ESAT INNOVATION SCHOOL | ESCUELA SUPERIOR DE ARTE Y TECNOLOGÍA

Práctica - Motores de Videojuegos
MEMORIA

Nombre Asignatura: Motores de Videojuegos
Módulo Asociado: 1PVG
Profesor:Juan Diego Alegre
Curso: 2022/2023
Autor/es: Sergio Madaleno Figueroa

Índice/Index

1.- Contexto del juego

El juego que he creado es un shooter en tercera persona en el que el jugador tiene que ir eliminando a los distintos enemigos. Los enemigos también pueden ir bajando la vida del jugador hasta que lo eliminen por completo.

El primer nivel tiene un estilo inspirado en Crossy Road, donde el jugador tiene que ir esquivando los distintos vehículos que aparecen a lo largo del nivel. Si el jugador colisiona con un vehículo, el jugador reaparece en el sitio principal y se le quitará algo de vida.

En el segundo nivel, he decidido crear un parkour en el que el jugador deberá ir saltando hasta llegar a una llave que al recogerla permitirá abrir una puerta. Si el jugador cae al suelo, se le quitará algo de vida y puede volver a subir por una rampa para volver a intentarlo.

Por último, he agregado una bandera del videojuego de Mario Bros como en el que si el jugador colisiona con la bandera, el menú de inicio será mostrado para poder volver a jugar una nueva partida.

También está la opción de pausa en el que si se presiona la tecla ‘P’, la partida será pausada.

2.- Mecánicas/acciones que se pueden realizar

2.1-Movimiento

2.1.1 Movimiento del jugador

He creado un Vector3 utilizando los valores de entrada horizontales y verticales. Multiplicando este vector por moveSpeed y Time.deltaTime, se escala el vector de movimiento en función de la velocidad y el tiempo transcurrido.

Finalmente, se actualiza transform.position. Esto aplica el vector de movimiento transformado en relación con la rotación del jugador, lo que asegura que el movimiento se produzca en la dirección correcta en el mundo del juego.

2.1.2 Salto del jugador

Se realiza un raycast hacia abajo desde la posición del jugador. Si el rayo golpea el suelo a una distancia menor o igual a la distancia de verificación del suelo, el jugador está en contacto con el suelo y la variable "grounded" se establece como verdadera. Si el rayo no golpea el suelo, la variable "grounded" se establece como falsa. La distancia para verificar está algo por encima de 0 para que pueda ser detectado.

2.1.3 Rotar al jugador

Se lee el movimiento del ratón en el eje X utilizando la función Input.GetAxis("Mouse X") y lo multiplica por la velocidad de rotación turnSpeed_. Esto determina la cantidad de rotación que se aplicará al jugador.

Luego, se actualiza la rotación del objeto en el eje Y utilizando la función Player.transform.eulerAngles. Se crea un nuevo Vector3 para especificar la rotación, manteniendo los valores en los otros ejes (X y Z) en 0 y sumando el valor calculado anteriormente al ángulo actual en el eje Y del objeto.

2.1.4 Enemigo mira al jugador

Este código se encarga de que un enemigo esté siempre mirando al jugador.

En cada ejecución, el enemigo ajusta su orientación utilizando la función LookAt. Esta función hace que el enemigo gire automáticamente para apuntar hacia la posición actual del jugador.

2.2-Disparo

2.2.1 Disparo del jugador

En el método Start, se obtiene la referencia al componente Transform del cañón y se guarda en la variable TR_.

En el método Update, se verifica si se ha presionado el botón de disparo (clic izquierdo del ratón). Cuando se detecta el botón de disparo, se ejecuta el método Fire().

En el método Fire, se crea una instancia de la bola de cañón utilizando el prefab definido anteriormente. Se establece la posición y rotación de la bola de cañón para que coincida con la posición y orientación del cañón.

A continuación, se obtiene el componente Rigidbody de la bola de cañón y se le aplica una fuerza utilizando el método AddForce. La dirección de la fuerza es hacia adelante multiplicada por el valor de la variable impulse_. Se utiliza el modo de fuerza ForceMode.Impulse para aplicar la fuerza de manera instantánea.

Cuando la bala colisiona con alguna pared con el tag de 'Wall', el objeto de la bala es destruido.

2.2.1 Disparo del enemigo

Lo único que se diferencia al código del jugador es que el disparo enemigo se ejecuta cada 7 segundos.

2.3-Coger item

Cuando el jugador colisiona con el objeto de una llave, la llave se destruye y la puerta se abre.

2.4-UI/HUD

En la parte del HUD he implementado una barra de vida para el jugador que irá decrementando si hay una colisión entre una bala y el jugador. Si la vida llega a 0, la escena del menú principal se enseña para empezar una nueva partida.

2.- ¿Cómo se juega?

Los controles necesarios para jugar son los siguientes:

- Para poder mover al jugador se utilizan las teclas W,A,S,D
- Para poder saltar se utiliza la tecla de espacio
- Para poder disparar se utiliza el clic izquierdo del ratón
- Para poder pausar el juego se utiliza la tecla 'P'

3.- Bibliografía

Unity API:

- https://docs.unity3d.com/Manual/

Unity Asset Store:

- https://assetstore.unity.com/