



**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**

## Práctica 3: Primer proyecto

### 1. Crea un nuevo proyecto en Unity, del tipo móvil 2D.

- Este debe llamar se DAM\_UN1\_Nombre

The screenshot shows a dual-monitor setup. The left monitor displays a Microsoft Word document with the title 'Práctica 3: Primer proyecto'. The right monitor displays the Unity Hub application. In the Unity Hub, a new project is being created with the name 'DAM\_UN1\_Manuel'. The '2D (UWP) Core' template is selected. The Unity Editor window is visible on the far right, showing a simple 2D scene with a cube.

The screenshot shows the Unity Editor running on a second monitor. The main view displays a 2D scene with a large, colorful torus object that is rotating. A Unity Package Manager window is open at the bottom of the screen, showing the progress of installing packages for the project 'DAM\_UN1\_Manuel'. The Unity logo is visible in the bottom left corner of the editor window.



**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**

- b. Tenéis que crear un documento y pasar el proyecto.

No sé a qué te refieres, te paso el repositorio del proyecto por si es a lo que te refieres.

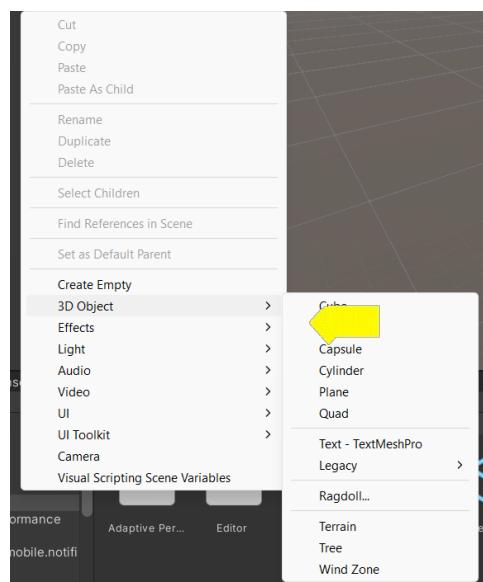
<https://github.com/manuelmsni/androidUnityGame>

Es el commit con id: 0e2adbc

Creación de una pelota que rebota

2. Vamos a crear un nuevo objeto.

- Click derecho en jerarquía
- 3D object.
- Seleccionamos “Sphere” para crear la pelota.

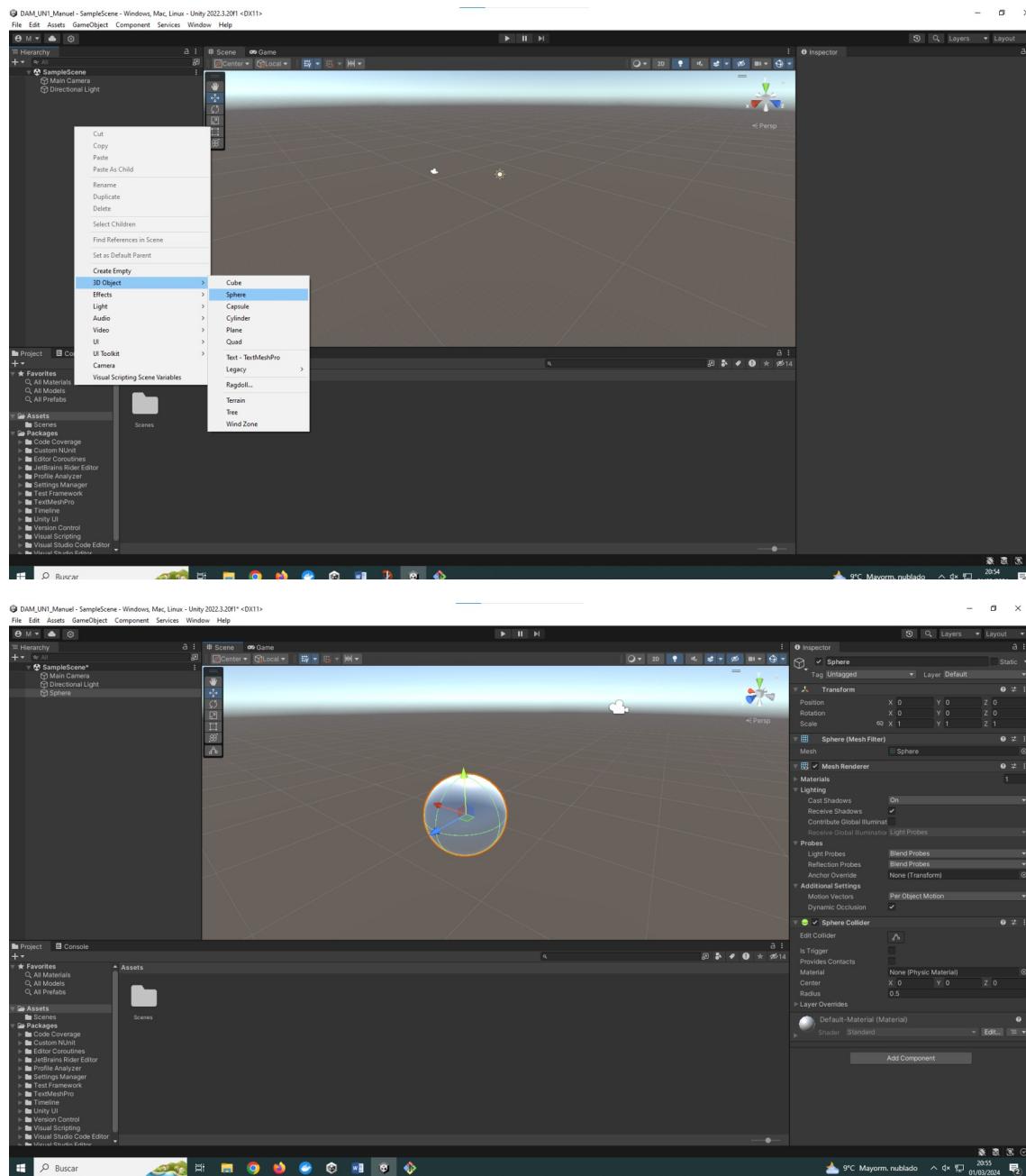




# 2º DAM

## Programación multimedia y móviles

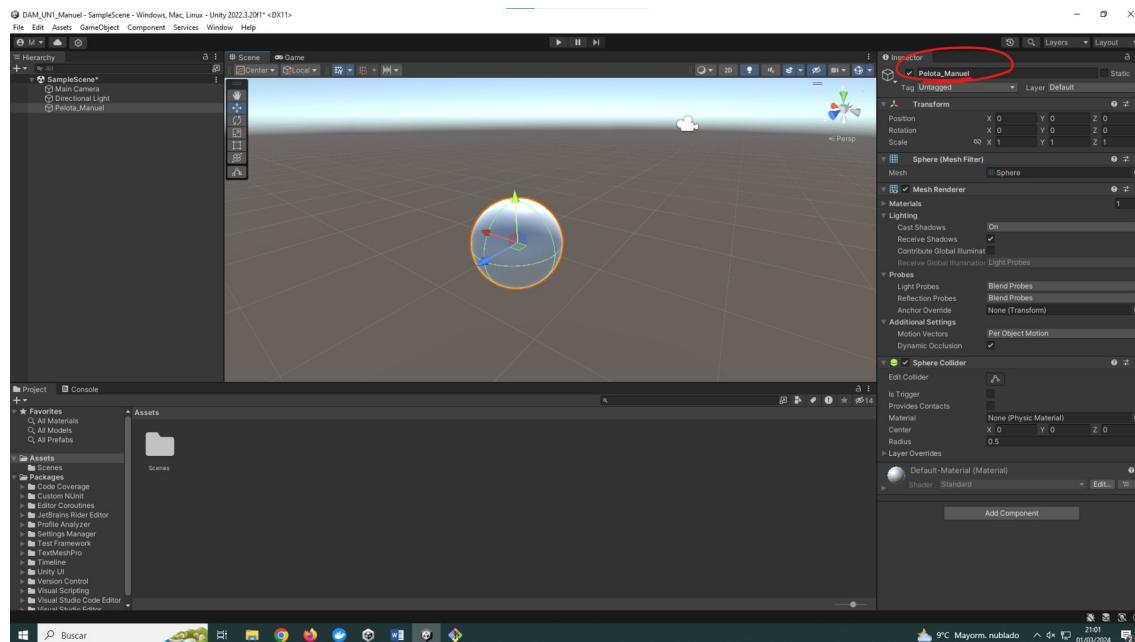
### U.D. 05.- Unity



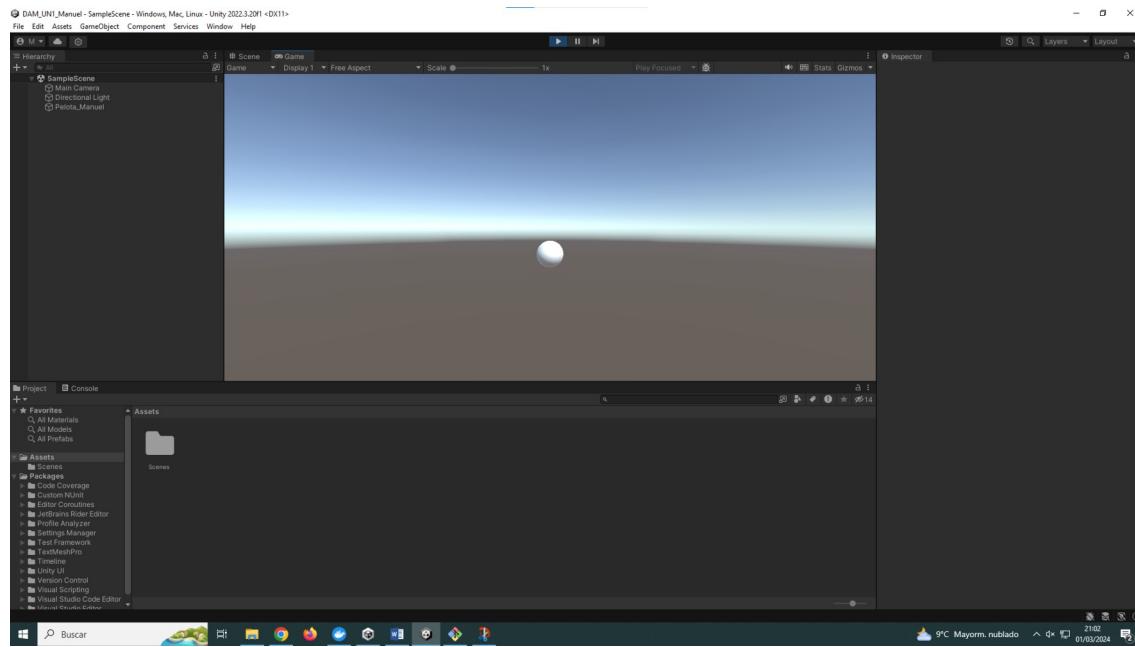
3. Selecciona la pelota y comprueba sus características en el inspector.
  - a. Cambia el nombre por Pelota\_nombre



**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**



**4. Inicia la aplicación. ¿Qué sucede con la pelota?**



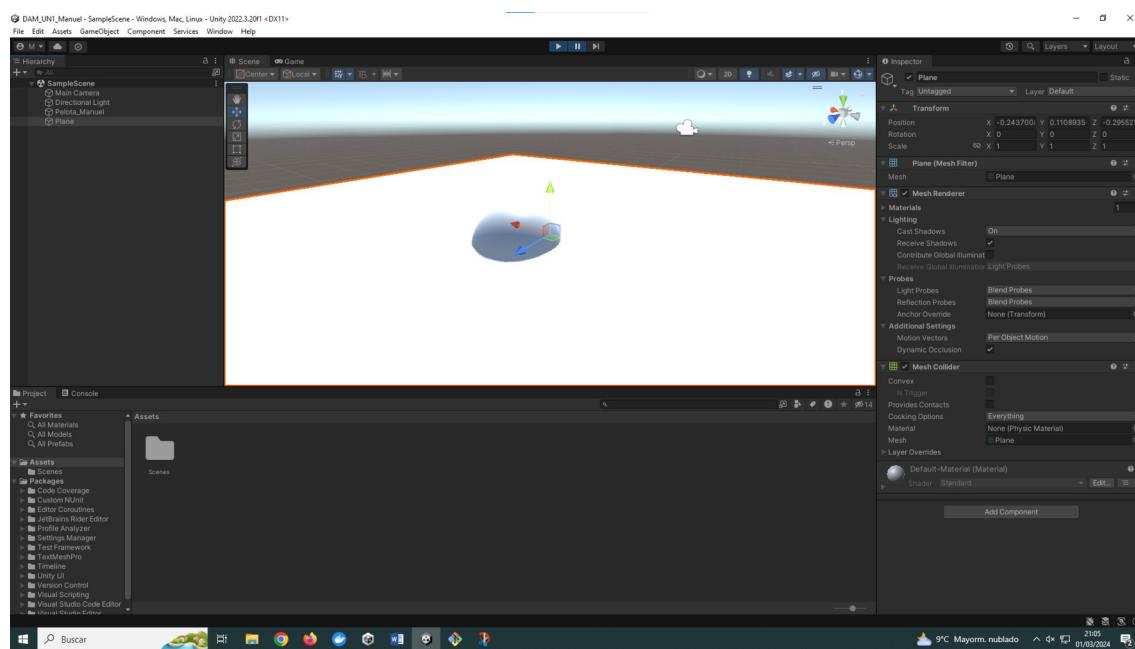
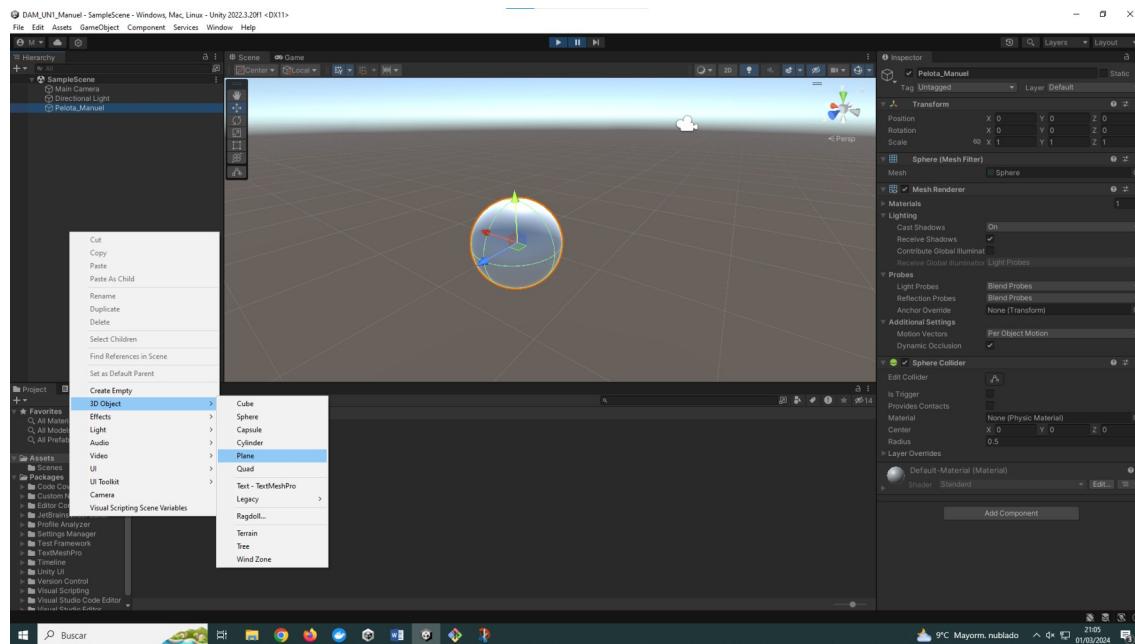
Se queda parada porque no tiene RigidBody o alguna clase de físicas.

