



IES AUGUSTO GONZÁLEZ DE LINARES
DEPARTAMENTO DE INFORMÁTICA


Anexo I

Requisitos PMDM

PROYECTO DICIEMBRE 2023 2º CURSO

**GRADO SUPERIOR DE DESARROLLO DE
APLICACIONES MULTIPLATAFORMA**

2022/2023

	Ciclo: Desarrollo de Aplicaciones Multiplataforma	Grupo: DAMT2
	Título: Requisitos PMDM	

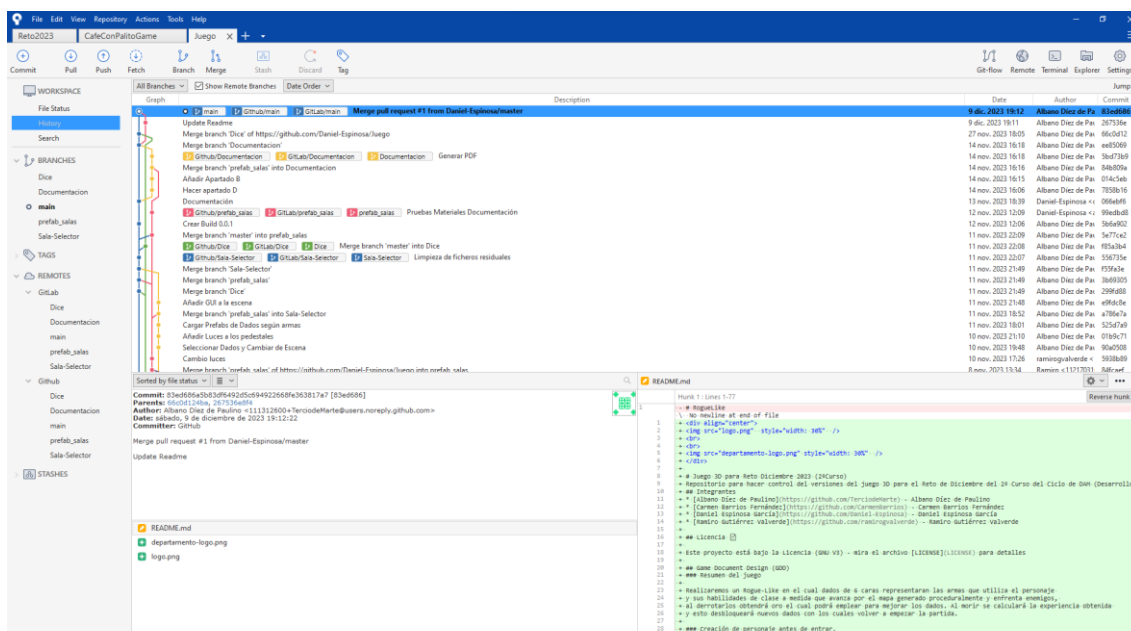
Índice

1.	Documentación PMDM.....	2
1.1.	Generación de GDD del videojuego incluido en README del GitLab, donde se explicitarán las características principales del juego.	2
1.2.	Animaciones 2D o 3D, ya sea de personaje o de algún elemento del escenario.....	3
1.3.	Incluirá los elementos ya vistos en el módulo hasta el momento: lógica de juego, inclusión de asset (sprites o modelos 3D), sonido, efectos... ..	5
1.4.	Varias escenas correspondientes con: escena de inicio, varios niveles de juego (al menos uno por integrante del equipo), escena de fin.	6
1.5.	Pruebas y optimización: uso del profiler de Unity y realización de pruebas de usuario.....	7
1.6.	Generación ejecutable: generar ejecutable para Windows (.exe). ..	10
1.7.	Generación APK: el proyecto debe incluir la posibilidad de generar el juego para dispositivos móviles Android, el control deberá ajustarse para utilizar entradas táctiles.....	12
1.8.	Utilización de Assets: pueden utilizarse asset gratuitas del store de Unity o externas siempre de acuerdo con la licencia de uso que tengan. .	13
1.9.	Creación de un instalador para el juego	Error! Marcador no definido.

