

**MP\_0489. Programación  
multimedia y dispositivos móviles**

**UF4. Análisis de motores de juegos**

**4.4. Game Objects**

# Índice

---

☰	Objetivos	3
☰	Qué es un GameObject	4
☰	Componentes de un GameObject	6
☰	Trabajar con GameObjects	9
☰	Qué es un Prefab	13
☰	Resumen	16

# Objetivos

---

Con esta unidad perseguimos los siguientes objetivos:

1

Saber que es un *GameObject* en Unity.

2

Aprender a trabajar con *GameObjects*, así como a modificar sus componentes.

3

Agrupar *GameObjects*.

4

Descubrir que son los *Prefabs*.

---

¡Ánimo y adelante!

# Qué es un GameObject

El *GameObject* es el **tipo de objeto más importante en Unity**, ya que cada objeto en nuestro juego es un *GameObject*.

Los *GameObjects* **necesitan propiedades especiales** antes de que puedan convertirse en un personaje, un ambiente, o un efecto especial.

**i** Los *GameObjects* son **contenedores** que guardan las diferentes piezas para hacer un personaje, una luz, un árbol, un sonido, o lo que sea que queramos construir. Estas piezas son los **componentes**.

Dependiendo del objeto que creamos, agregaremos diferentes combinaciones de componentes al *GameObject*. Unity tiene varios tipos de componentes integrados, y también se pueden crear componentes utilizando *scripts*.

Al crear un proyecto nuevo, por defecto, tendremos dos *GameObjects*, la cámara y la luz.



---

## Es importante recordar estos tres puntos:

- 1 Los *GameObjects* pueden contener otros *GameObjects*. En un nivel básico, este comportamiento útil permite la organización de los *GameObjects* que están relacionados entre sí. Y, más importante aún, los cambios en los *GameObjects* de los padres pueden afectar a sus hijos.
- 2 Los *Models* se convierten en *GameObjects*. Unity crea *GameObjects* para las distintas piezas del *Model*, que se pueden modificar como cualquier otro *GameObject*.
- 3 Todo lo contenido en la *Jerarquía* es un *GameObject*, incluso cosas como las cámaras y las luces. Si está en la *Jerarquía*, es un *GameObject*.

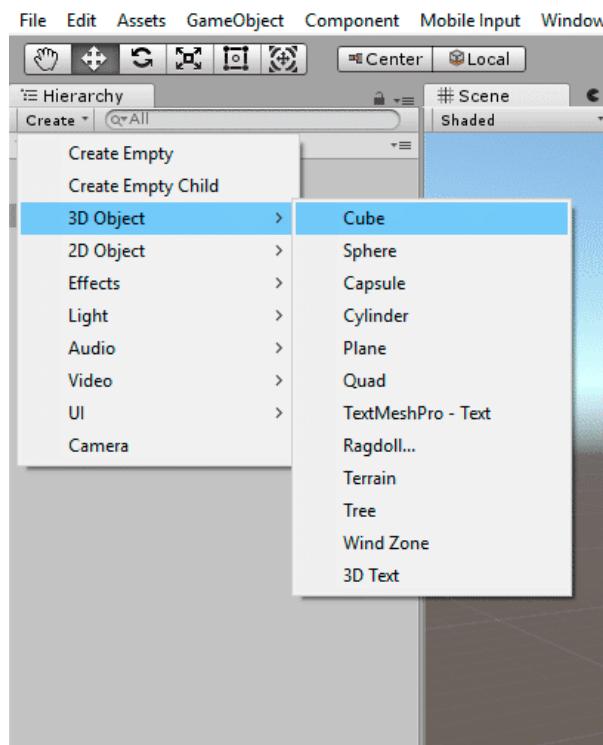
# Componentes de un GameObject

En el siguiente apartado te mostraremos los **componentes básicos** de un *GameObject*.

Y lo haremos paso a paso.

