VIDEOJUEGO 3D EN UNITY: CUBEDGE

Ángel Guerrero López

Grado en Ingeniería Informática, Universidad de Salamanca

Plaza de los Caídos, s/n, 37008, Salamanca, España

angelgl@usal.es

Índice

1. Introducción

CUBEDGE es el típico juego 3D en el que vas avanzando de forma lineal y automática por un escenario tratando de esquivar obstáculos. Cuando un nivel es completado, se avanza al siguiente incrementándose la dificultad de este.

2. Fases de desarrollo

Se irán describiendo a grandes rasgos los elementos incluidos en cada una de las fases.

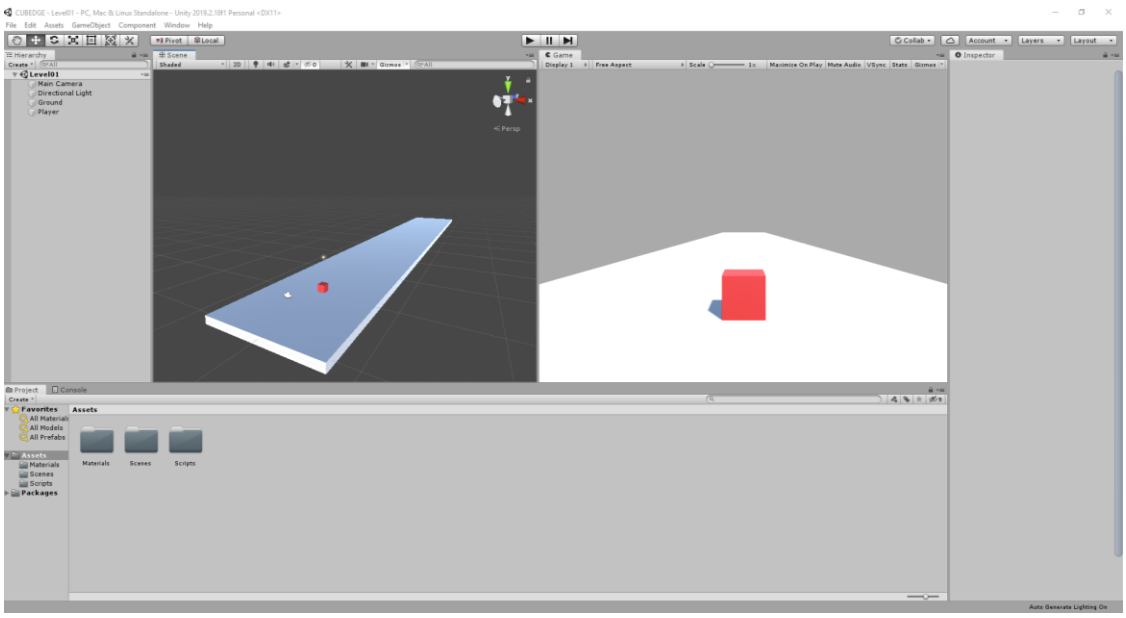
2.1 Creación de escenario y personaje

El escenario es un objeto cubo cuya coordenada Z se ha incrementado para estirarlo.

El jugador también es un cubo al cual se ha modificado su material para resaltarlo sobre el escenario por el que se desplazará. Se añade su componente de físicas, así como un script que le suministra una fuerza constante en el eje Z para el avance y una fuerza constante en el eje X para el desplazamiento horizontal según se presione la tecla “A” o “D”.

Para evitar que el cubo rote sobre sí mismo por el escenario conforme avanza, en vez de congelar su rotación en el eje X (puesto que se necesitará para futuras colisiones), se crea un nuevo Physic Material con ambas fricciones a 0 y se añade al escenario.

Finalmente, se hace que la cámara siga al personaje con un nuevo script. No se establece al jugador como padre de la cámara porque en las rotaciones no tendría un efecto deseado.