**5. Una vez creada vamos a gestionar que rebote.**

- Creamos un plano con click derecho Object3D -> Plain



**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**



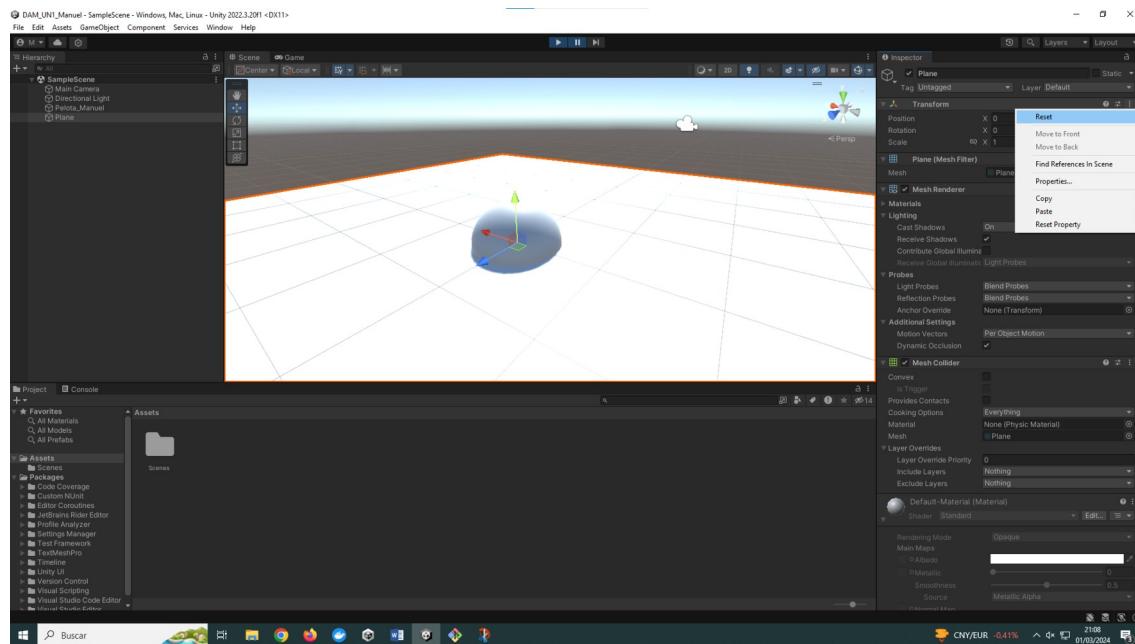
b. Transform -> Reset para colocarlo en el 0,0.



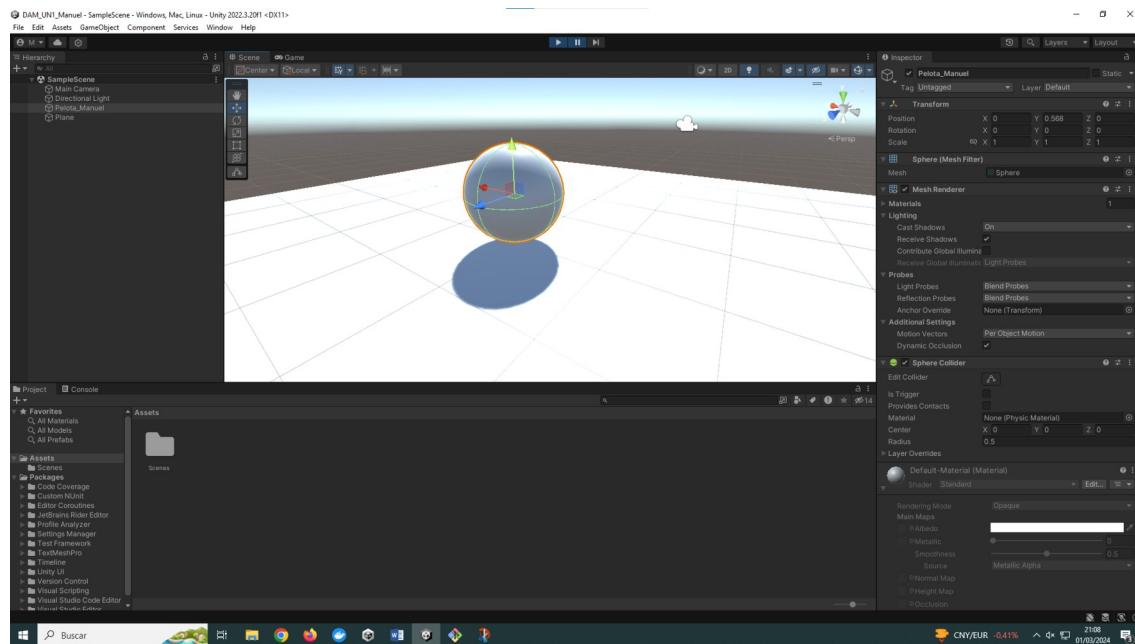
## 2º DAM

### Programación multimedia y móviles

#### U.D. 05.- Unity



6. Subimos un poco la pelota.



7. Para hacer que rebote en el plano.

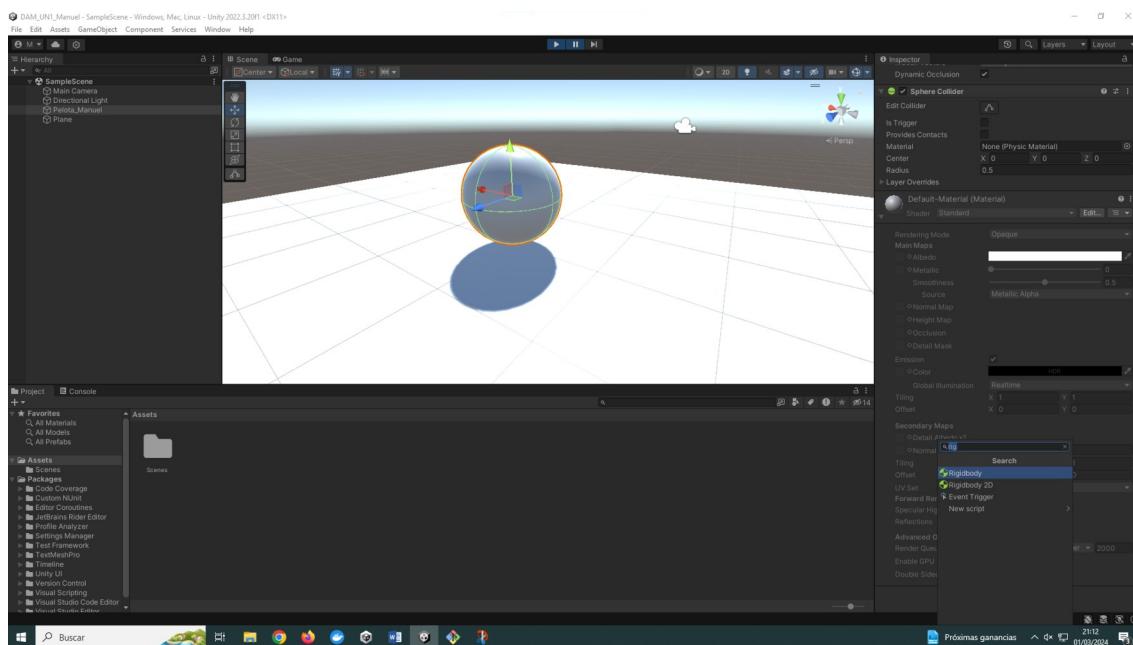
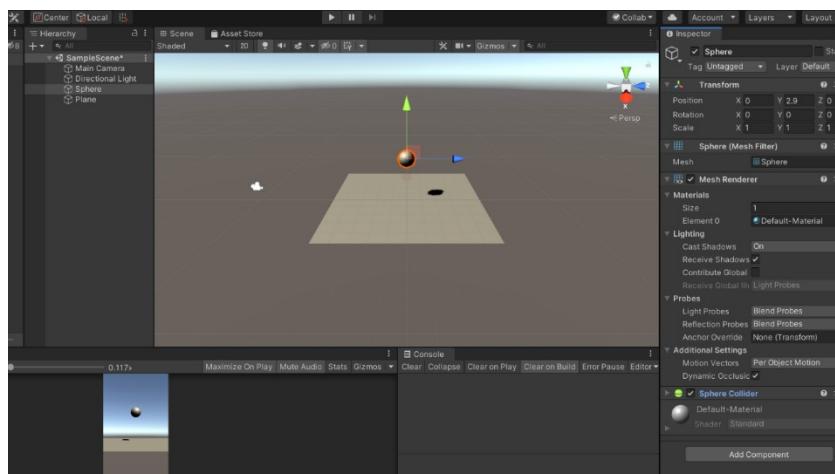
- Necesitamos un componente **RigidBody** que es el que nos va a dar la gravedad, masa y demás características.
- Le damos a **Add Component** y creamos un **RigidBody**.



2º DAM

## Programación multimedia y móviles

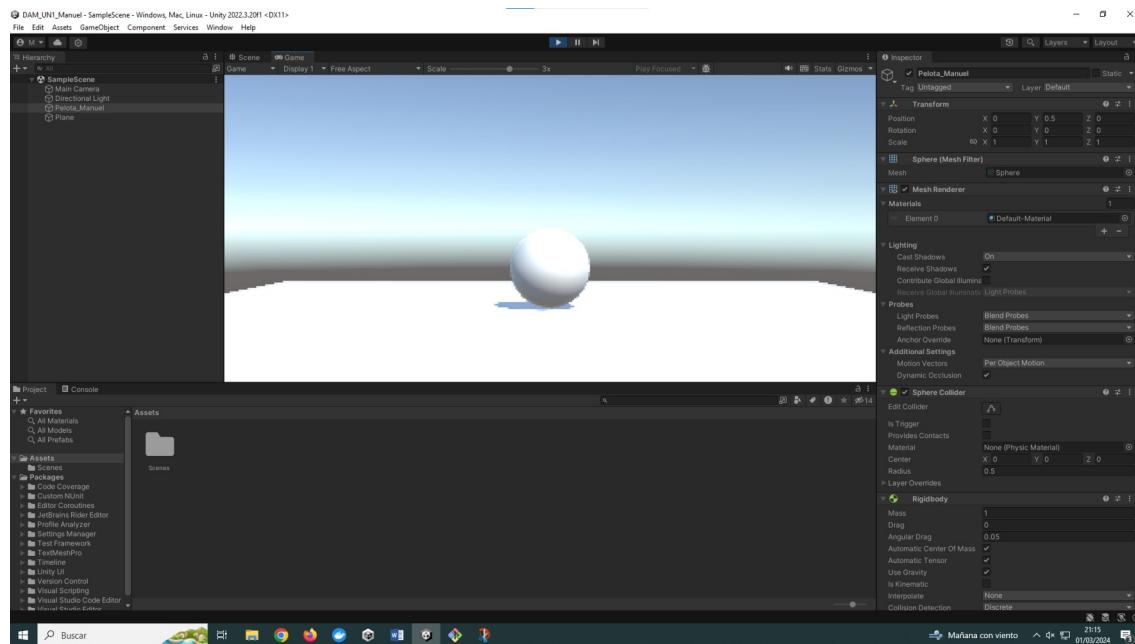
U.D. 05.- Unity



8. Vuelve a ejecutar la aplicación, ¿qué sucede esta vez con la pelota?

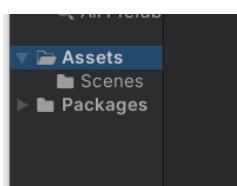


**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**



Se cae al plano, porque el RigidBody tiene físicas.