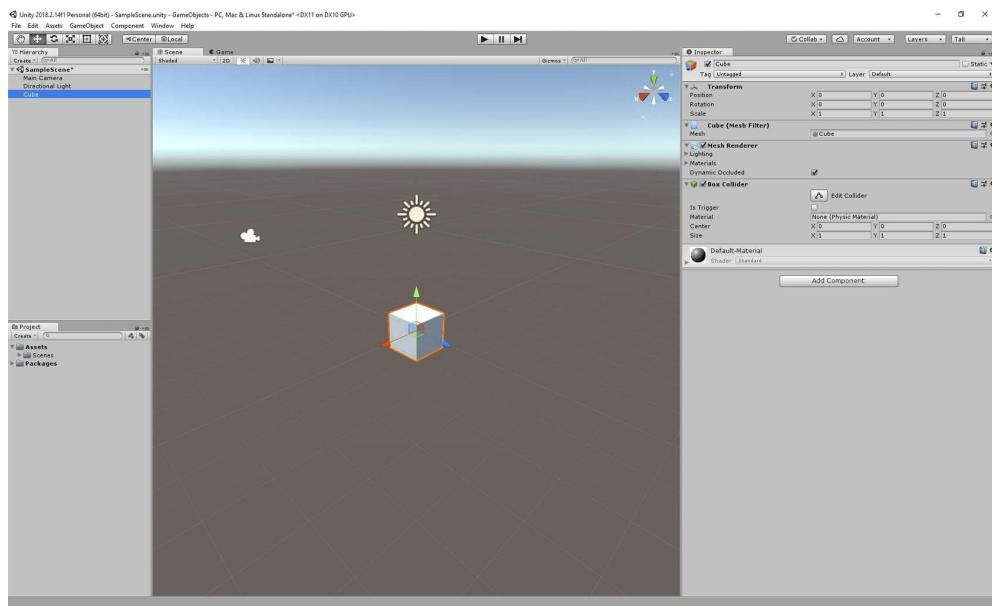
1

Vamos a **añadir** un sencillo *GameObject* a nuestra escena, en este caso un cubo 3D.



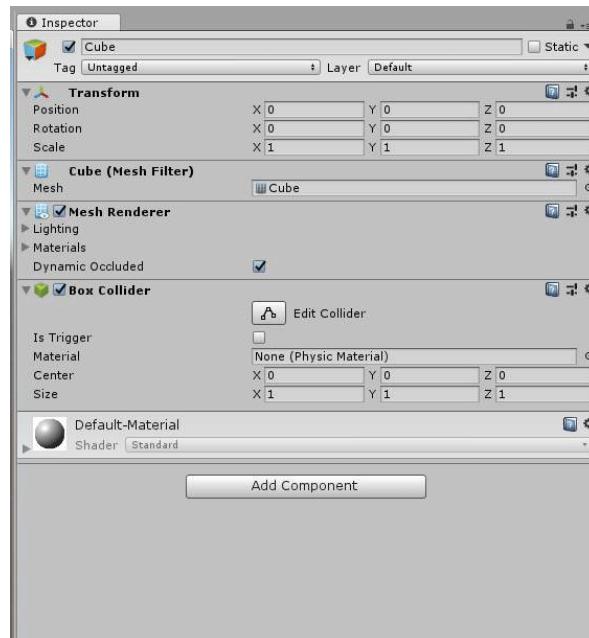
2

Podemos observar el cubo situado en el centro de la escena.



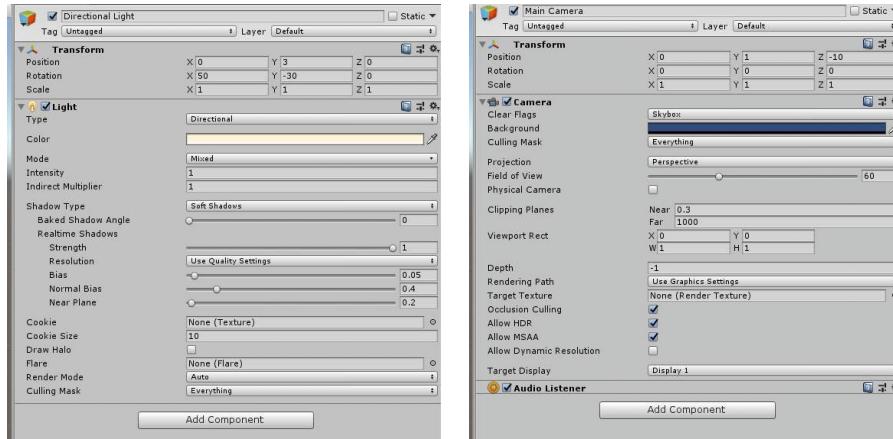
3

En el panel de *Inspector* aparecen los componentes del *GameObject*, entre ellos el de *Transform*, que poseen todos los componentes sean del tipo que sean.



