# Unity Corrutinas

Nos permiten ejecutar métodos y controlar su ejecución de manera que podemos controlar el flujo del código. Ademas la ejecución se ejecuta de manera independiente. Por ejemplo si usamos un for, este se ejecutara entero pero si lo ponemos en un corrutina podemos controlar que cada ejecución de este se ejecute en un frame, por ejemplo. Equivaldria a un Hilo/Thread. Dentro de los métodos podemos decirle que se detenga hasta el siguiente fotograma.

En el Start, llamamos a la Corrutina con:

StartCoroutine(metodo()) // Se le puede pasar el nombre de la corrutina como parametro string en lugar del método, pero no permite pasar parámetros.

Además se puede llamar en el OnEnable y desde otra Corrutina. Si es desde otra, será una corrutina asíncrona que quiere decir que son independientes una de otra.

El método tiene que ser de tipo IEnumerator y se le pueden pasar parametros:

IEnumerator método(){

… codigo

yield return null

}

yield return es el que controla las ejecuciones, indica cuando termina la ejecución de la corrutina por frame. Después se reanuda en ese mismo punto.

Para llamar a una Corrutina de manera sincrona dentro de otra. Sicrona quiere decir que hasta que no acabe la ejecución de la corrutina que llamamos, no sigue la principal. Se usaría:

yield return StartCoroutine(método())

Para parar la ejecución un tiempo determinado:

yield return new WaitForSeconds(segundos)

Parar hasta el proximo FixedUpdate:

yield return new WaitForFixedUpdate()

Parar hasta el final del fotograma:

yield return new WaitForEndOfFrame ()

Para parar todas las ejecuciones de corutinas dentro de un componente, se usa la función:

StopAllCoroutines()

Para salir de una ejecución de una corrutina se puede usar:

yield break

Detener una corrutina. Se le puede pasar el nombre de la corrutina como parametro string:

StopCoroutine(Corrutina)