Video juegos

## I. Introducción

En el presente informe se describe el desarrollo de un juego llamado “cubeRoll” en Unity utilizando el lenguaje de programación C#. El juego consiste en controlar un cubo que se mueve en un entorno 3D y debe evitar obstáculos moviéndose a la izquierda y a la derecha. El objetivo del juego es alcanzar el final del nivel sin chocar con los obstáculos para completarlo.

## II. Descripción del Juego

El juego está basado en un entorno 3D y presenta una cámara en tercera persona que sigue al cubo controlado por el jugador. El cubo se mueve hacia adelante de forma automática, y el jugador puede desplazarlo hacia la izquierda y hacia la derecha para evitar los obstáculos.

## III. Mecánicas del Juego

Las mecánicas principales del juego son las siguientes:

1. Movimiento del Jugador: El cubo se desplaza en el eje Z hacia adelante de forma constante mediante la aplicación de una fuerza. El jugador puede controlar el movimiento del cubo presionando las teclas "a" y "d" para moverse hacia la izquierda y hacia la derecha, respectivamente.

```
using UnityEngine;

public class PlayerMovement : MonoBehaviour
{
    public Rigidbody rb;
    public int health = 10;
    public float forwardForce = 2000f;
    public float sidewaysForce = 500f;
    void FixedUpdate()
    {
        rb.AddForce(0, 0, forwardForce * Time.deltaTime);

        if(Input.GetKey("d"))
        {
            rb.AddForce(sidewaysForce * Time.deltaTime, 0,0, ForceMode.VelocityChange);
        }
        if (Input.GetKey("a"))
        {
            rb.AddForce(-sidewaysForce * Time.deltaTime, 0, 0,ForceMode.VelocityChange);
        }
        if (rb.position.y <-1f)
        {
            FindObjectOfType<GameManager>().EndGame();
        }
    }
}
```

2. Obstáculos: Se han implementado obstáculos en movimiento que aparecen a lo largo del nivel. Estos obstáculos pueden colisionar con el jugador y, en caso de ocurrir una colisión, el juego termina y se reinicia.

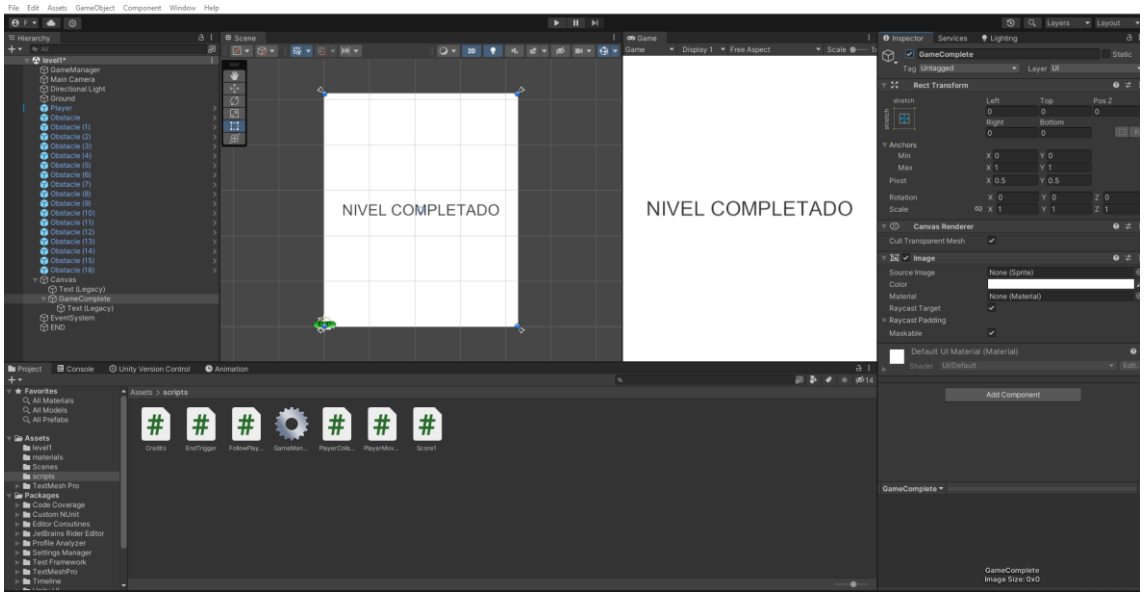
```
using UnityEngine;

public class PlayerColission : MonoBehaviour
{
    public PlayerMovement movement;
    void OnCollisionEnter(Collision collisionInfo)
    {
        if(collisionInfo.collider.tag  == "Obstacle")
        {
            movement.enabled = false;
            FindObjectOfType<GameManager>().EndGame();
        }
    }
}
```

```
using UnityEngine;
using UnityEngine.SceneManagement;
public class GameManager : MonoBehaviour
{
    bool gamehasEnded = false;
    public float restartDelay = 1f;
    public GameObject completeLevelUI;

    public void CompleteLevel()
    {
        completeLevelUI.SetActive(true);
        Invoke("Restart", restartDelay);

    }
    public void EndGame()
    {
        if(gamehasEnded == false)
        {
            gamehasEnded = true;
            Invoke("Restart", restartDelay);
        }
    }

    void Restart()
    {
        SceneManager.LoadScene(SceneManager.GetActiveScene().name);
    }


}
```