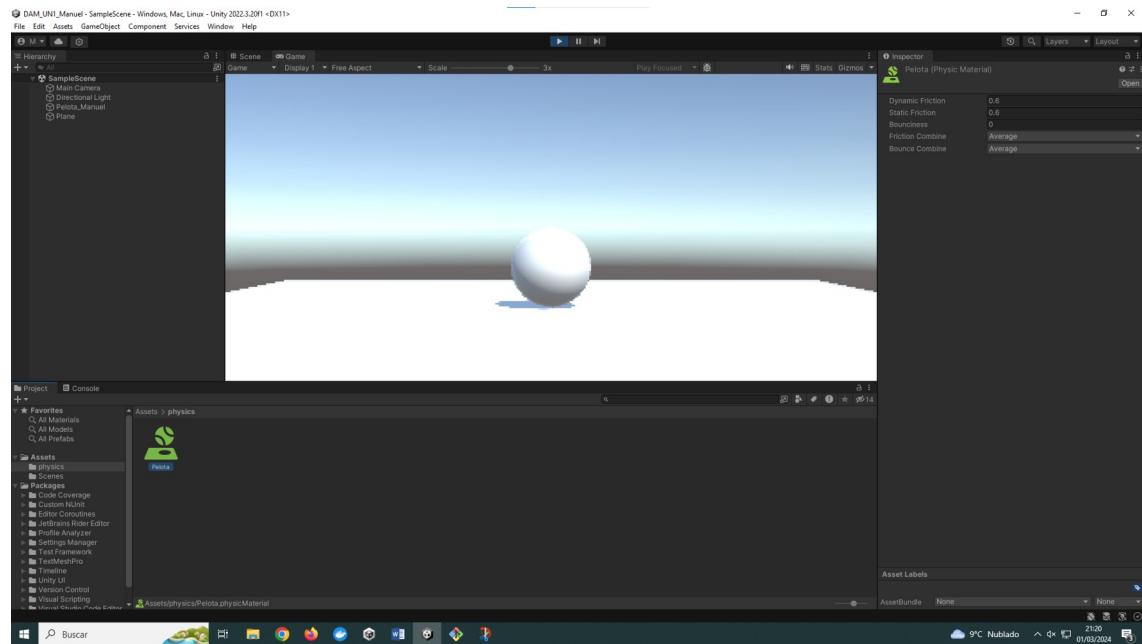
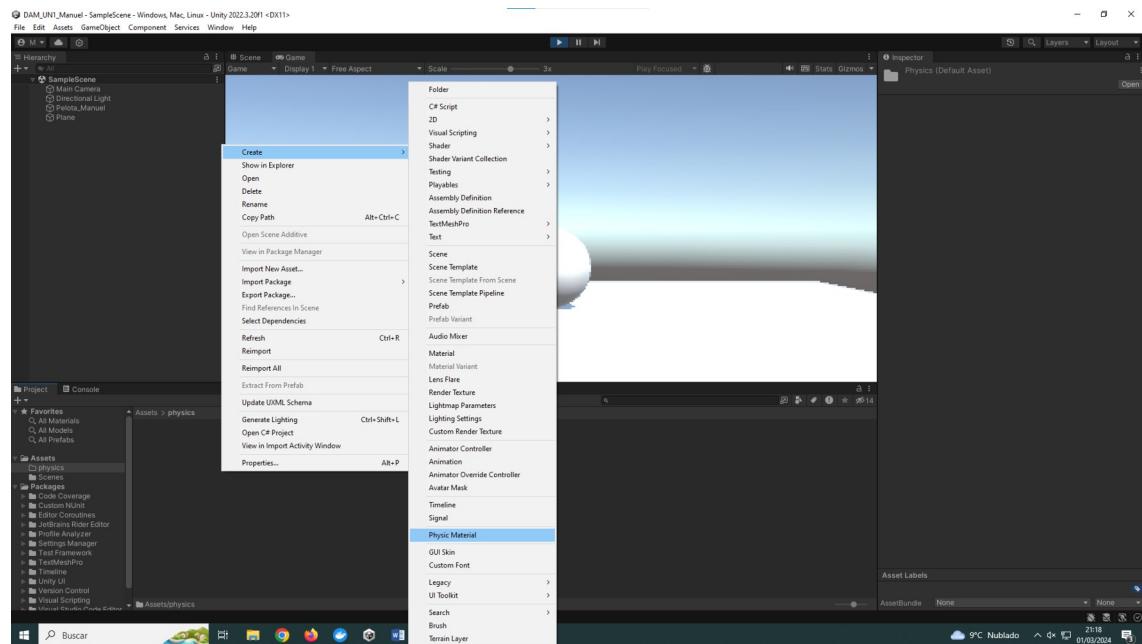
9. Tenemos que configurar que no sólo caiga, sino que rebote.
  - a. Para hacer que rebote creamos un **Physic Material**, que trae diferentes características como:
    - i. fricción dinámica
    - ii. fricción estática (fricción con el aire y con otros objetos lo que hace que la pelota se vaya parando)
    - iii. bounciness que es para el rebote
  - b. Click derecho sobre Assets -> Create



y le llamamos “**Pelota**”, por ejemplo.



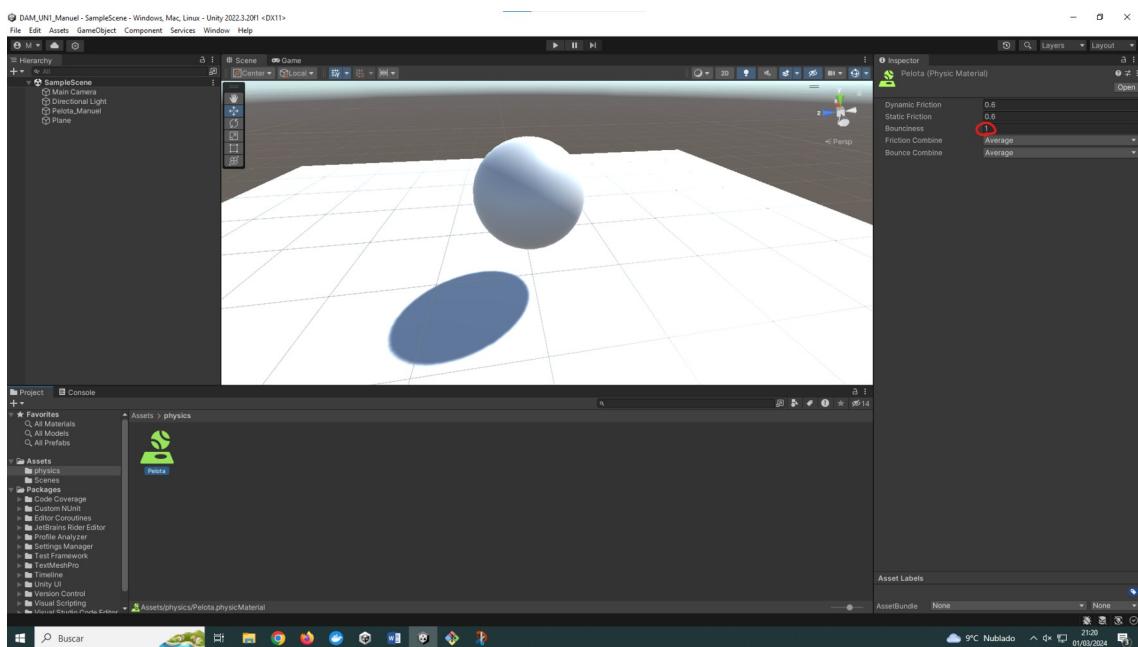
**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**



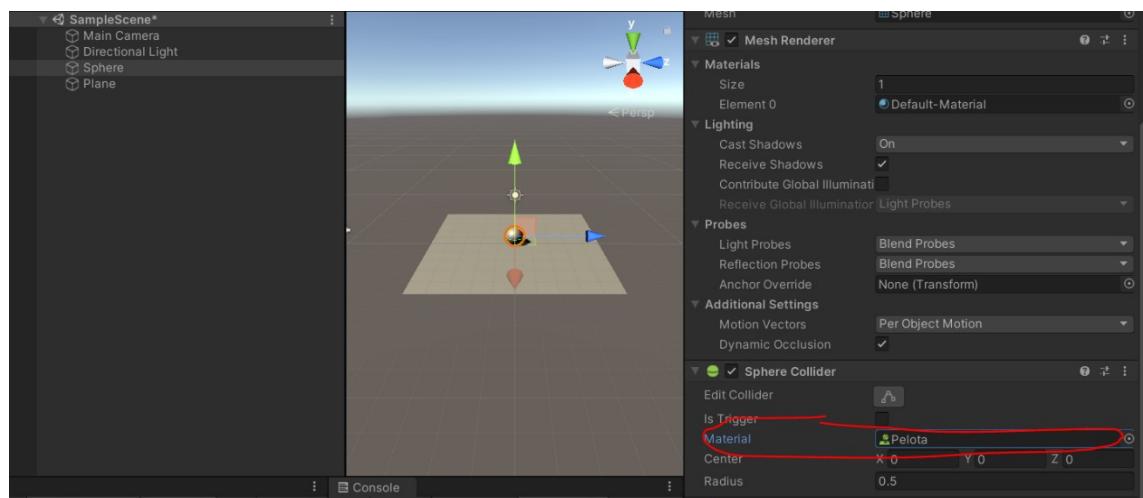
10. Cambiamos el bounciness a 1 (por defecto está a 0)



**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**



11. Vamos a la Sphere y en Sphere Collider cambiamos el material a “Pelota”

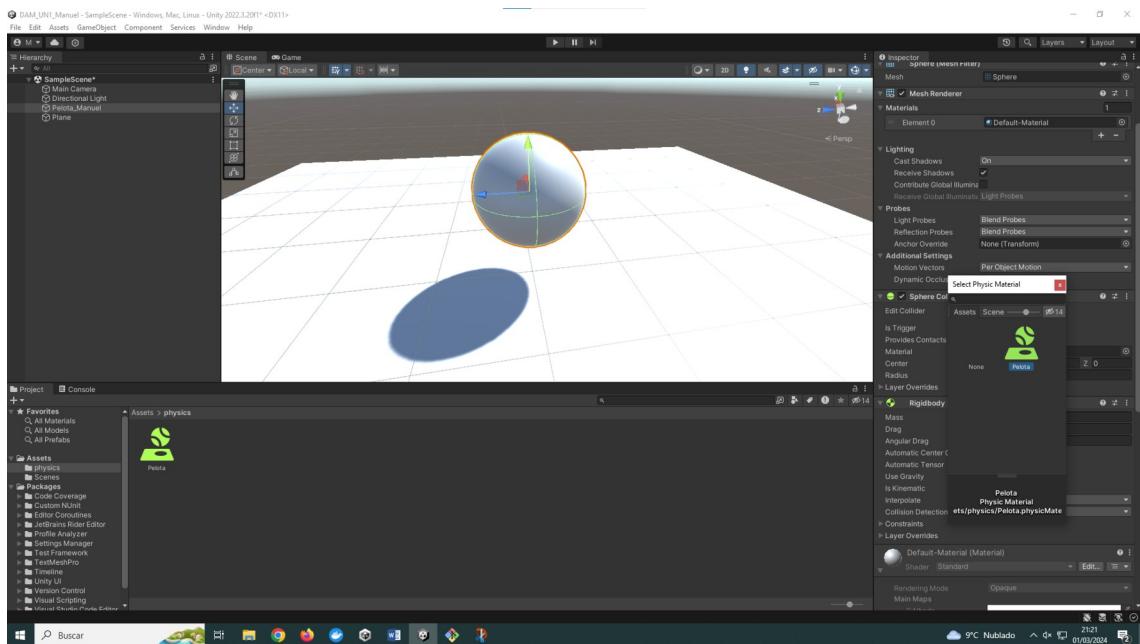




## 2º DAM

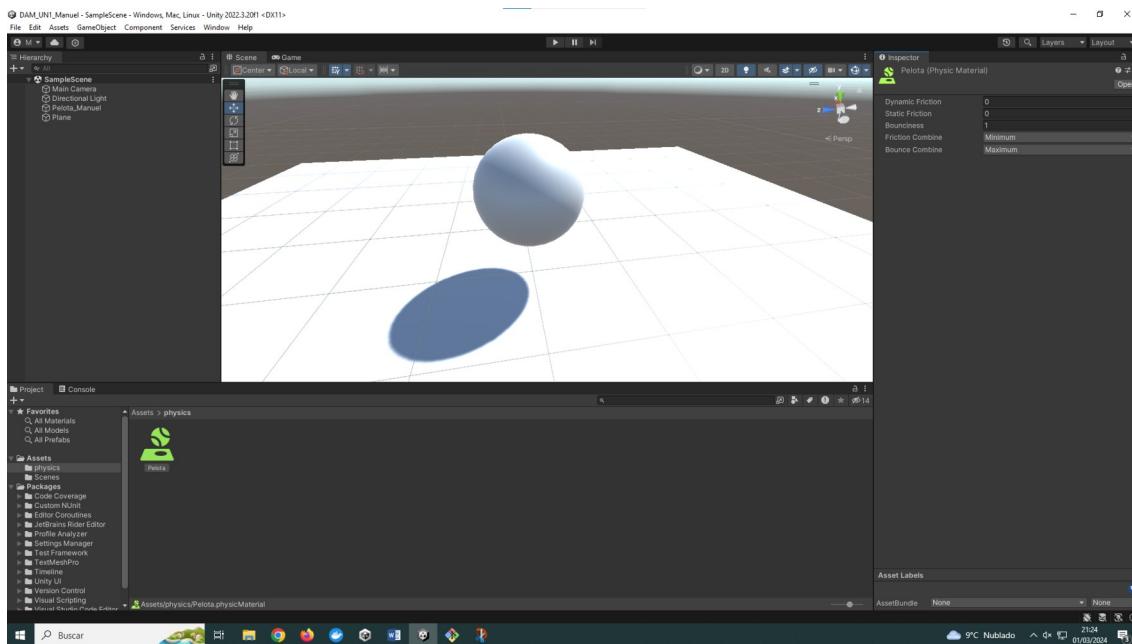
### Programación multimedia y móviles

#### U.D. 05.- Unity



12. Cuando le damos al Play ahora vemos que rebota un poco la pelota (hemos puesto el bounciness a 1), pero queremos que rebote más.

- Cambiamos la fricción estática y dinámica a cero y Bounce Combine a “Maximun” para que rebote lo más posible.



- Ejecuta la aplicación y manténla un rato ejecutando. ¿Qué sucede?

