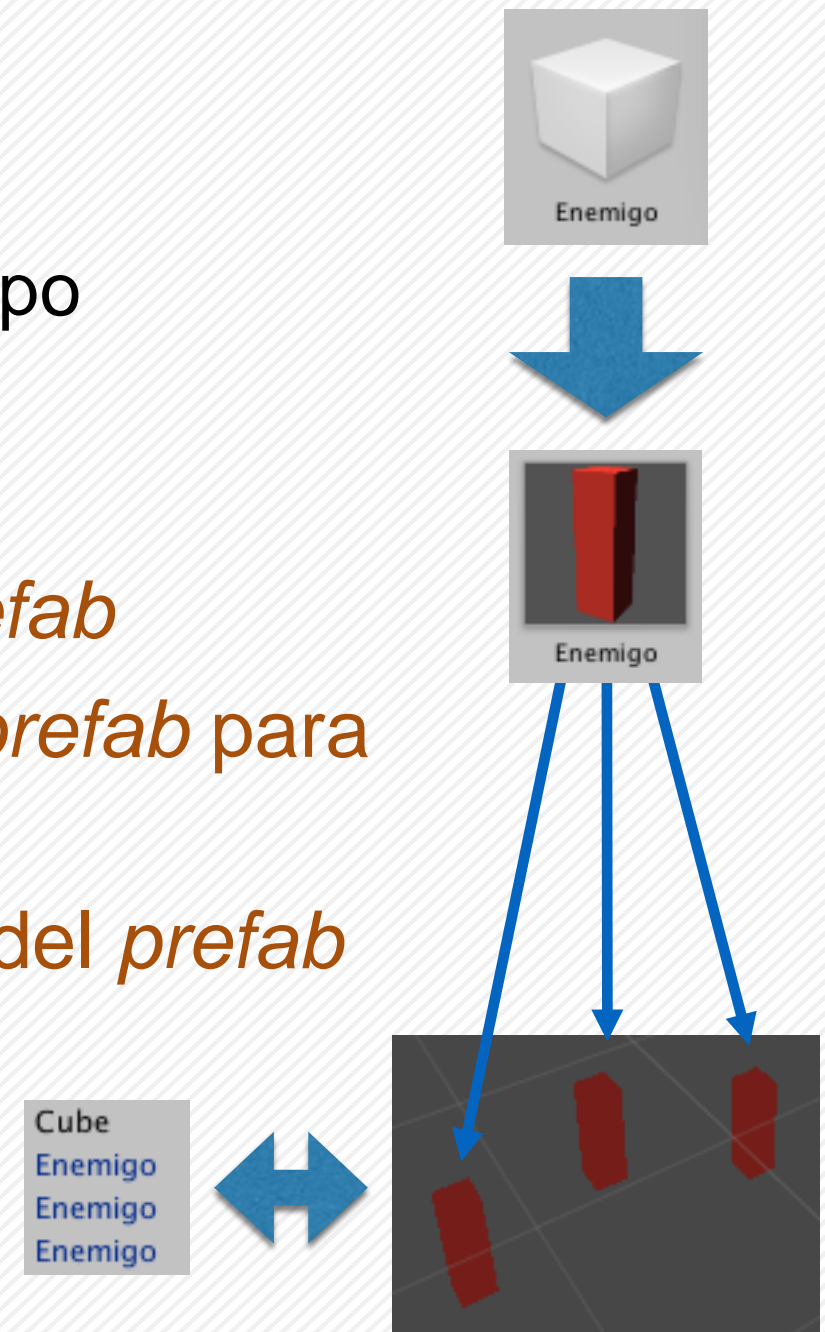
- Definen un prototipo de *game object*
- Podemos crear varios *game objects* de dicho tipo arrastrando el *prefab* sobre la escena

Crearemos un *prefab* con *Assets > Create > Prefab*

Arrastramos un *game object* existente sobre el *prefab* para utilizarlo como prototipo

Todos los objetos de la escena creados a partir del *prefab* aparecen en azul

- Indica que están vinculados a él
- Si cambiamos el *prefab*, todos cambian





# Scripts

## Los *scripts* son un tipo de *Assets*

- Permiten personalizar el comportamiento de los objetos mediante programación

## Se añaden como componente a los objetos

- Cuando un *Game Object* tiene un componente *script*, este código se ejecuta sobre el objeto