4

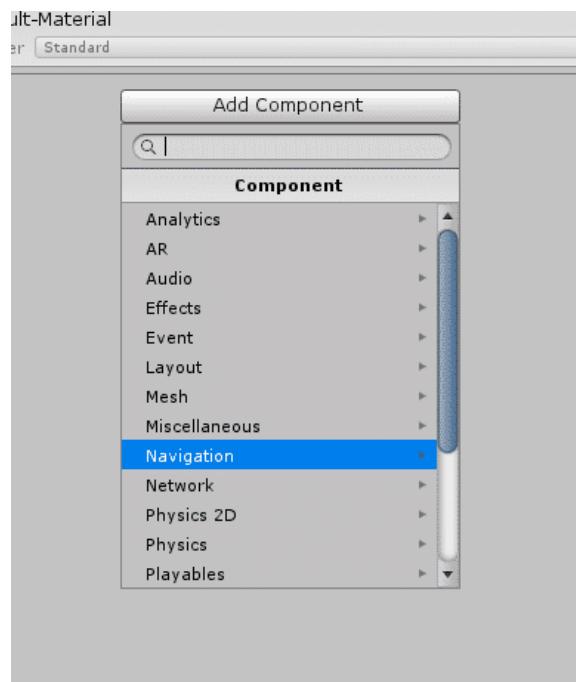
Si seleccionamos tanto la luz como la cámara, también podremos ver sus componentes y, como hemos dicho, aparecerá siempre el componente *Transform*.



5

Observamos que es posible añadir más componentes a nuestro *GameObject*; tan solo debemos pulsar sobre el botón *Add Component* que encontraremos en el panel *Inspector*.

Aparecerá un desplegable con un listado de componentes agrupado por distintos tipos.



# Trabajar con GameObjects

Veamos a continuación cómo trabajar con los componentes de un *GameObject*.

Y cómo añadir un nuevo *GameObject*.

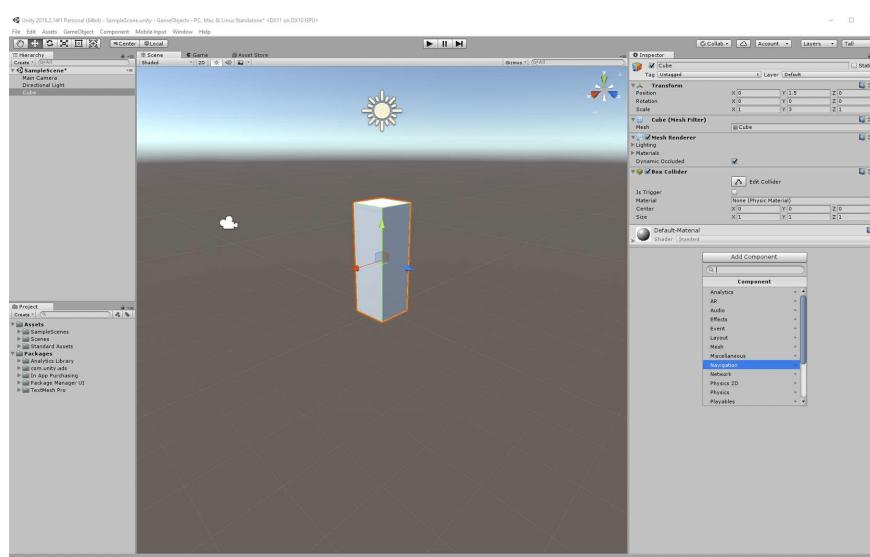
En el panel de *Inspector* vamos a ver el componente *Transform*, que tiene tres características:

- **Position:** indica la posición del *GameObject* con valores en las tres coordenadas x, y y z.
- **Rotation:** indica la rotación del *GameObject* con valores en las tres coordenadas x, y y z.
- **Scale:** indica la escala del *GameObject* con valores en las tres coordenadas x, y y z.

Hagamos algunas modificaciones en nuestra escena anterior para ver cómo funciona:

1. Cambiamos la escala de nuestro cubo al valor 3 en la y.
2. Cambiamos la posición a 1.5 también en la y.
3. Además, modificamos la posición y de luz, poniendo 3.

Este es el nuevo aspecto de nuestra escena.