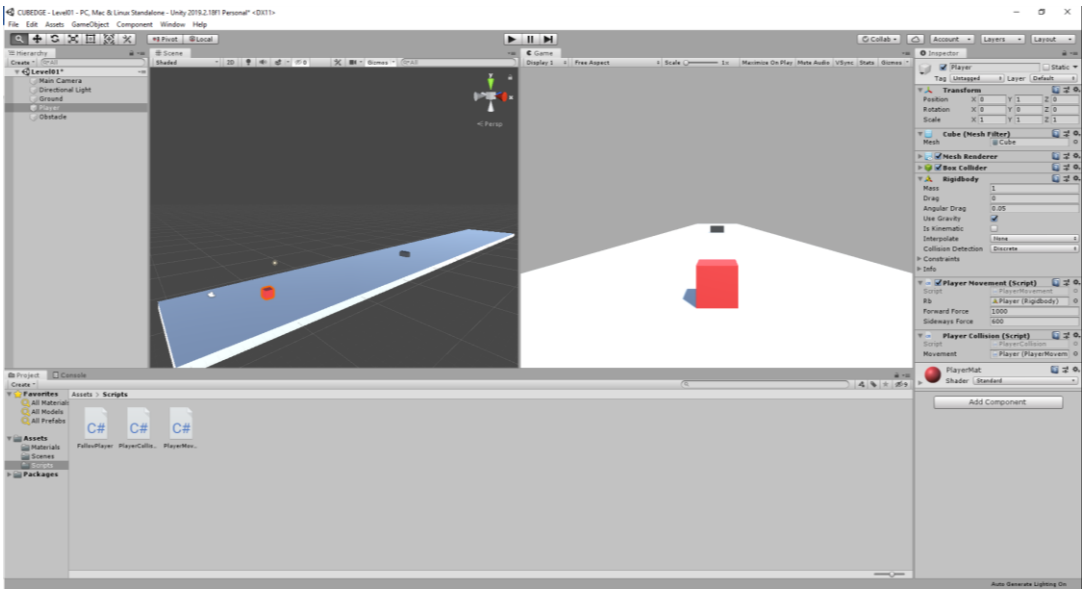


2.2 Colisiones

Se crea un nuevo script para implementar las colisiones y se asocia al jugador.

Se crea un nuevo GameObject cubo que serán los obstáculos con los que se colisionará, también se genera un nuevo material para ellos, así como su componente de físicas, y se posiciona en el escenario.

En el nuevo script, se crea una referencia al script del movimiento del jugador, para deshabilitarlo cuando se produzca una colisión.

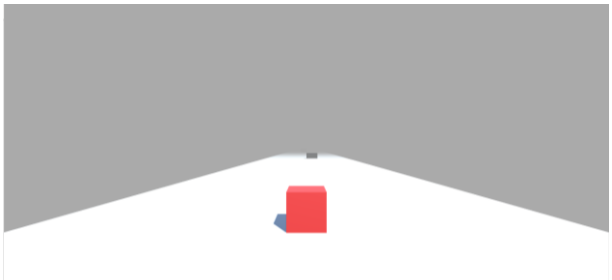


2.3 Diseño de nivel

Se crea un prefab con el objeto obstáculo recientemente creado para poder crear múltiples de ellos, redimensionarlos y posicionarlos en el escenario para el diseño del nivel.

A la hora de diseñar un nivel y hacer pruebas, se puede comprobar su balance para ajustar diferentes parámetros. En este caso, se decide incrementar el índice de rozamiento del jugador con el escenario, para ello se modifica la propiedad Drag de su componente de físicas.

En los ajustes de iluminación se marca y ajusta la niebla para conseguir un efecto que hace que los obstáculos se vayan viendo progresivamente.

