Ciclo: Animaciones 3D, Juegos y Entornos Interactivos **Curso:** 2020/21

Módulo: Desarrollo de Entornos Interactivos Multidispositivo

Nombre y apellidos: Mónica Valverde Calvo

EXAMEN TEÓRICO – 1ª EV **Grupo B**

Escribe tu nombre y apellidos en la cabecera de este documento, y a continuación explica qué herramientas vistas a lo largo del curso, tanto de Unity como de programación, utilizarías para lograr los objetivos planteados en el ejercicio práctico

IMPORTANTE: no te limites a enumerarlas, explica por qué usarías esas y no otras:

Cada apartado valdrá 2 puntos, y la nota final representará el 25% de la nota final de la evaluación.

Interactividad y gestión del movimiento

Haría un Script con un código en el que se ponga el movimiento que hace la nave tanto en horizontal, vertical como en profundidad.

transform.position = new Vector3(0,0,0);

con este código podríamos mover dicho objeto en los tres ejes x,y y z.

Seguimiento del jugador con la cámara:

Para que la cámara siga al objecto hay varias formas:

```
//Con esta línea, nuestro objeto tendrá la misma posición que el jugador
transform.position = playerPosition.position;
```

//Con esta línea, la cámara mantiene su posición en X y Z, pero sigue al jugador en Y //Útil para juegos de plataformas

transform.position = new Vector3(transform.position.x, playerPosition.position.y, transform.-position.z);

//Con este código, la cámara seguirá al jugador, pero alejado algo en el eje Z

transform.position = new Vector3(playerPosition.position.x, playerPosition.position.y, player-Position.position.z – 10);

//Con este código, conseguimos que siga al objeto pero con suavidad //La velocidad de suavizado, cuanto menor sea más brusco será el movimiento

Vector3 targetPosition = new Vector3(transform.position.x, playerPosition.position.y, transform.position.z); transform.position = Vector3.SmoothDamp(transform.position, targetPosition, ref camaraVelocity, smoothVelocity);

//Opcional: que la cámara mire al objeto (la cámara no puede ser ortográfica)

transform.LookAt(playerPosition);

Creación de elementos (enemigos) de forma aleatoria y a intervalos

Con Ramdon.rage podemos crear enemigos de forma aleatoria ponemos cuantos aparecen nada mas empezar, la velocidad a la que aparecen, si uno desaparece que aparezca otro también que se creen nuevos enemigos mas rápido según avanza el juego y programar cada cuanto queremos que salgan con un Corrutina.

User Interface (tiempo transcurrido, nº de columnas y alerta)

Se crea un canvas para uno de los tres.

El primer canvas se programa con una cuenta para en segundos.

El numero de columnas deberá ir asociado a la creación de columnas, con una comunicación entre scripts, para saber cuantas se están creando y que ese dato lo remita al canvas, con un return.

La alerta ira asociado al script de la nave y delimitado en sus movimientos para cuando incumpla lo programada salte el texto canva, igual que el anterior se hará una comunicación entre scripts.

Colisiones

Creamos un mesh collider en el objeto para delimitarlo y ver en que punto el objeto se destruiría o se pararía el juego.

Cambien le ponemos un Rigibody para establecer la velocidad.

Con el método On Trigger detectamos la colisión, antes tendremos que etiquetar nuestro objeto enemigo para que nuestro "nave lo detecte.

Entrega

Cuando tengas completo el documento, expórtalo a pdf con este formato:

Apellidos_nombre_ExTco1EV.pdf

Guárdalo dentro del repositorio, y súbelo en un *commit* de GitHub, el cual se acompañará al *Pull Request* del final del examen.