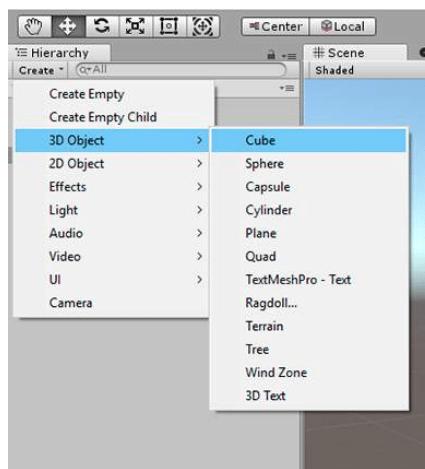


## Añadir un nuevo GameObject

Vamos a añadir ahora un nuevo *GameObject* a la escena, y veremos cómo agruparlo para poder trabajar con un *GameObject* compuesto de varios *GameObjects*.

1

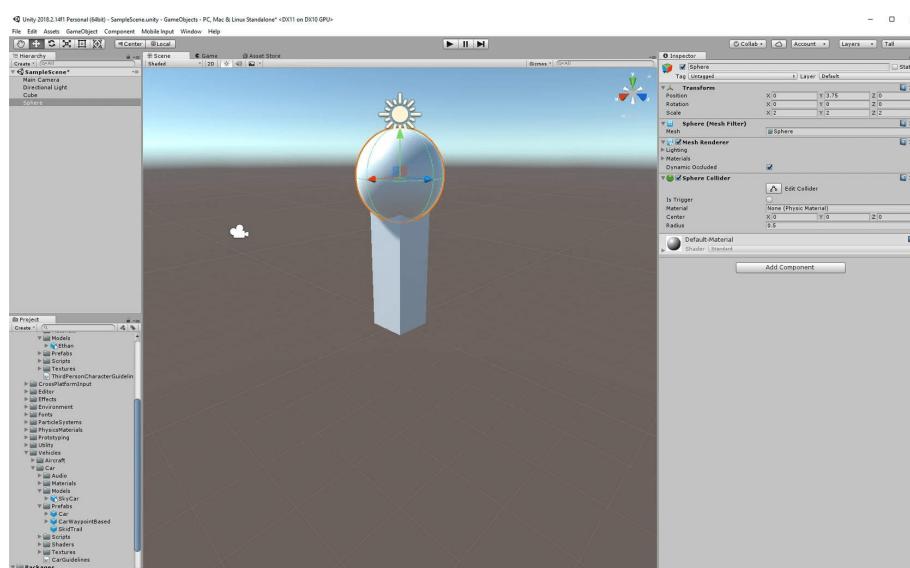
Lo primero que haremos será **añadir una esfera a nuestra escena**. Lo haremos desde el menú *Create* del panel *Jerarquía*.



2

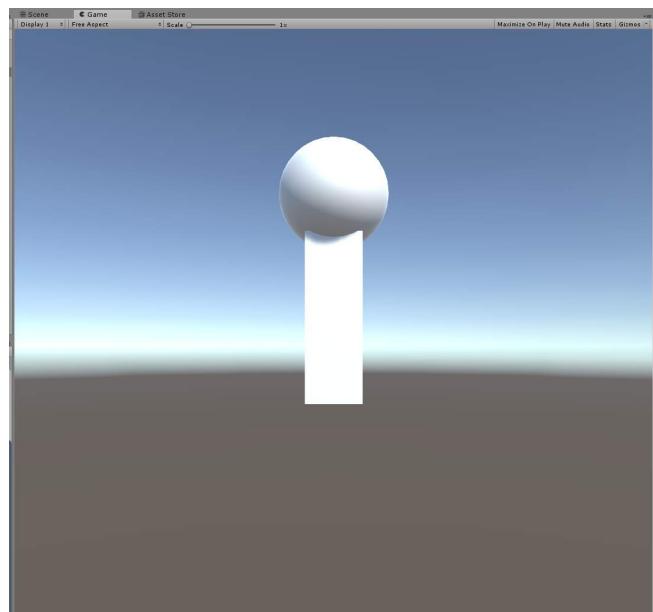
Una vez añadida, modificamos algunos valores en el componente *Transform*:

- En la posición, pondremos y en 3.75
- Y cambiaremos los tres valores de la escala a 2.



3

Al ver la **vista de la cámara**, deberíamos encontrar algo así:



### Hacer modificaciones en el nuevo **GameObject**

Si queremos hacer modificaciones en este momento, deberíamos hacerlas tanto en el cubo como en la esfera, lo que será un incremento de trabajo que, en el caso de un *GameObject* más complejo, podría suponer una labor titánica.