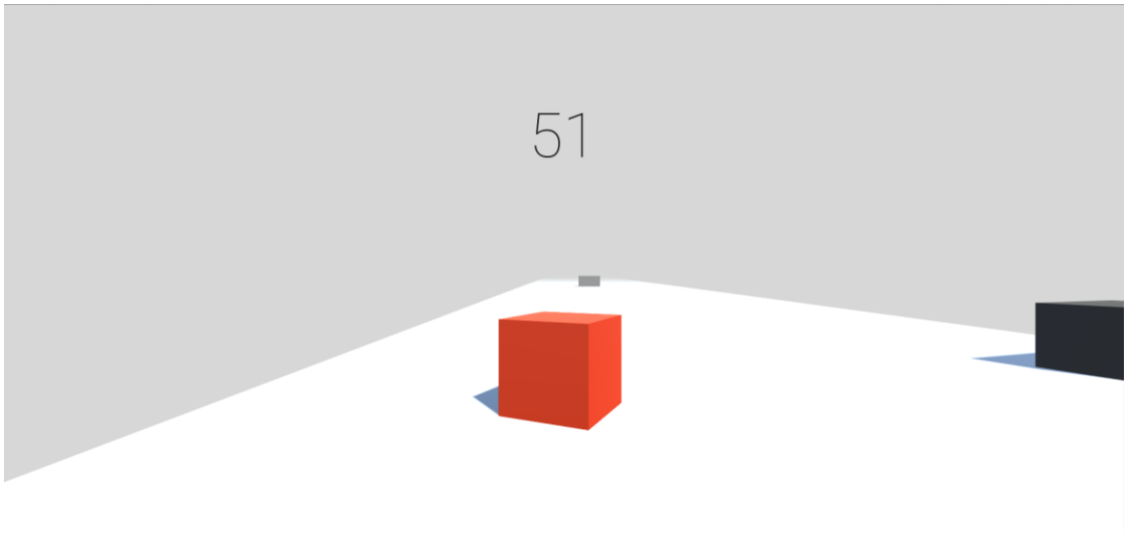


2.4 UI element: puntuación

Se ha creado un elemento de interfaz de tipo texto que mostrará la puntuación actual, además se ha descargado una fuente de Google llamada Roboto para utilizarla aquí y en posteriores elementos de interfaz.

Para actualizar la puntuación (basada en la distancia recorrida por el jugador en el eje Z) se crea un nuevo script con una referencia al jugador y al propio texto.

En este punto también se decidió cambiar un poco el color del material usado en el jugador.



2.5 Gestión de fin de partida

Se ha creado un objeto GameManager que gestionará el fin de partida, tanto al colisionar como al salir del escenario y reiniciará el nivel actual. Este es su script asociado en este punto

```
using UnityEngine;
using UnityEngine.SceneManagement;

2 referencias
public class Manager : MonoBehaviour
{
    bool gameHasEnded = false;
    public float restartDelay = 1f;

    2 referencias
    public void EndGame()
    {
        if (gameHasEnded == false)
        {
            gameHasEnded = true;
            Debug.Log("GAME OVER");
            Invoke("Restart", restartDelay);
        }

    }

    // Obtiene la escena actual y la reinicia
    0 referencias
    void Restart()
    {
        SceneManager.LoadScene(SceneManager.GetActiveScene().name);
    }
}

```

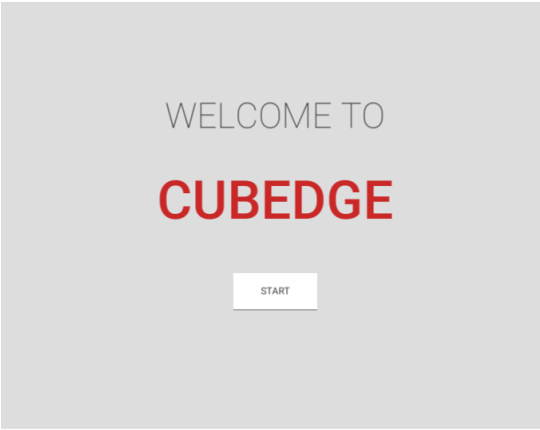
2.6 Completar niveles

Se crea un nuevo objeto cubo, se redimensiona ocupando el ancho del escenario y se desplaza hasta el final del nivel, de esta forma podrá disparar un evento que indique que se ha completado el nivel.

Se crea un nuevo UI element de tipo panel para mostrar al jugador un anuncio con una pequeña animación (consistente en cambiar la opacidad de los elementos progresivamente) informando de que ha terminado el nivel. Dicho panel es rellenado con dos elementos de tipo texto. Cuando esta animación termina se pasa a la siguiente escena mediante un Animation Event establecido.

2.7 Panel de bienvenida y créditos

Se crea una nueva escena con un panel, varios textos y un botón. Dicha escena se duplica, pues la pantalla de inicio y de créditos son muy similares: una dará paso a comenzar el juego y la otra lo finalizará.



El orden de escenas se configura en Build Settings, listando las escenas en el orden apropiado. El paso de una a otra se hace tomando el índice que en esta lista representa a la escena actual e incrementándolo en 1 (por ello es necesaria una escena final de créditos)

3. Referencias

El juego se realizó siguiendo una guía de este canal:

Brackeys - https://www.youtube.com/channel/UCYbK_tjZ2OrIZFBvU6CCMiA

El diseño de los tres niveles que componen el juego es propio.

Música del menú: https://opengameart.org/content/menu-music