Ahora rebota de manera infinita, porque al no haber pérdida de energía en el sistema, al impactar en la superficie, la pelota experimenta un empuje igual pero opuesto al que le

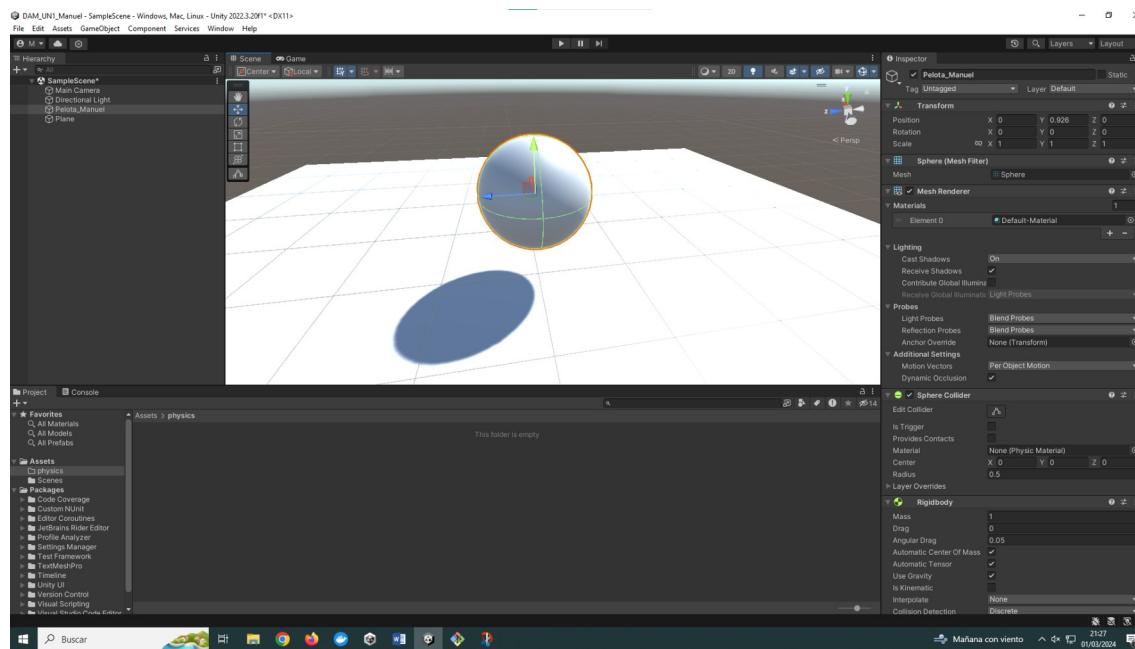


**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**

transmite en el impacto a la superficie, de echo algo mayor y por eso cada vez rebota más alto.

13. Como cada vez rebota más, no vamos a utilizar un **Physic Material**, sino que haremos un script. Aunque como vemos este tipo de componente es muy útil.

- a. Borramos “Pelota”.

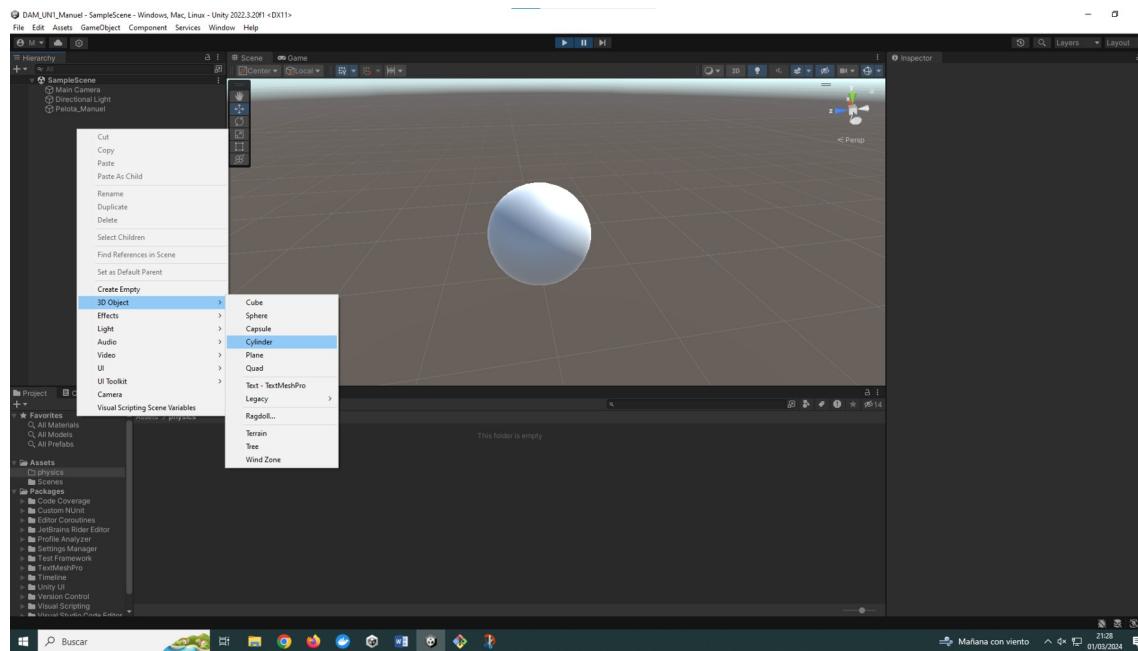


Creación del mapa por el que se moverá la bola

14. Vamos ha crear una torre por la que la bola irá bajando y que solo pueda rebotar una vez en cada peldaño.
15. Borrar el plano, para agregar directamente el cilindro que vamos a usar (3DObject -> Cylinder).

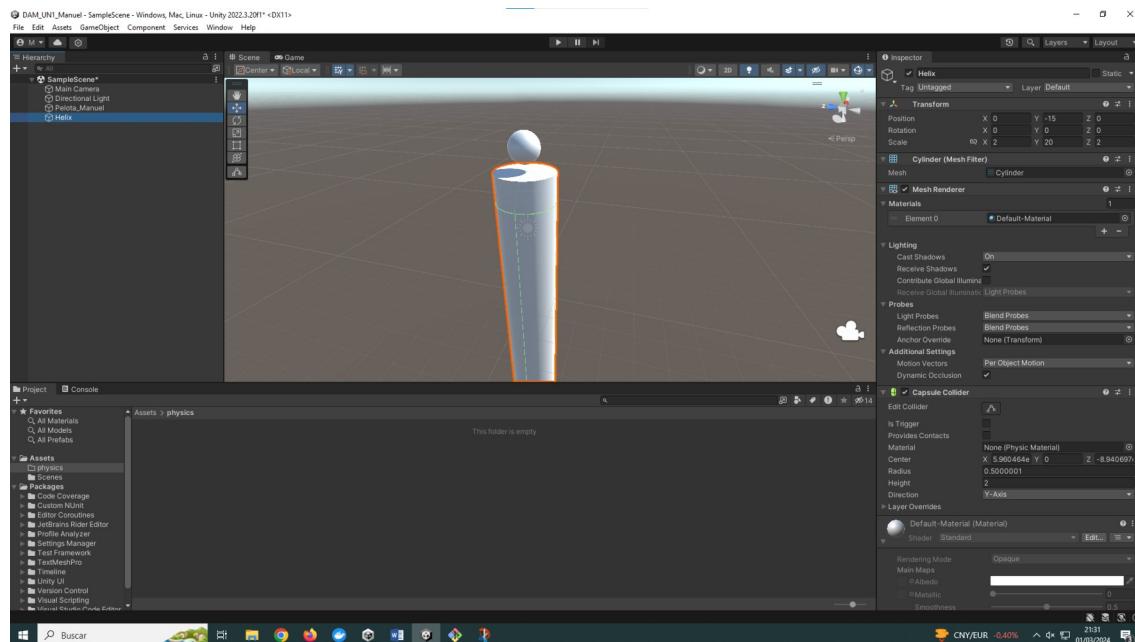


**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**



- a. Le llamamos **Helix** y le damos a Reset para la posición 0,0,0 y cambiamos:

- Su eje Y de Position a -15.
- En Scale la X a 2, la Y a 20 y la Z a 2 para que sea muy grande y muy alto



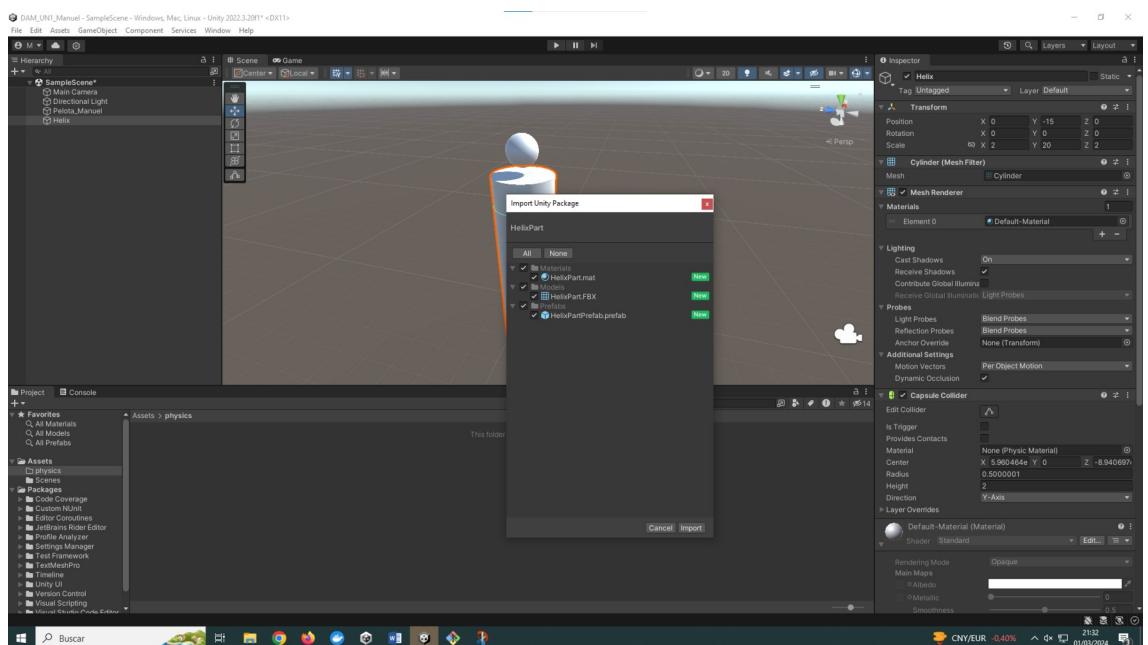


**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**

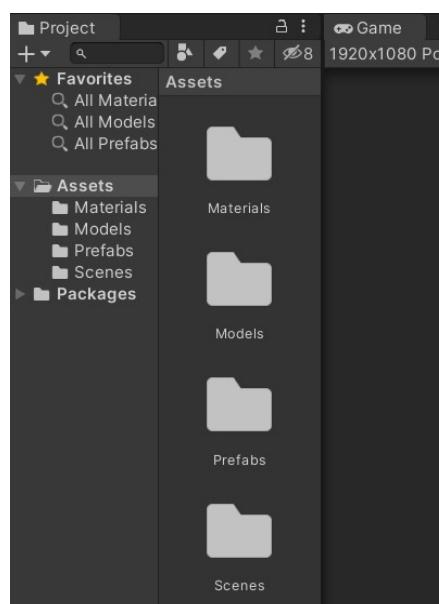
16. Nos descargamos el paquete

<https://mega.nz/file/MDZRharC#LgFy7sUnfl6fUDi72AtKxo9Hhvthqlt1DSjBKCLBeQ> que va a ser el que tiene las formas que vamos a añadir al cilindro (con quesitos, que harán de peldaños).

a. Lo abrimos y directamente se añade.



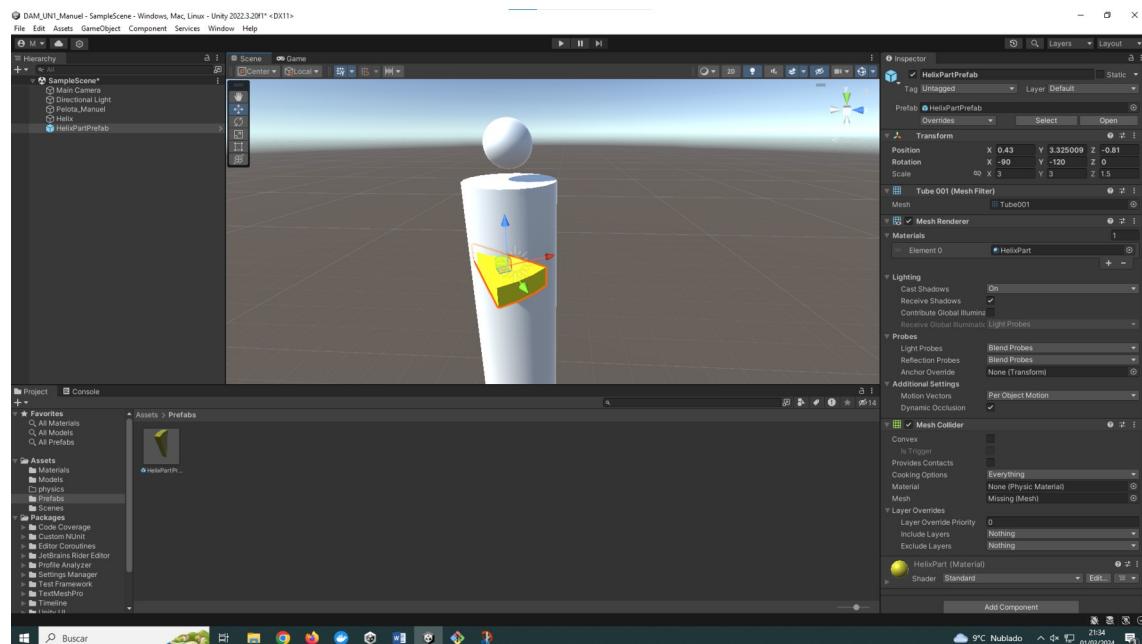
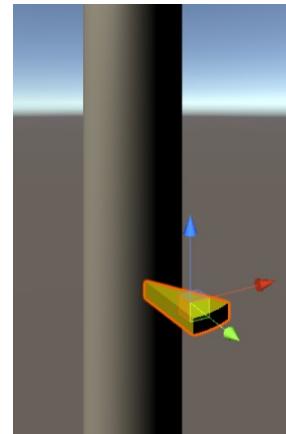
b. Le damos a import y se nos añaden carpetas.