Por eso vamos a crear un *GameObject* padre en el que insertaremos el cubo y la esfera, para poder hacer cambios de forma más sencilla.

1

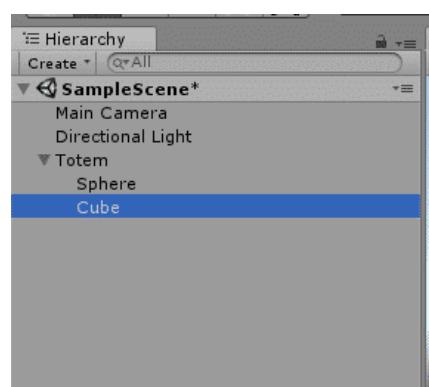
Lo primero que haremos será **crear un *GameObject* vacío** desde el menú *Create* del panel de *Jerarquía*.

2

Después, **lo renombraremos y lo llamaremos *totem***.

3

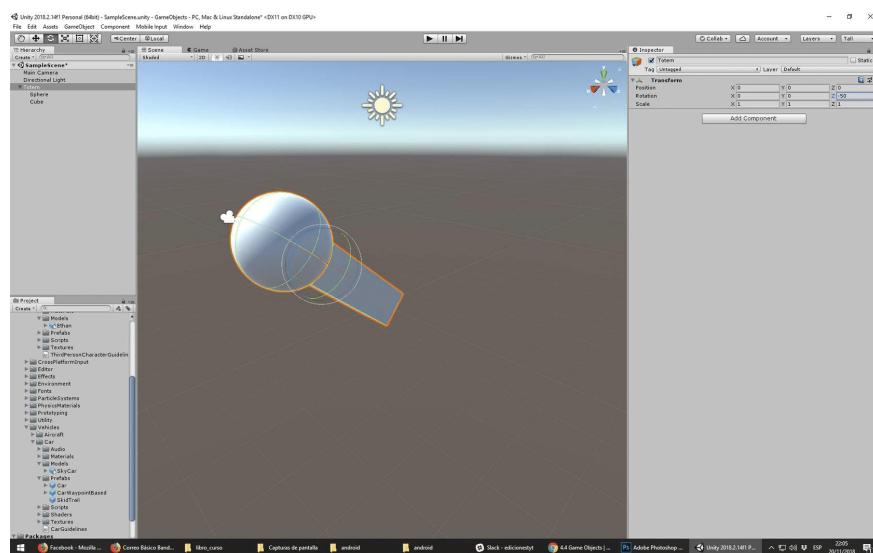
Por último, arrastraremos el cubo y la esfera dentro de *totem* en el panel de *Jerarquía*.



4

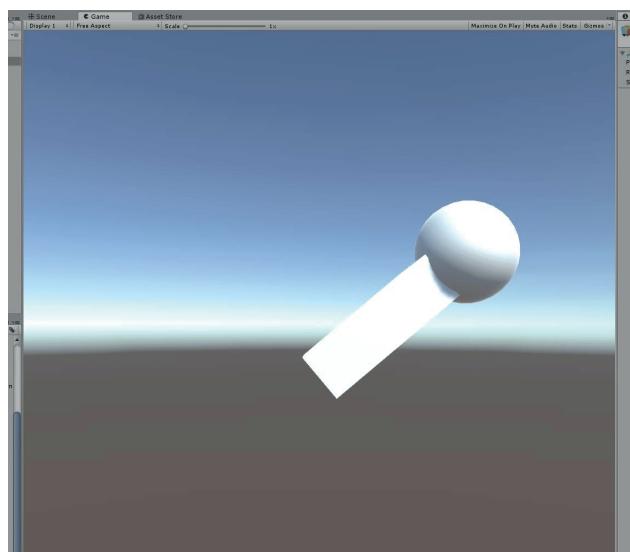
Ahora podemos modificar los valores de *Transform* de *totem*.

Por ejemplo, ponemos la rotación del eje z en el valor -50. Observaremos cómo se han movido tanto el cubo como la esfera como si fueran un solo objeto.



5

Y así lo veríamos a través de la cámara:



Cuando tengamos un *GameObject* dentro de otro, la posición del *GameObject* hijo no cambiará. La diferencia es que la posición del *GameObject* hijo ahora estará posicionada en relación con el padre.

