# Unity GameObjects

Son la esencia del motor. Todo son GameObjects con distintas propiedades.

Tags: Etiquetas que sirven para identificar un Objeto y luego llamarlo desde Scripts mediante esa etiqueta. Ej: GameObject.FindGameObjectWithTag(“Player”)

Layers: Capas que sirven para agrupar objetos. Ej: LayerMask.GetMask(“LayerName”)

**Obtener un componente:**

Si es del mismo objeto que tiene el script: GetComponent<Componente> ()

GetComponent<AudioSource>()

Si es de otro objeto: GameObject. GetComponent<Componente> ()

Si se quiere obtener un componente del objeto padre:

GetComponentInParent<Componente>()

Transform.parent.GetComponent<Componente>()

**Llamadas a otro Script:**

Cualquier script se puede obtener con la función GetComponent<Nombre del Script>(). Con esto se accede a cualquier variable y función pública.

Métodos de llamar a otro Script:

* Crear una variable publica y arrastrar el componente desde el View: public PlayerHealth playerHealth; playerHealth.currentHealth.
* Si se crea una variable public static podrá ser accesible mediante ScriptName.variable
* Buscar el objeto por su etiqueta. Despues con el GetComponent se podría obtener el componente Script. Sería: GameObject.FindGameObjectWithTag(“Tag”).GetComponent<Nombre del Script>()

Destoy(Objeto, float): Sirve para destruir un GameObject o un componente de la escena u objeto y el float es para pasar un tiempo de espera hasta que se destruye. Aunque en el código sea lo primero, el destroy siempre se ejecuta lo último en la escena.

Destory(GetComponent<MeshRenderer>())

Transform:

Tiene la posición en los ejes x, y, z

Tiene la rotación en el espacio.

Tiene la Escala.

Metodos:

SetPositionAndRotation: Sirve para establecer la posición y rotación en una sola instrucción.

transform.SetPositionAndRotation(posición, rotación)

Translate(Vector): Mueve el Objeto en la dirección y distancia que apunta el Vector.

transform.Translate(Vector3.forward)

Rotate(Vector, float, GameObject): Rota un objeto en el angulo que se le pase en el Vector, los grados que se le pasen en el float y sobre el gameObject relativo. Este ultimo si no se pasa, por defecto es el objeto mismo.

transform.Rotate(Vector.up, 5.0f)

LookAt(transform, Vector):Sirve para rotar un GameObject apuntando a la direccion del transform pasado por parámetro, además como opcional se puede pasar un segundo parámetro para indicar sobre que eje va a rotar.

transform.LookAt(Cubo.transform)

SetParent(transform): Sirve para asociar un GameObject Padre al GameObject. Si se le quiere dejar sin padre se le pasa null como parámetro.

transform.SetParent(Cubo.transform)