17. En la Carpeta de Prefabs tenemos un **HellyxPartPrefab** con todos los materiales añadidos. Lo arrastramos a la escena.



**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**



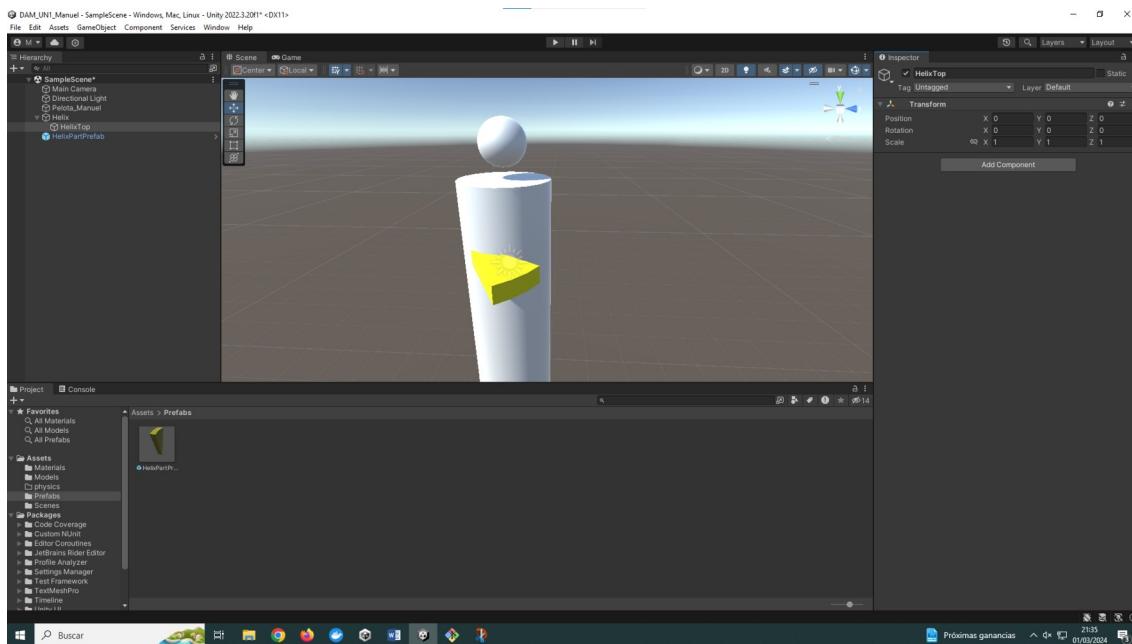
18. Sobre el cilindro Helix pulsamos click derecho a Create Empty y le llamamos **HelixTop**.



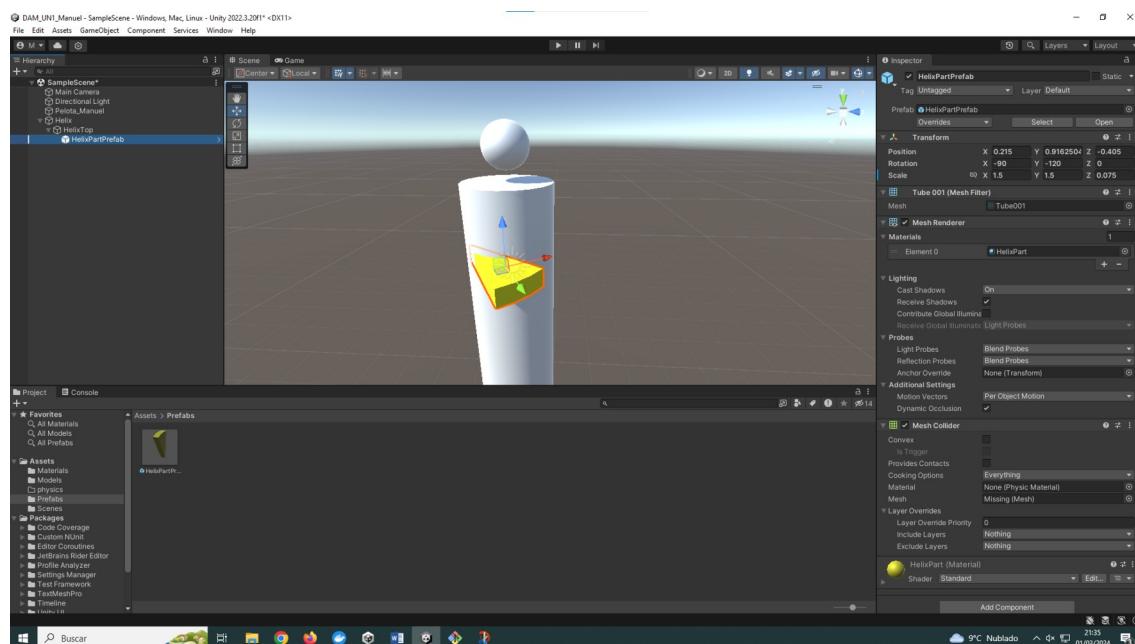
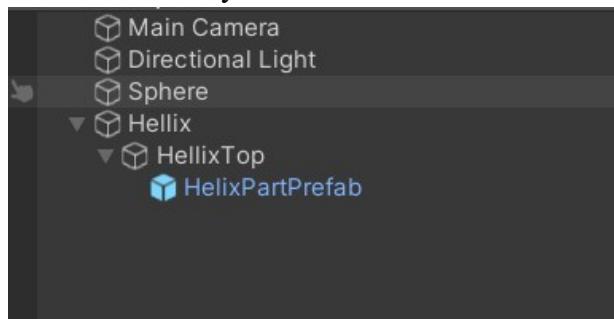
## 2º DAM

### Programación multimedia y móviles

#### U.D. 05.- Unity



- Metemos el HellyxPartPrefab dentro del HellixTop.

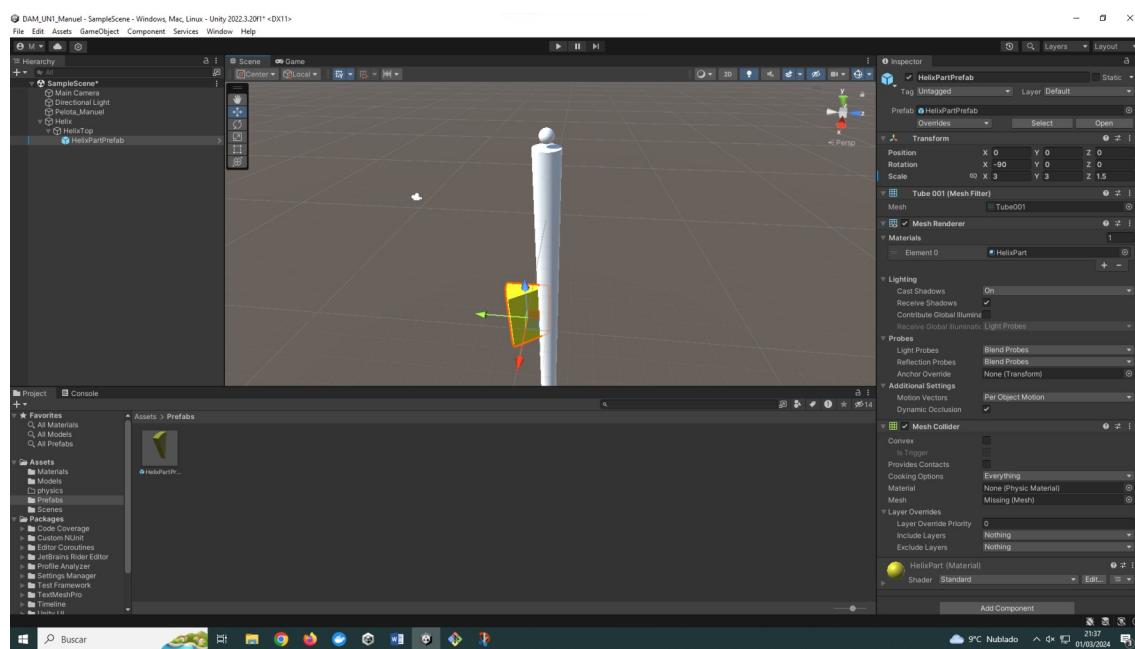




**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**

b. Del HellyxPartPrefab cambiamos:

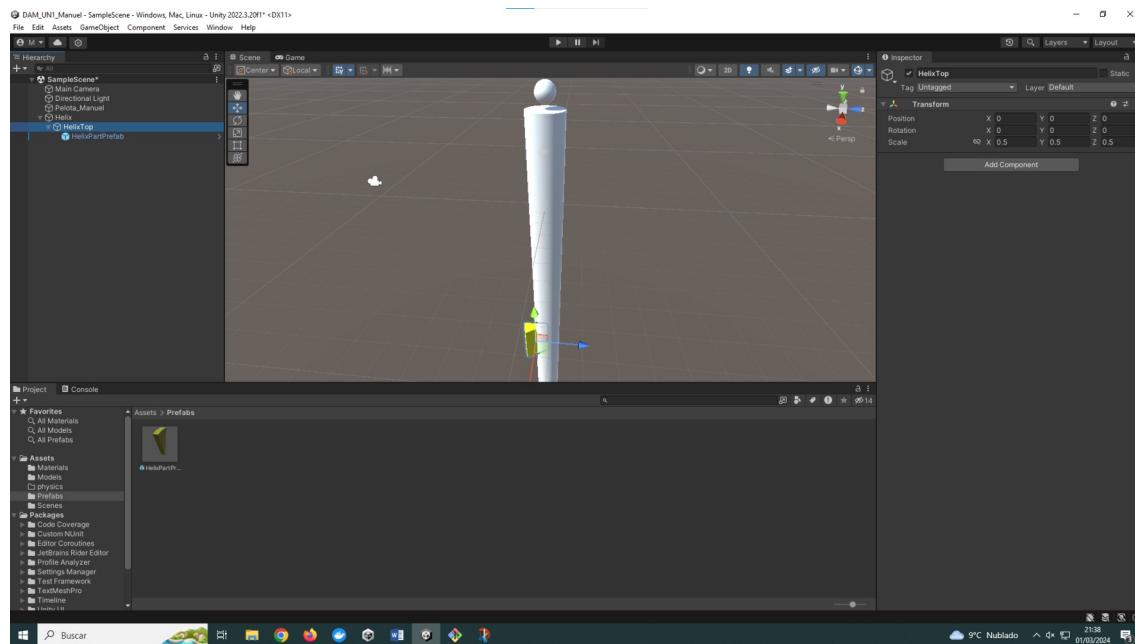
- Position: la X a 0, la Y a 0 y la Z a 0
- Rotation: la X a -90, la Y a 0 y la Z a 0
- Scale: la X a 3, la Y a 3 y la Z a 1.5



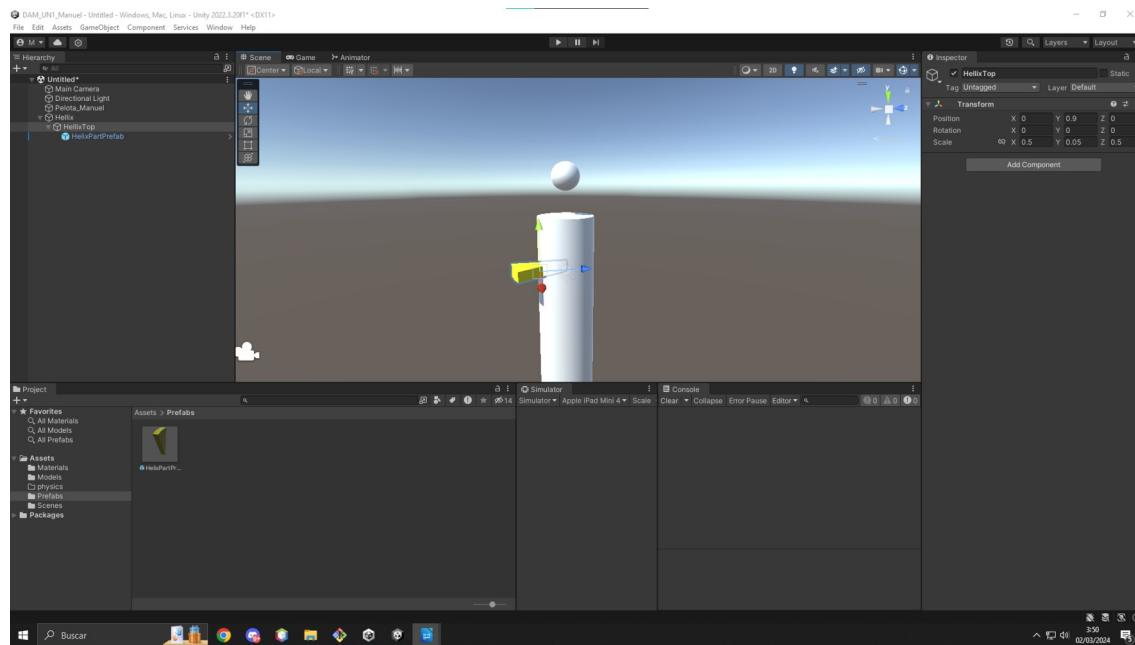
c. Así queda muy grande de alto, pero como está dentro del Hellitop, podemos cambiar en éste en Scale la X a 0.5, la Y a 0.05 y la Z a 0.5 y ya estaría bien.



**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**



- d. Para subir ahora la pieza arriba en el mismo HellixTop cambiamos en Position la Y a 0.9.



