**Universidad Nacional Autónoma de México**

**Facultad de Ingeniería**

Fundamentos de programación

Actividad Asíncrona #11

Alumno: Camacho Garduño Miguel Angel

Grupo: 3

Fecha: 27/11/2020

Actividad Asíncrona 15

Estructuras de control

Para mi proyecto que es realizar un videojuego de disparos en primera persona, decidí usar una plantilla de Unity llamada [FPS Microgame](https://learn.unity.com/project/fps-template) y en ella la estoy modificando para agregarle más objetivos y enemigos

# Actor.cs

Lo ocupo para que, si un enemigo es eliminado, este desaparece del mapa.

# ChargedWeaponEffectsHandler.cs

Lo utilizo para que, si un arma por ejemplo el arma de rayos se calienta, esta muestra una animación mostrando que se enfría y vuelve a su forma original

# Damageable.cs

Lo utilizo para que se multiplique el daño si es que un enemigo ataca al jugador, se cancela si el daño se lo hace el mismo jugador

# DisplayMessage.cs

Lo usé para determinar internamente el tiempo en que tarda en aparecer el mensaje y que se muestre el mensaje

# EnemyMaganer.cs

En este archivo estoy ocupando el if para que al eliminar un enemigo me aparezca una notificación de enemigos restantes en el mapa

# EnemyMobile.cs

Lo utilizo para que, si el dron enemigo ve al jugador, lo ataque y siga al jugador done este vaya.

# EnemyTurret.cs

Lo utilizo para que la torreta enemiga ataque al jugado cuando ésta lo ve, en cambio no ataca si no ve al jugador

# GameFlowManager.cs

Usé if-else para determinar primero cuando tiempo tarda en carga la escena final, si no, determinar si los objetivos se cumplieron o el jugador está muerto

# Health.cs

Lo utilizo para que, si el jugador recibe daño por un enemigo o por caer de una gran altura, vaya disminuyendo su barra de salud, y en caso de que muera de la forma que sea, muestre el game over.

# HealthPickup.cs

Lo ocupo para que, si el jugador recoge salud, esta desaparece y el jugador tiene su salud restaurada

# Jetpack.cs

Lo utilicé para que, si el jugador está en el suelo, no haga nada el jetpack, en cambio si el jugador da un doble salto, el jetpack empieza a llevar al jugador a donde este quiera

# ObjectiveKillEnemies.cs

Al igual que [EnemyManager.cs](#_EnemyMaganer.cs), lo utilicé para determinar cuántos enemigos quedan en el mapa para completar el objetivo que consiste en eliminar todos los enemigos del mapa

# ObjectiveManager.cs

Lo utilicé para detectar si se han completado los objetivos, en caso de que no continua el ciclo y regresa un false

# Pickup.cs

Lo utilizo para que, si el jugador recoge un arma, un jetpack o salud, se genere sonido de recogido.

# PlayerCharacterController.cs

En este archivo lo utilicé para que al morir cayendo al vacío, produzca un sonido de muerte, en cambio si cae en una superficie plana se produce el sonido que hace al tocar el suelo al caer de cierta altura

# PlayerInputHandler.cs

Lo ocupo para poner la velocidad máxima a la que puede ir el jugador como definir los controles básicos

# PlayerWeaponsManager.cs

Lo utilizo para que, al disparar mucho un arma, esta tenga un retroceso.

# WeaponPickup.cs

Lo ocupo para que, si el jugado recoge una nueva arma, va a tener la habilidad de cambiar un arma de otra, y se va a eliminar el arma del suelo si el jugado la recoge