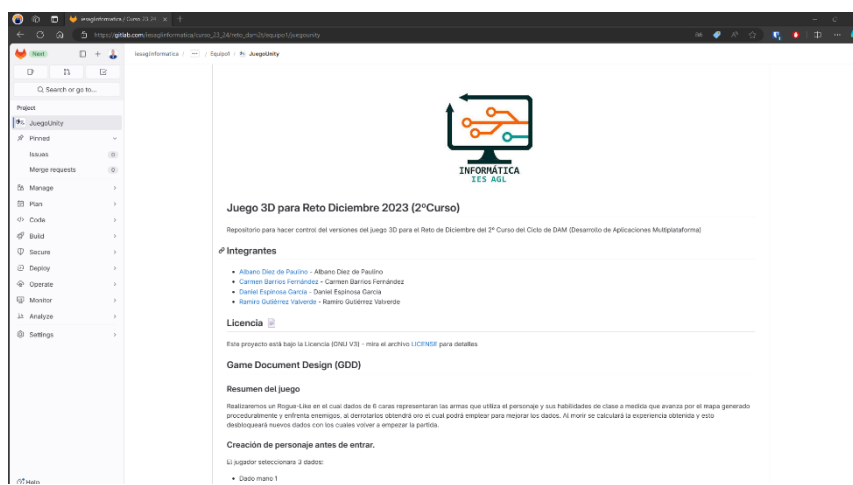
1. Documentación PMDM

1.1. Generación de GDD del videojuego incluido en README del GitLab, donde se explicitarán las características principales del juego.


Partimos del GDD que diseñamos para la asignatura ya que en este reto vamos a continuar con el desarrollo de este, lo primero fue subir nuestro repositorio de GitHub en GitLab, esto se puede hacer si alguien tiene clonado el repositorio en su ordenador y establece múltiples remotos, y actualizar el archivo “readme.md”.