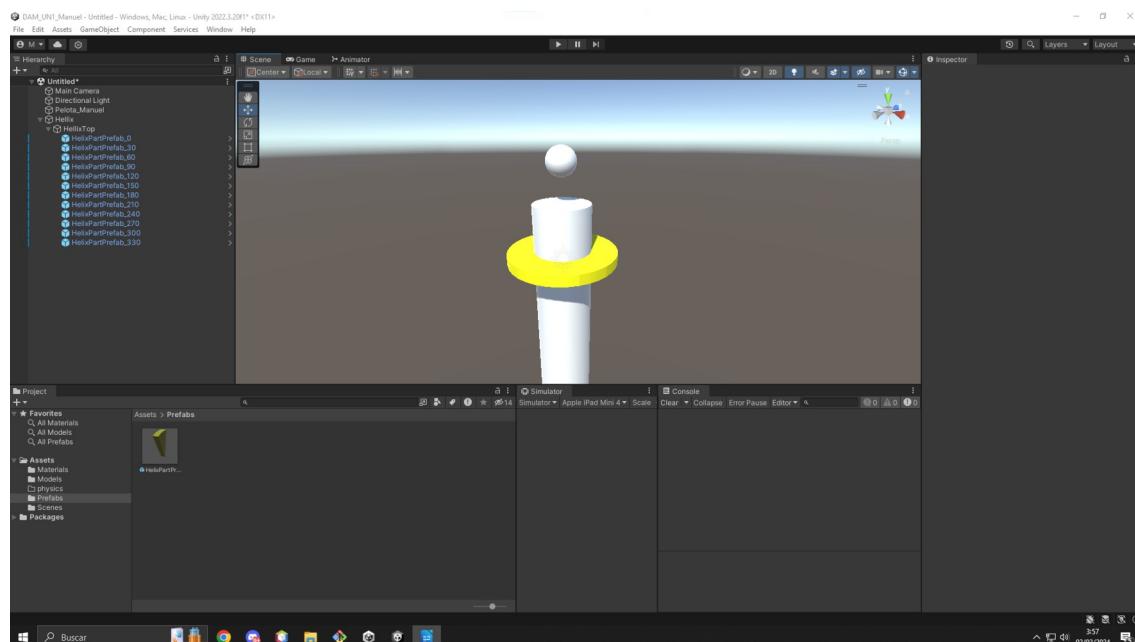
19. Ahora necesitamos crear un círculo entero de estas fichas.

- a. Para esto, hacemos Control + D sobre HellyxPartPrefab, lo que hace es copiarlo.



**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**

- b. Quiero muchas piezas de este tipo para crear el círculo, y lo que tenemos que hacer es ir poniéndolas al lado unas de otras.
- c. Para hacer esto, sólo tenemos que cambiar **la rotación de las nuevas piezas en el eje Y cada 30**.
- d. En total para completar el círculo necesitamos 12 piezas de HellyxPartPrefab en cada una iremos aumentando el eje Y en Rotation en 30.
- e. Así la primera pieza tendrá en Rotation en Y un 0, la segunda tendrá un 30, la tercera un 60, y así hasta que la pieza número 12 tendrá 330.



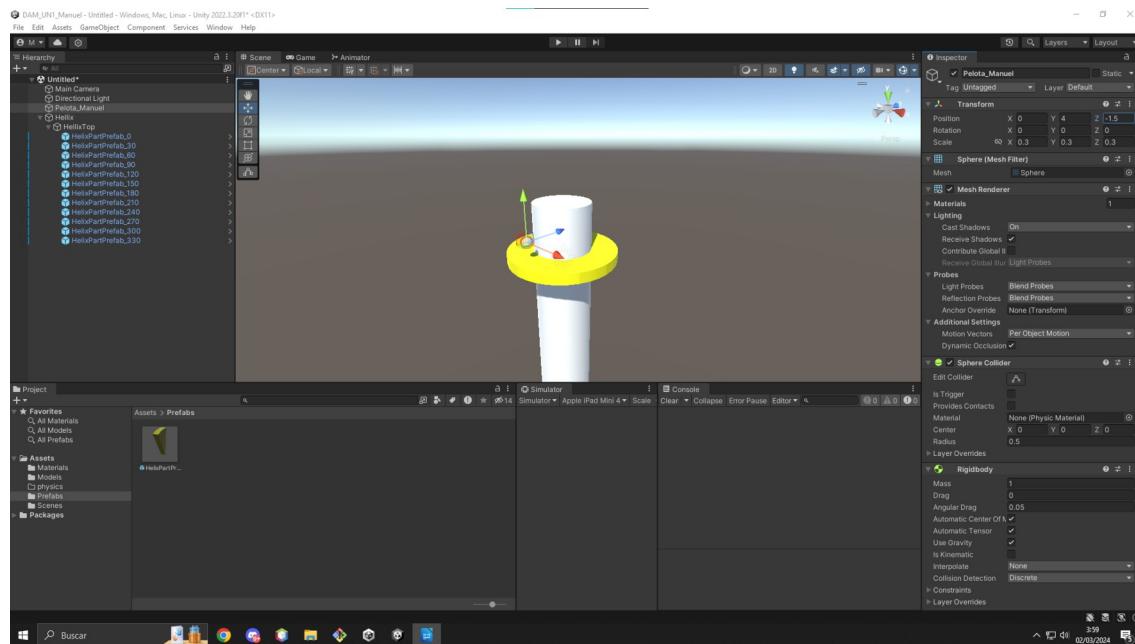
Los grados no se corresponden con los indicados, ya que van de -180 a 180, no de 0 a 360.

20. Hacemos más pequeña la bola cambiando su escala a (0.3,0.3,0.3).

- a. Ponemos la Position en (0,4,-1.5) para verla encima.

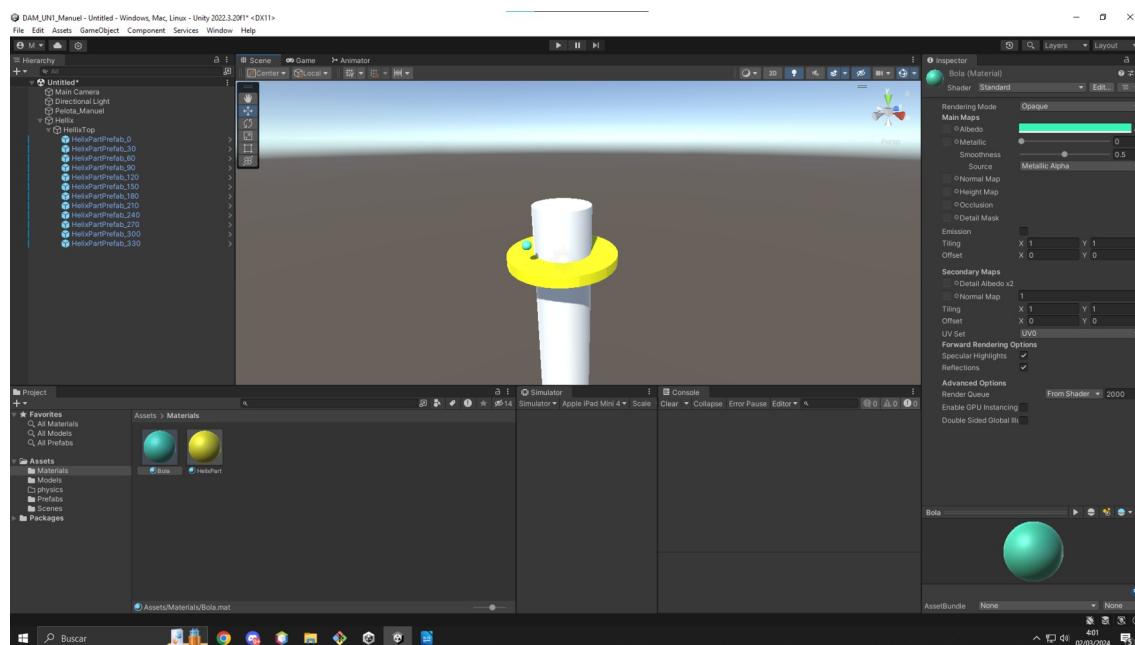


**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**



21. Vamos a crear más material.

- Dentro de la carpeta “Material”, le damos a click derecho Create -> Material, le llamamos Bola y le ponemos color azul, por ejemplo.
- Si la arrastramos a la escena donde está la bola la cambiamos.

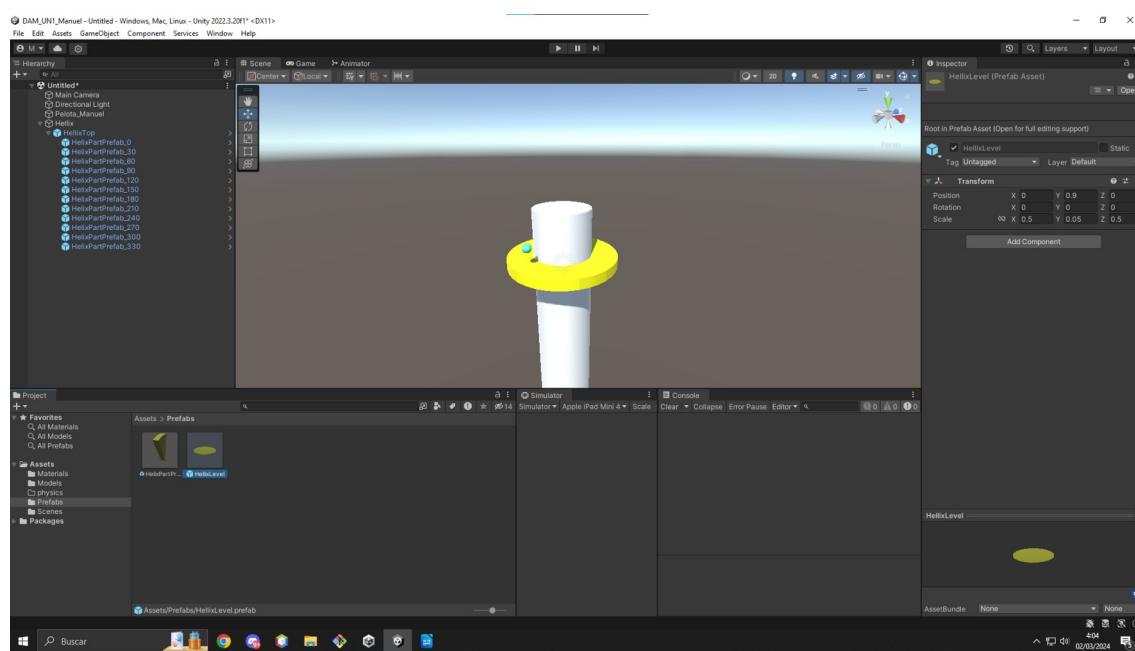
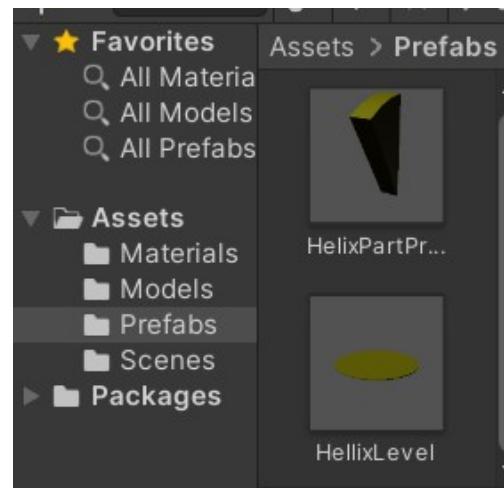




**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**

22. Tenemos que añadir ahora ese círculo completo que hemos hecho a material prefabricado, es decir, añadir el HellixTop a la carpeta Prefabs.

- Para ello, simplemente lo arrastramos.
- Una vez arrastrado, le cambiamos el nombre a HellixLevel.

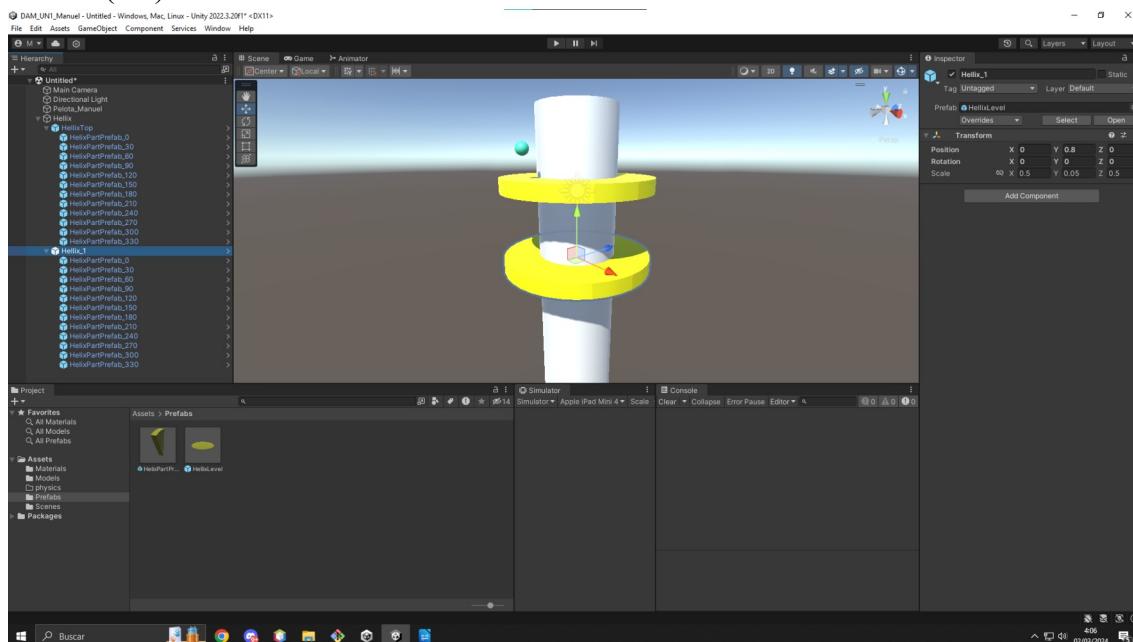


- Si vamos añadiendo ahora HellixLevel a Hellix desde la carpeta Prefabs vemos que nos adjunta el elemento entero.