## Se pueden utilizar diferentes lenguajes

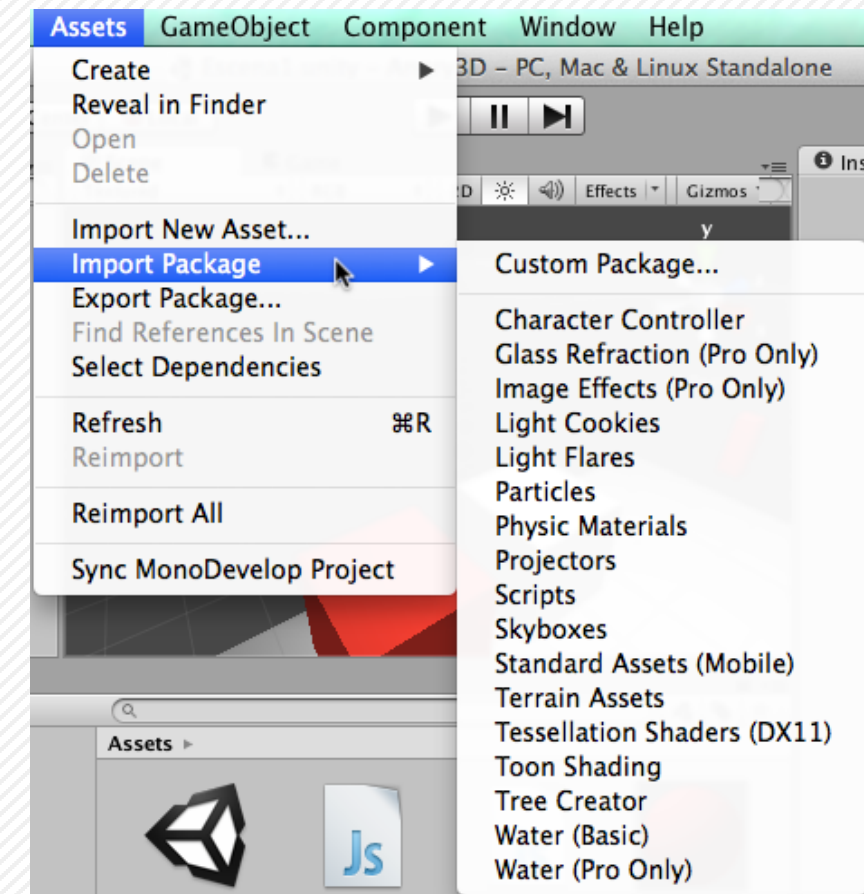
- *UnityScript* (basado en *Javascript*)
- *C#*
- *Boo*



## Paquetes de assets

### Podemos importar paquetes de assets

- Unity incorpora una serie de Assets predefinidos
- Menú *Assets > Import Package*



### Algunos ejemplos:

#### Scripts

##### *Cameras*

Scripts para cámaras

##### *Characters*

Scripts para manejo de personajes

#### Render

##### *Vegetación*

##### *Agua*

##### *Skyboxes*



# Controlador de personajes primera persona

## Importamos *assets* del paquete *Characters*

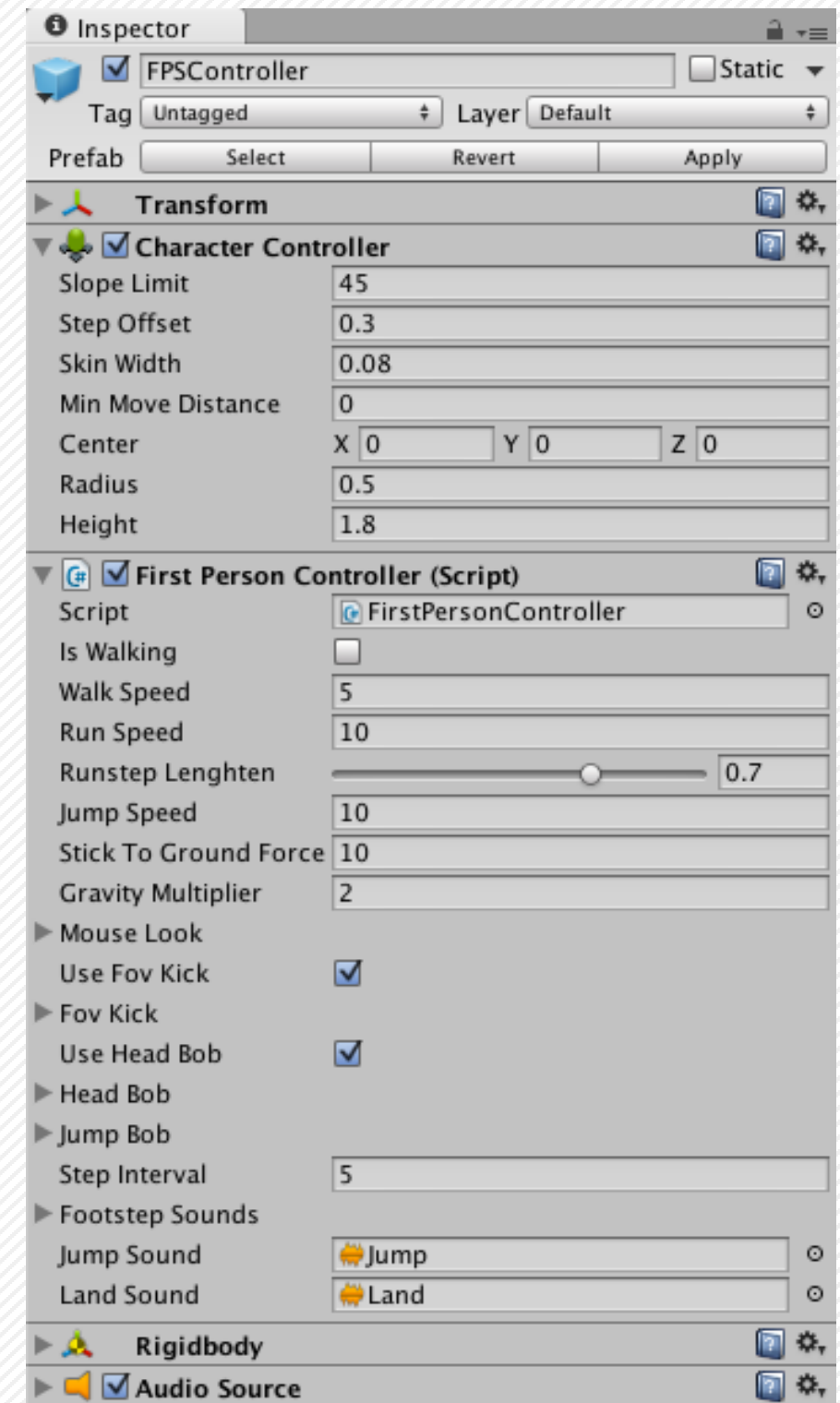
- Añadimos a la escena el prefab *FPS Controller*