Software 1 – SourceTree con múltiples remotos



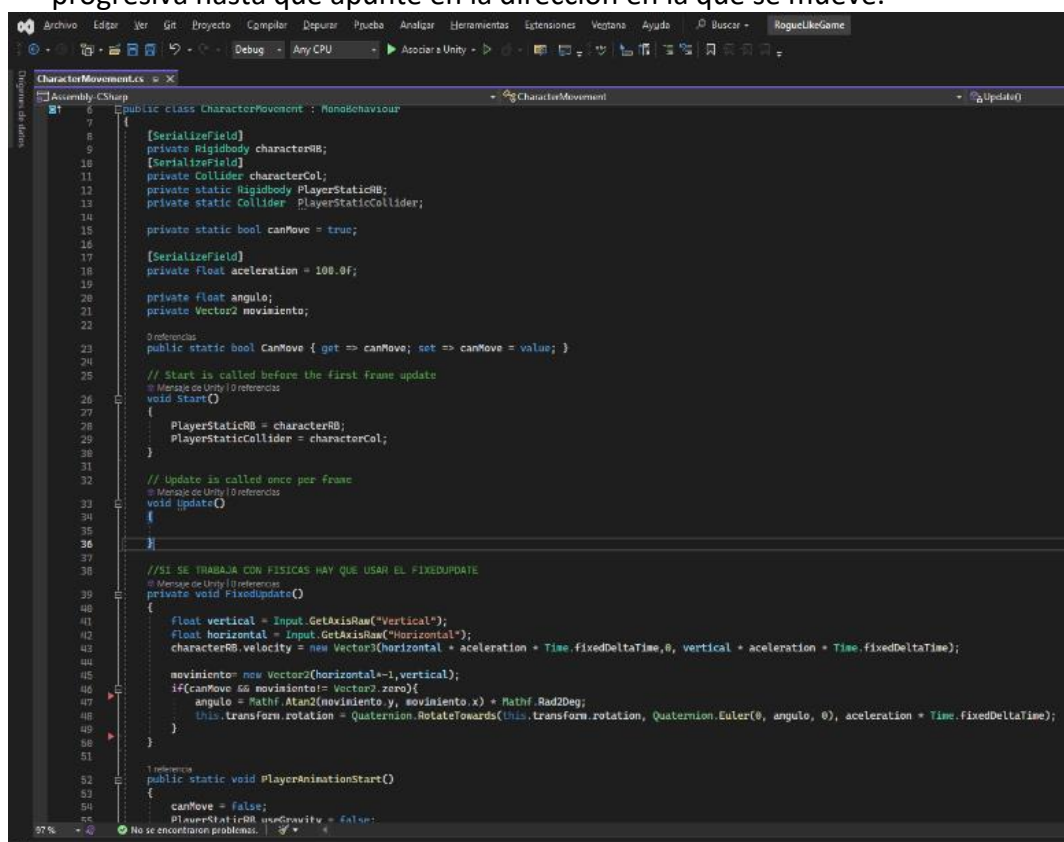
Web 1 – Repositorio Remoto GitLab

	Ciclo: Desarrollo de Aplicaciones Multiplataforma	Grupo: DAMT2
	Título: Requisitos PMDM	

1.2. Animaciones 2D o 3D, ya sea de personaje o de algún elemento del escenario.

El Script CharacterMovement se ocupa de controlar el movimiento del personaje utilizando los Input Axis de Unity.

Al mover al personaje sobre el espacio esta ira rotando de forma progresiva hasta que apunte en la dirección en la que se mueve.




```

6 public class CharacterMovement : MonoBehaviour
7 {
8     [SerializeField]
9     private Rigidbody characterRB;
10    [SerializeField]
11    private Collider characterCol;
12    private static Rigidbody PlayerStaticRB;
13    private static Collider PlayerStaticCollider;
14
15    private static bool canMove = true;
16
17    [SerializeField]
18    private float acceleration = 100.0f;
19
20    private float angulo;
21    private Vector2 movimiento;
22
23    // 0 referencias
24    public static bool CanMove { get => canMove; set => canMove = value; }
25
26    // Start is called before the first frame update
27    // Mensaje de Unity 0 referencias
28    void Start()
29    {
30        PlayerStaticRB = characterRB;
31        PlayerStaticCollider = characterCol;
32    }
33
34    // Update is called once per frame
35    // Mensaje de Unity 0 referencias
36    void Update()
37    {
38    }
39
40    // SI SE TRABAJA CON FISICAS HAY QUE USAR EL FIXEDUPDATE
41    // Mensaje de Unity 0 referencias
42    private void FixedUpdate()
43    {
44        float vertical = Input.GetAxisRaw("Vertical");
45        float horizontal = Input.GetAxisRaw("Horizontal");
46        characterRB.velocity = new Vector3(horizontal * acceleration * Time.fixedDeltaTime, 0, vertical * acceleration * Time.fixedDeltaTime);
47        movimiento = new Vector2(horizontal * 1, vertical);
48        if(canMove && movimiento != Vector2.zero){
49            angulo = Mathf.Atan2(movimiento.y, movimiento.x) * Mathf.Rad2Deg;
50            this.transform.rotation = Quaternion.RotateTowards(this.transform.rotation, Quaternion.Euler(0, angulo, 0), acceleration * Time.FixedDeltaTime);
51        }
52    }
53
54    // 1 referencia
55    public static void PlayerAnimationStart()
56    {
57        canMove = false;
58        PlayerStaticRB.useGravity = false;
59    }
60 }

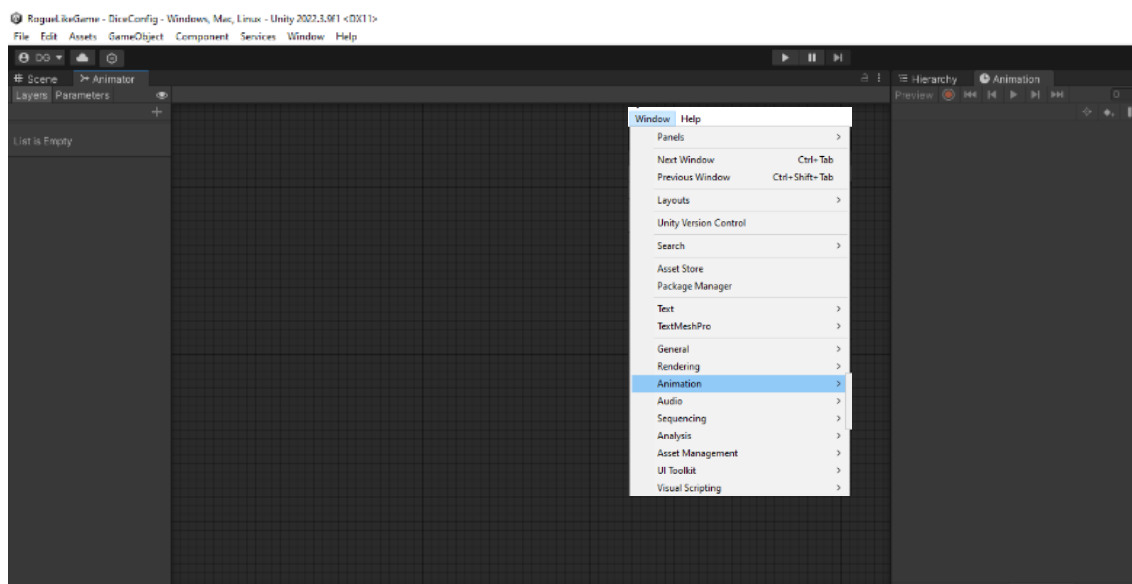
```