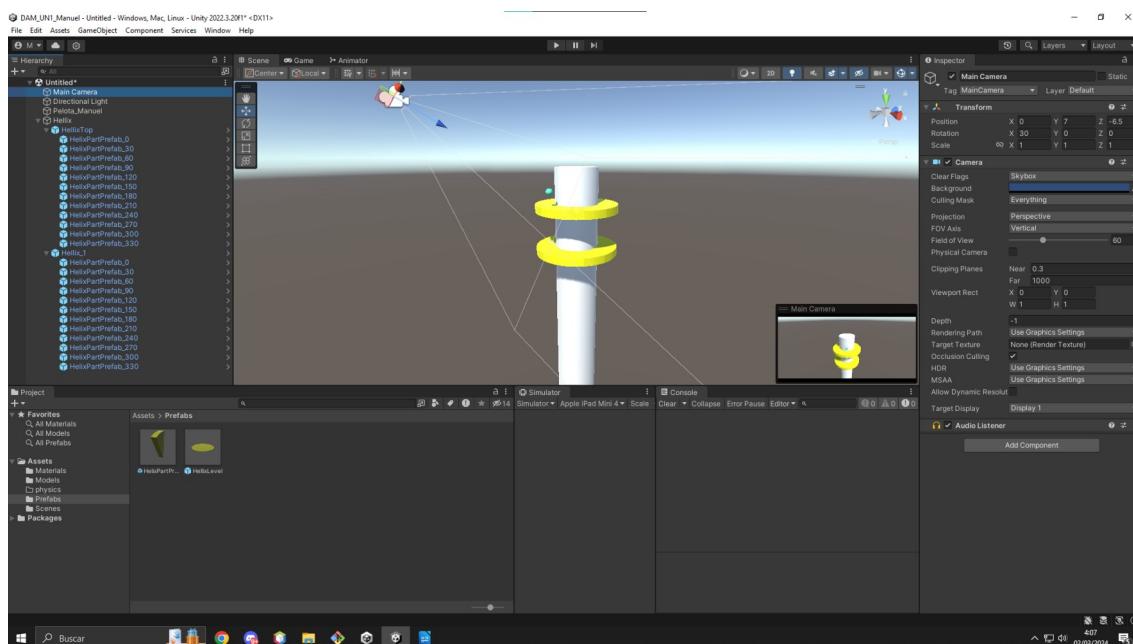
**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**

23. Copiar con Control + D el HellixTop y cambiar Y position a un poco menos (0.8).



24. Para tener una perspectiva buena, cambiamos coordenadas de Main Camera:

a. Position a (0, 7, -6.5) y Rotation a (30,0,0).



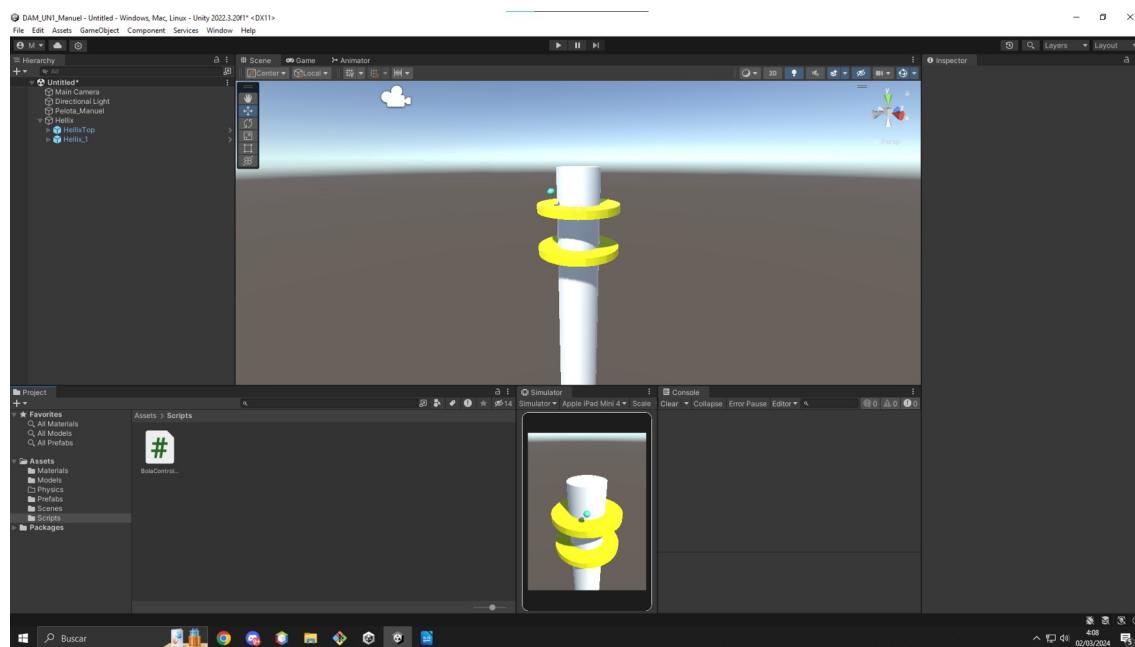
b. Podéis ir cambiándolas para mejorar la visión.



## Creación de los scripts

25. Vamos a hacer que la pelota rebote con un script.

- a. Para ello creamos en Assets una nueva carpeta que le vamos a llamar Scripts.
- b. La abrimos y creamos un C# Script al que le vamos a llamar BolaController.



- c. Lo abrimos con Visual Studio.
- d. Cambiamos el nombre de la clase a BolaController (importante que se llame igual).
- e. **Arrastramos el script al componente Bola.**
- f. Vemos que salen dos métodos:
  - i. **start** que es el que se ejecuta cuando le damos al play
  - ii. **update** que se ejecuta cada tiempo.
- g. Los borramos.

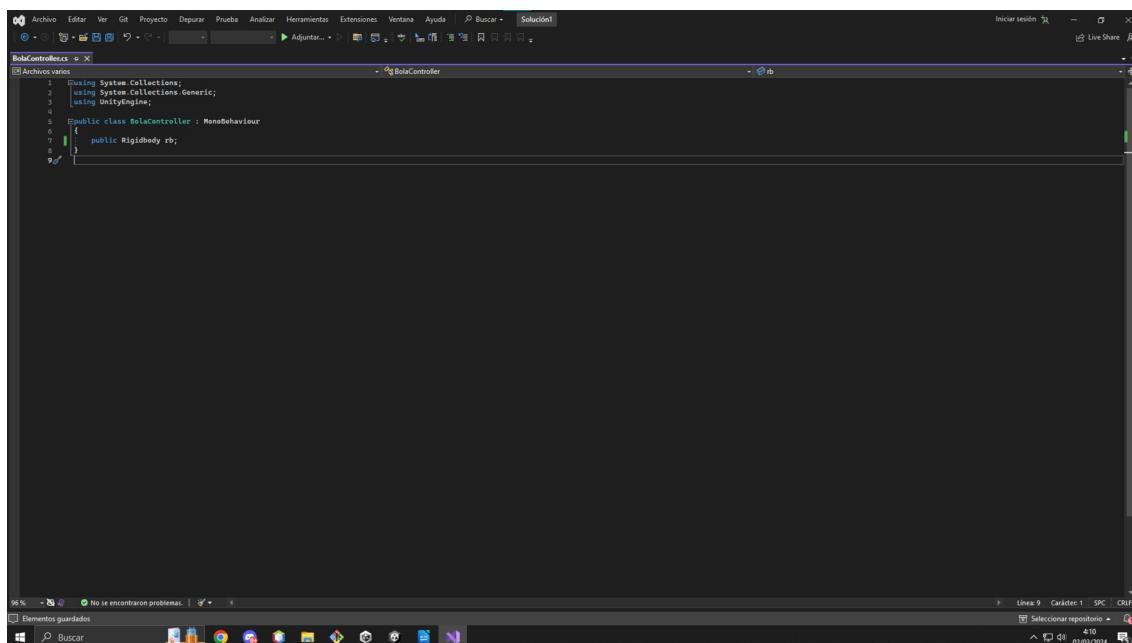


**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**

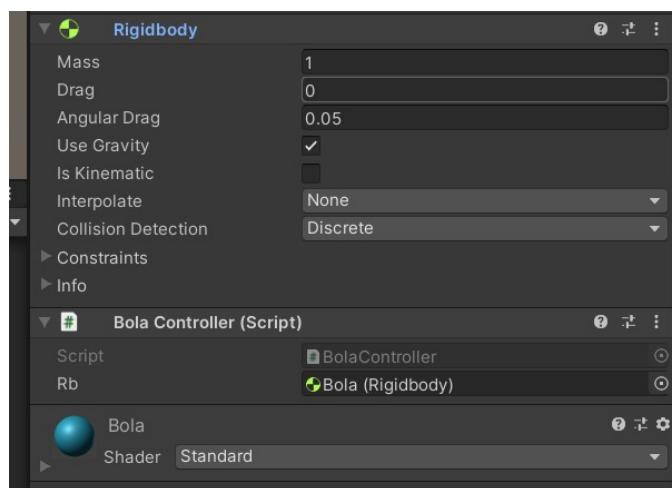
h. Tenemos que hacer lo siguiente:

- Primero una referencia al componente Rigidbody porque es al que vamos a añadir una velocidad para que la bola bote.

```
public Rigidbody rb;
```

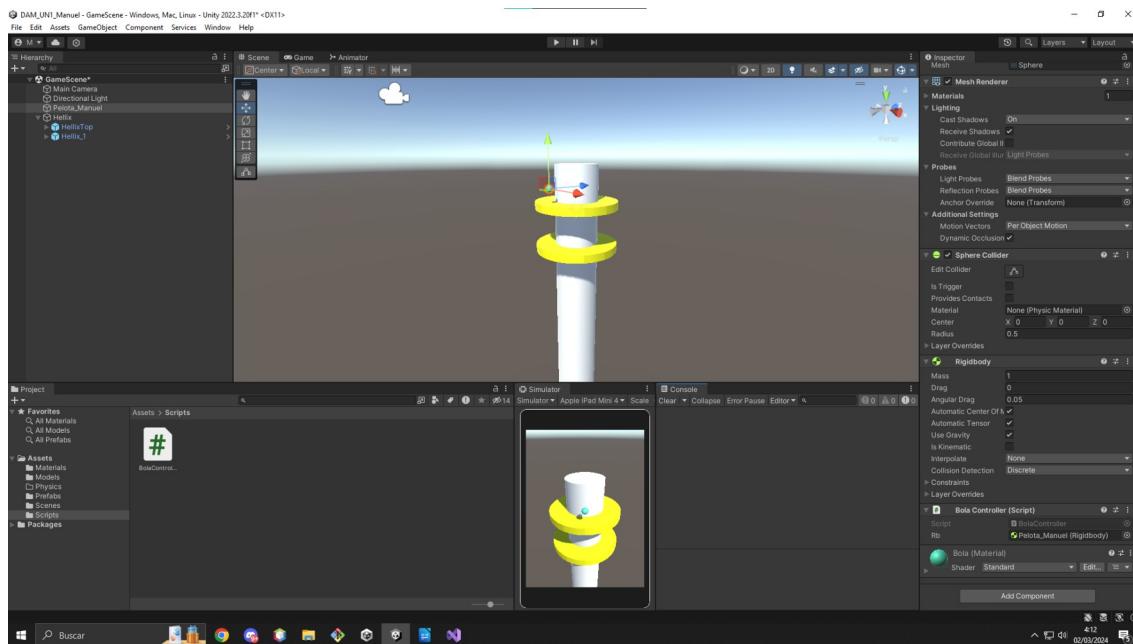


- Arrastramos ahora en Unity el componente Rigidbody al campo rb de la bola:





2º DAM  
Programación multimedia y móviles  
U.D. 05.- Unity



El código sería el siguiente:

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

public class BolaController : MonoBehaviour
{
    //Creamos un RigidBody
    public Rigidbody rb;

    //Necesitamos un impulso para controlar cuánto bota la bola
    public float fuerzaImpulso = 3f;

    //Añadimos un boolean para controlar que bote una vez sólo sobre una pieza (un quesito), no sobre dos
    //queremos que cuando bote una vez, no pueda botar otra vez hasta tantos segundos
    public bool ignorarSigienteColision;

    //¿Cuándo queremos que bote? Cuando toque el HollyTop o un HollyLevel
    //C# trae un método para esto: onCollisionEnter, cuando nuestra pelota entre en
    colisión con algo se ejecuta este método
```



```
private void OnCollisionEnter(Collision colision)
{
    //En el momento que bote una vez, salimos del método
    if (ignorarSiguienteColision)
    {
        return;
    }
    //Cuando colisionemos con algo, le añadimos velocidad de Vector3 a cero para
    evitar problemas de velocidad al colisionar
    //Vector3 son las tres coordenadas en cuanto a posicion
    rb.velocity = Vector3.zero;

    //Cuando choque queremos que vaya hacia arriba, por lo que le añadimos fuerza en
    el eje de la Y queriendo que vaya para arriba
    //lo que hemos declarado en fuerzaImpulso
    //El ForceMode Impulse añade el impulso del rigibody usando su masa
    rb.AddForce(Vector3.up*fuerzaImpulso, ForceMode.Impulse);

    //cuando ya ha botado una vez ponemos el boolean a true
    ignorarSiguienteColision = true;

    //Llamamos al método de permitirSiguienteColision() para que vuelva a botar la
    bola pasados 0.2 ms
    Invoke("PermitirSiguienteColision",0.2f);