## Es **configurable** desde el inspector

- Project Settings > Input

## Podremos **manejar** el personaje con

- *W,A,S,D*: Desplazarse
- *Espacio*: Saltar
- *Shift*: Andar / Correr
- Ratón: Mirar







# Otros controladores

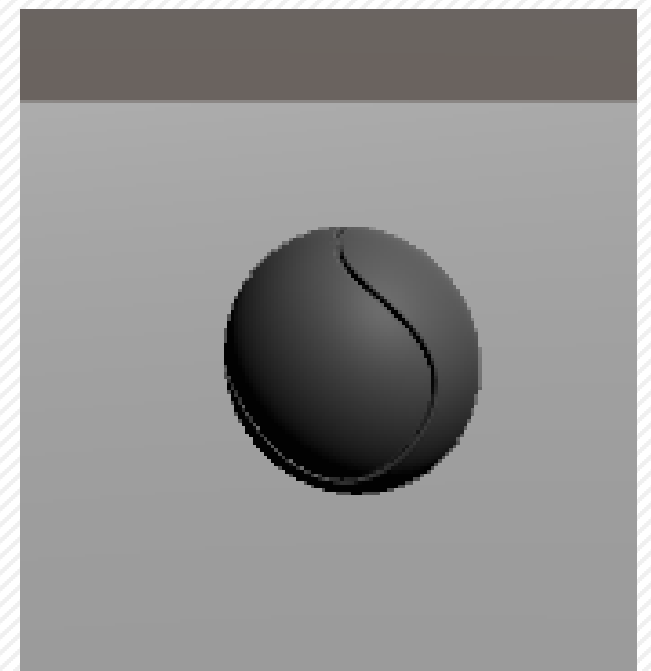
## Controlador de personaje en **tercera persona**

- Los controles son similares a los del FPS
- Añadimos a la escena el prefab *Third Person Controller*
- Podemos modificar el modelo gráfico y animaciones del personaje



## Controlador para **mover una pelota**

- Al pulsar los controles haremos que ruede por el escenario
- Añadimos a la escena el prefab *Roller Ball*
- Utilizará físicas





## Script PlayerController (C#)

```
using UnityEngine;
using System.Collections;

public class PlayerController : MonoBehaviour {

    public float velocidad;

    private Rigidbody rb;

    void Start()
    {
        rb = GetComponent<Rigidbody>();
    }

    // Update is called once per frame
    void FixedUpdate ()
    {
        float posH = Input.GetAxis ("Horizontal");
        float posV = Input.GetAxis ("Vertical");

        Vector3 movimiento = new Vector3 (posH, 0.0f, posV);

        rb.AddForce(movimiento * velocidad);

    }

}
```



# Script CameraController (C#)

```
using UnityEngine;
using System.Collections;

public class CameraController : MonoBehaviour {

    public GameObject player;

    private Vector3 desplazamiento;

    // Use this for initialization
    void Start () {
        desplazamiento = transform.position - player.transform.position;
    }

    // LateUpdate is called once per frame at the end
    void LateUpdate () {
        transform.position = player.transform.position + desplazamiento;
    }
}
```



# ¿Preguntas...?