---

Es decir, **establecer el hijo en (0, 0, 0)** moverá al hijo al centro del padre, no al frente del centro del juego.

# Qué es un Prefab

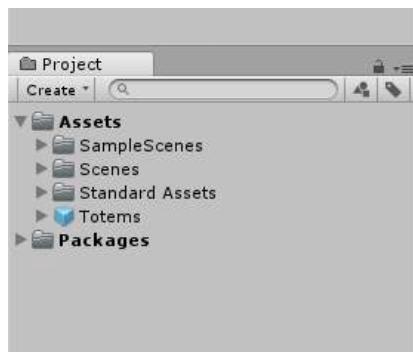
*Prefab* es un tipo de *asset* que permite almacenar un objeto *GameObject* con componentes y propiedades.

El *prefab* actúa como **una plantilla a partir de la cual se pueden crear nuevas instancias del objeto en la escena**. Cualquier edición hecha a un *prefab* será inmediatamente reflejada en todas las instancias que produzca, pero también se puede ajustar para cada instancia individualmente.

## Convertir un *Totem* en un *Prefab*

1

Vamos a convertir nuestro *totem* en un *prefab*. Para ello, desde el panel de *Proyectos*, creamos un *Prefab* y lo renombramos como *Totem*.



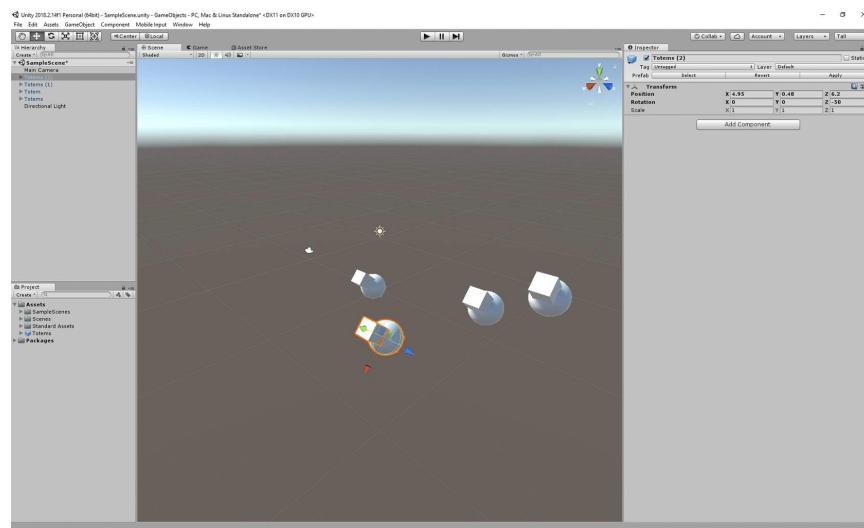
2

Después, arrastramos el *Totem* que tenemos en el panel de *Jerarquía* hasta el *Prefabs Totems* del panel de *proyectos*. De esta manera, nuestro *Prefab* vacío tiene ahora el *GameObject Totem* dentro.

3

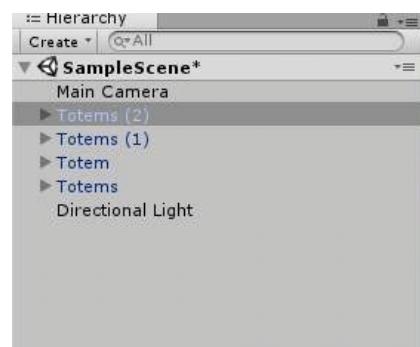
Arrastramos desde el panel de *proyectos* tres *prefabs* más a la *escena* y modificamos su posición para poder verlos desde la cámara. Los cuatro *Totems* que tenemos en *escena* tendrán estas posiciones:

- X: 0 Y: 0 Z: 0
- X: 4.95 Y: 0.48 Z: 6.2
- X: 0.56 Y: 2.16 Z: 9.92
- X: -0.2 Y: 3.66 Z: 12.87



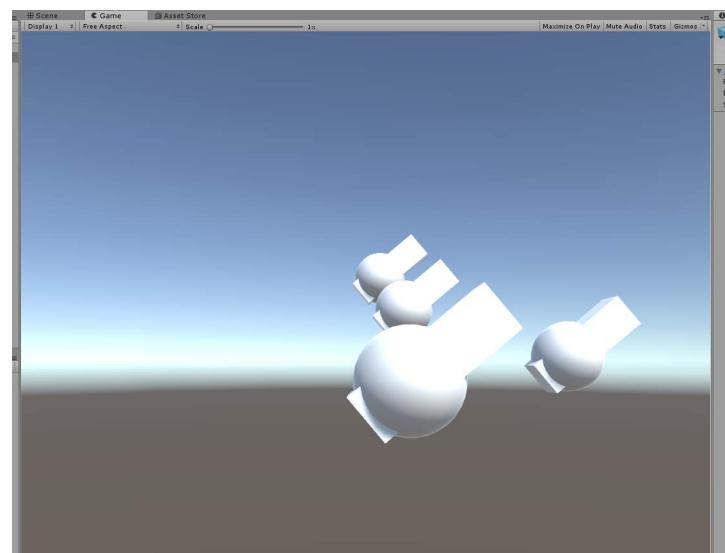
4

En el panel de *Jerarquía* veremos los cuatro *Prefabs* como distintos *GameObjects*.



5

Al observar la vista de la cámara, deberíamos ver algo así:

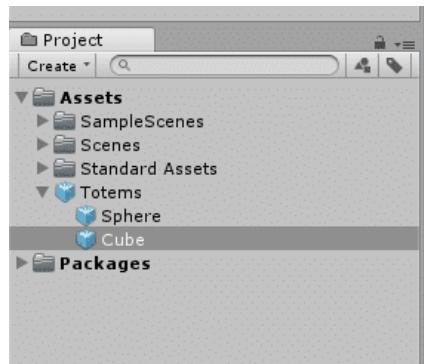


## Hacer modificaciones en el *Prefab*

Si queremos hacer modificaciones en el *Prefab*, podemos hacer que estas ocurran en todos los *prefabs* de la escena.

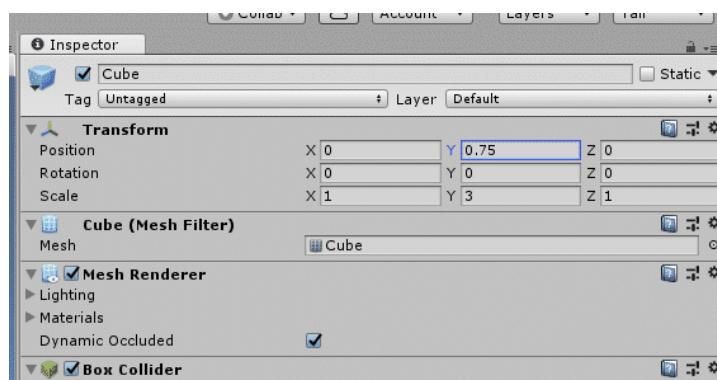
1

Para ello, seleccionamos el panel proyecto *Totems* y, una vez desplegado, seleccionamos el cubo.



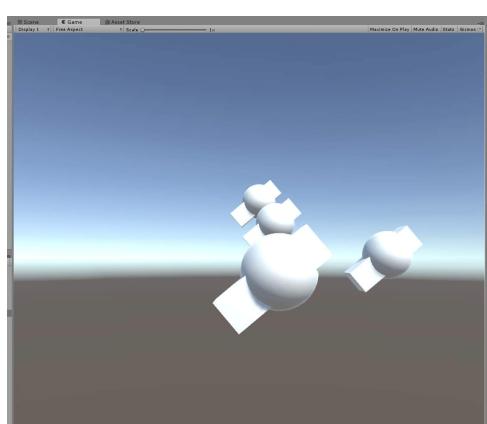
2

Ahora podemos modificar los valores de *Transform* del cubo. Por ejemplo, ponemos la posición en el eje y en el valor 0.75. Observamos cómo se han movido todos los cubos que se encuentran dentro de los *Prefabs*.



3

Y así lo veríamos a través de la cámara:



# Resumen

---

Hemos terminado la lección, repasemos los puntos más importantes que hemos tratado.

- A lo largo de esta unidad hemos aprendido que los *GameObjects* son todos los objetos que vamos a tener en una escena.
- Hemos aprendido a **modificarlos** con el componente básico que todos los *GameObjects* poseen: *Transform*.
- También hemos creado un *GameObject padre*, que contiene en su interior otros *GameObjects*.
- Y, para finalizar, hemos creado un *prefab*, que es un *asset* que permite usarse infinidad de veces y modificarse de golpe.