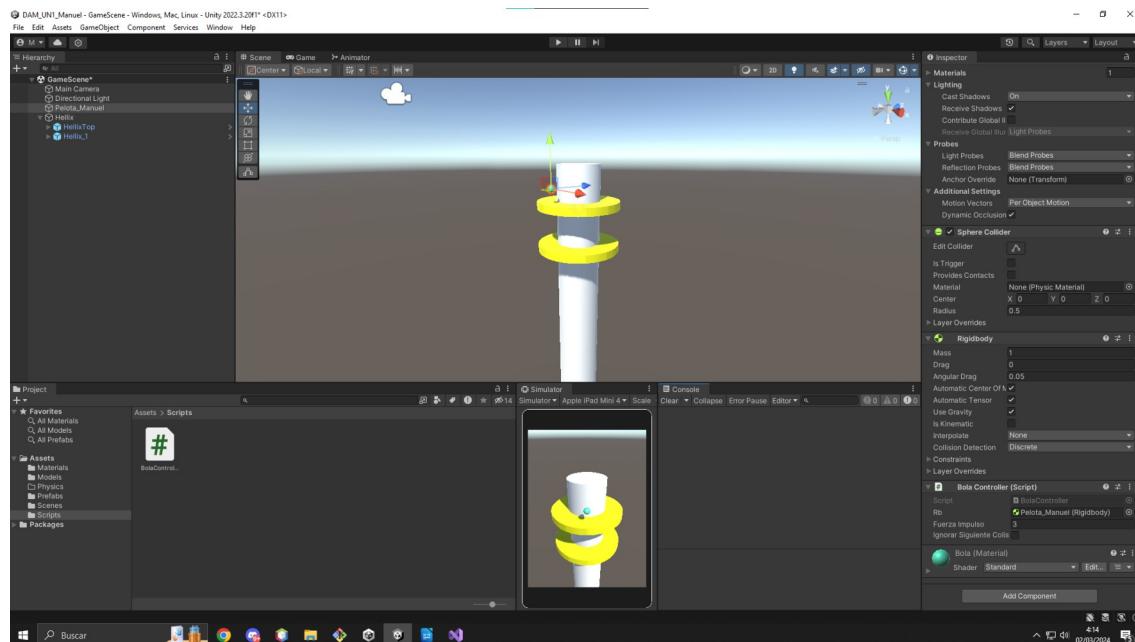
}

//Para controlar que pueda volver a botar pasados unos cuantos segundos
private void PermitirSiguienteColision()
{
    ignorarSiguienteColision = false;
}
```

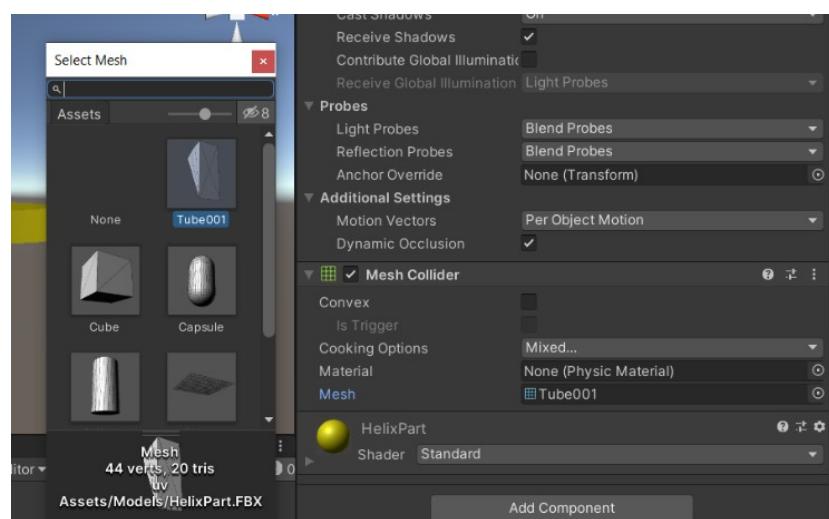
- Cuando volvemos a Unity vemos que en la Bola ya nos sale Fuerza Impulso, pudiéndolo modificar nosotros a nuestro gusto.



2º DAM  
Programación multimedia y móviles  
U.D. 05.- Unity



- Asignamos en el HellyxPartPrefab de la carpeta Prefabs el Mesh de MeshCollider a Tube001

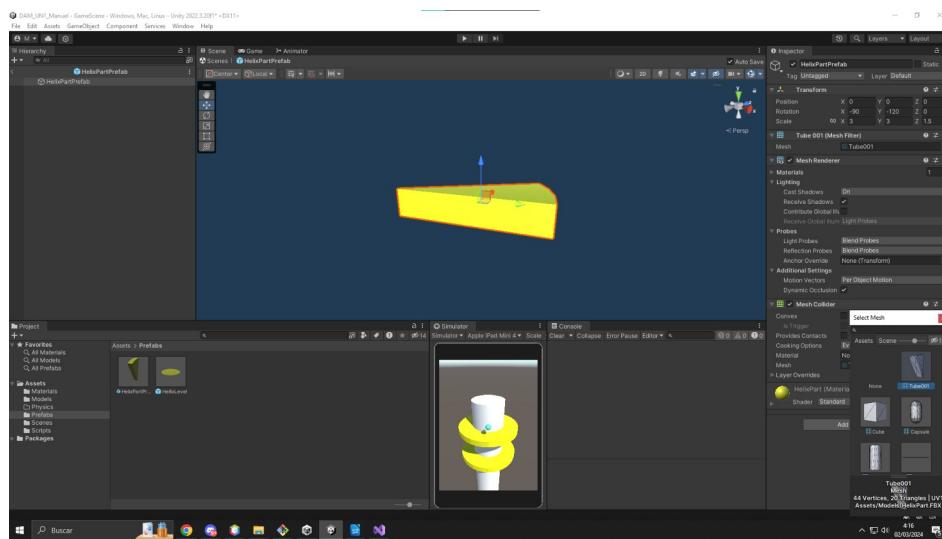




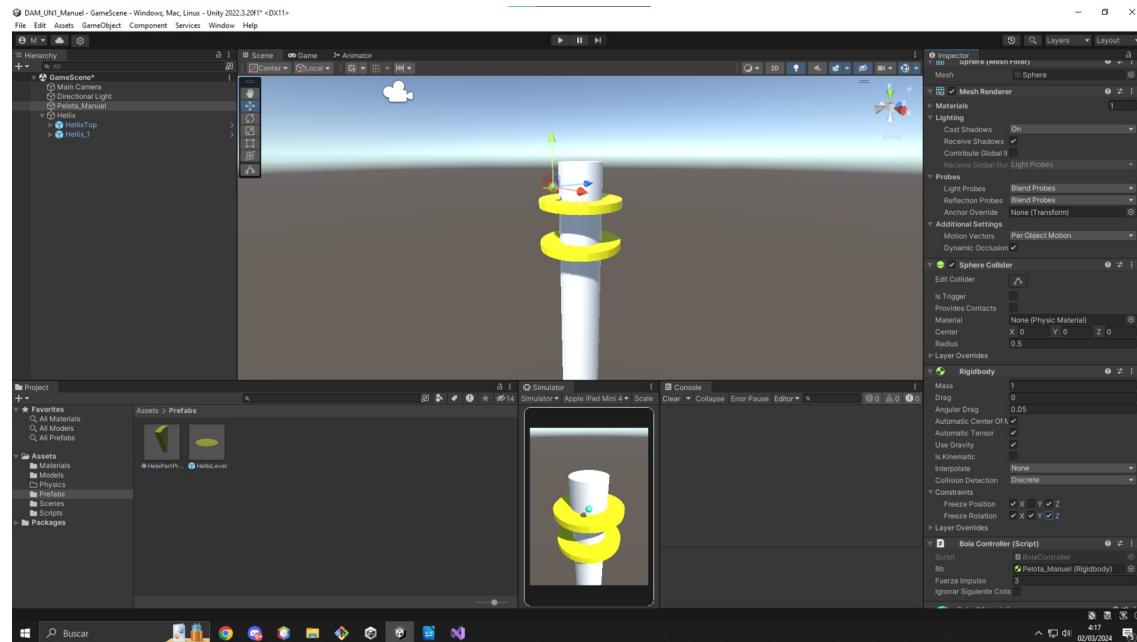
## 2º DAM

### Programación multimedia y móviles

#### U.D. 05.- Unity



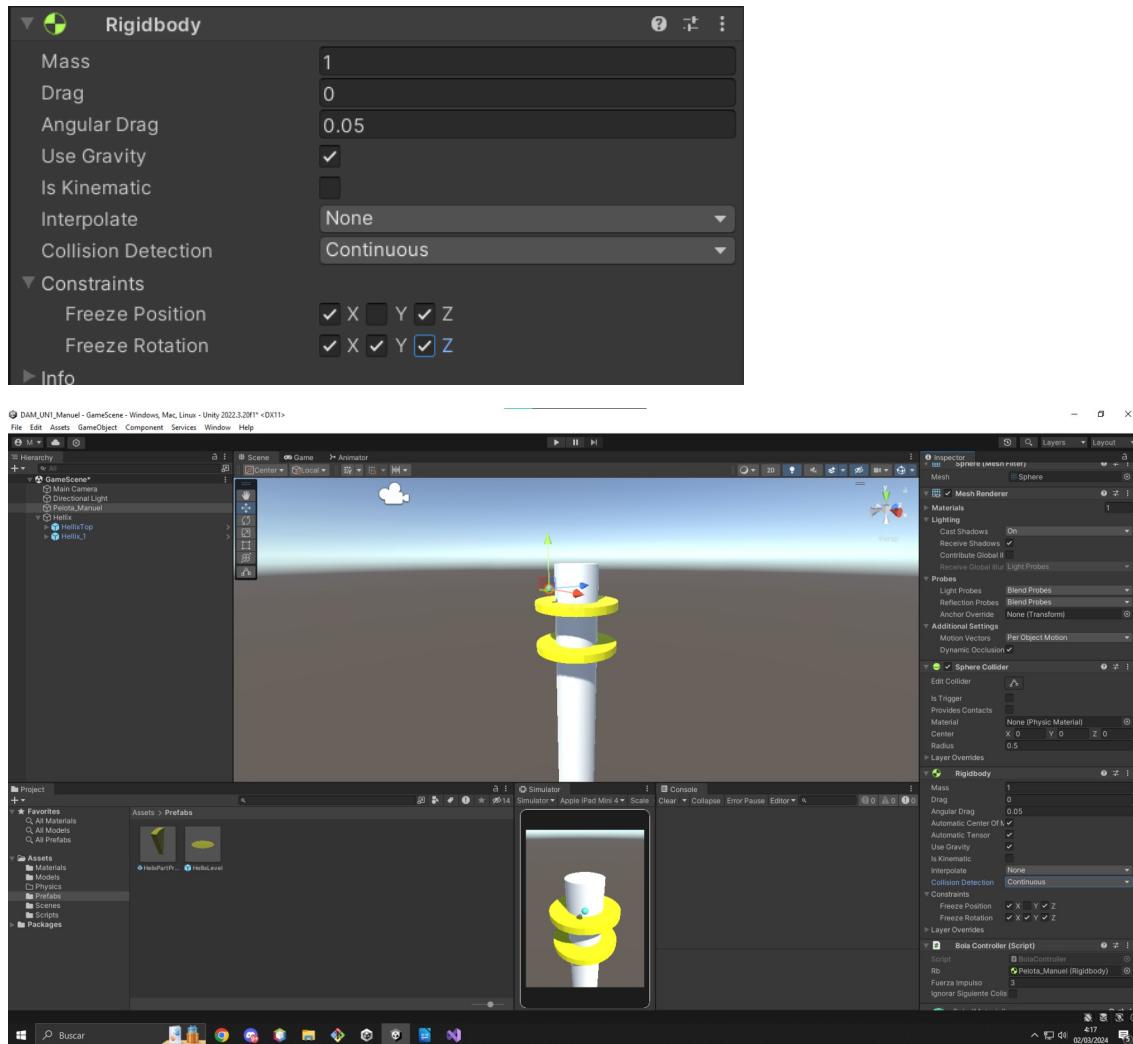
- Sólo queremos que la pelota se mueva en el eje Y y la rotación no nos hace falta
  - en el Rigidbody ponemos en Constraints como Freeze Position la X y la Z y como Freeze Rotation la X, la Y y la Z (no permitimos esas posiciones ni esas rotaciones, las bloqueamos).





**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**

- Además, para que no detecte mal los discos y las colisiones la bola, cambiamos la **Collision Detection a Continuous**:



26. Creamos un Script para Main Camera.

- a. Para ello en la carpeta Scripts creamos un C# Script llamado **CamController** y lo arrastramos al componente Main Camera.
- b. Tenemos los dos métodos creados automáticamente (start y update).
- c. Lo que queremos conseguir es que baje o suba la pelota, siempre tengamos la misma perspectiva de la imagen.
- d. Para ello tendremos que crear una referencia a BolaController, un float para mantenernos siempre a una altura de la pelota.



e. El código será el siguiente:

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

public class CamController : MonoBehaviour
{
    //Creamos una referencia a BolaController
    public BolaController bola;

    //Creamos un float que nos indique la altura a la que mantenemos de la pelota
    public float altura;

    // Start is called before the first frame update
    void Start()
    {
        //al darle al play instanciamos la altura que queremos mantener la pelota
        //Esta altura será la altura que tengo en la cámara menos la altura que tiene la
        //pelota
        altura = transform.position.y - bola.transform.position.y;

    }

    // Update is called once per frame
    void Update()
    {
        //Creamos un Vector3 con las coordenadas de mi posición actual
        Vector3 posicionActual = transform.position;

        //En todo momento, cuando se actualice, tenemos que respetar que en la
        //posición en la que me encuentre, se deje la altura establecida
        //para mantener la pelota, por lo que a la altura de la posicionActual tenemos
        //que sumarle a la altura de la pelota,
        //la altura que mantenemos sobre ésta
        posicionActual.y = bola.transform.position.y + altura;

        //Asignamos ahora que nuestra altura sea ésta
        transform.position = posicionActual;
```



**2º DAM**  
**Programación multimedia y móviles**  
**U.D. 05.- Unity**

```
}
```

```
}
```

**Arrastramos el componente Bola al campo Bola al Cam Controller de Main Camera:**