3. Puntuación: La puntuación del jugador se basa en la posición en la que se encuentra el cubo en el eje Z. A medida que el jugador avanza en el nivel, la puntuación aumenta. Esta puntuación se muestra en pantalla y proporciona un incentivo para que el jugador intente llegar lo más lejos posible en el juego.

```
using UnityEngine;

public class FollowPlayer : MonoBehaviour
{
    public Transform player;
    public Vector3 offset;
    void Update()
    {
        transform.position = player.position + offset;
    }
}
```

4. Final del Nivel: Al llegar al final del nivel sin colisionar con ningún obstáculo, el jugador completa el nivel y se muestra un mensaje de "Nivel completado". Tras esto, se reinicia el juego para seguir jugando.




IV. Implementación del Juego

El juego ha sido desarrollado utilizando Unity y el lenguaje de programación C#. A continuación, se presenta una descripción de los scripts principales utilizados en el juego:

1. PlayerMovement.cs: Este script controla el movimiento del jugador. Se utiliza un componente Rigidbody para aplicar fuerzas físicas al cubo. El jugador puede mover el cubo hacia la izquierda y hacia la derecha mediante las teclas "a" y "d". Además, se verifica si el cubo cae por debajo de una cierta posición en el eje Y, en cuyo caso se llama al método EndGame() del objeto GameManager para reiniciar el juego.

2. FollowPlayer.cs: Este script se encarga de hacer que la cámara siga al jugador. La posición de la cámara se actualiza en cada fotograma para estar siempre en la posición del jugador más un desplazamiento determinado.

3. PlayerCollision.cs: Este script maneja las colisiones del jugador con los obstáculos. Si el jugador colisiona con un obstáculo, se desactiva el componente PlayerMovement y se llama al método EndGame() del objeto GameManager para reiniciar el juego.

4. EndTrigger.cs: Este script detecta cuando el jugador alcanza el final del nivel mediante un "trigger". Al entrar en el trigger, se llama al método CompleteLevel() del objeto GameManager para mostrar el mensaje de "Nivel completado".

5. GameManager.cs: Este script se encarga de administrar el flujo del juego. Tiene dos métodos principales: CompleteLevel() y EndGame(). El método CompleteLevel() muestra el mensaje de "Nivel completado " y reinicia el juego después de un breve retraso. El método EndGame() se llama cuando el jugador colisiona con un obstáculo y reinicia el juego después de un breve retraso.

## V. Conclusiones

El juego desarrollado consiste en controlar un cubo para evitar obstáculos en movimiento y llegar al final del nivel. Se ha implementado un sistema de puntuación basado en la posición del cubo en el eje Z, así como mensajes de " Nivel completado " y "Game Over" al completar el nivel o chocar con un obstáculo, respectivamente. El desarrollo de este juego ha permitido aplicar los conocimientos adquiridos en la materia y mejorar las habilidades de desarrollo de videojuegos en Unity.

Github código, video y documentación

https://github.com/freddmejia/video-juegos.git