Código 1 - Script CharacterMovement


	Ciclo: Desarrollo de Aplicaciones Multiplataforma	Grupo: DAMT2
	Título: Requisitos PMDM	

Unity cuenta con su gestor de animaciones el cual permite montar secuencias de animación de Objetos.

En lo que llevamos realizado de proyecto no hemos podido implementar aun las animaciones para el personaje principal, pero llegaremos a ello en el futuro.

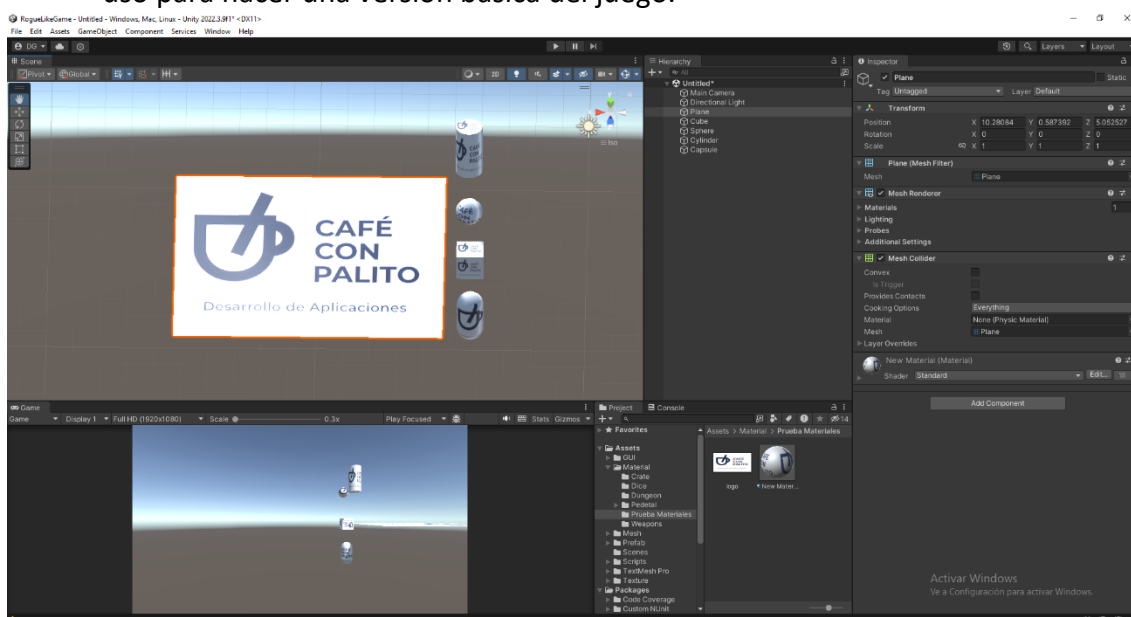


Software 2 – Unity Animator View

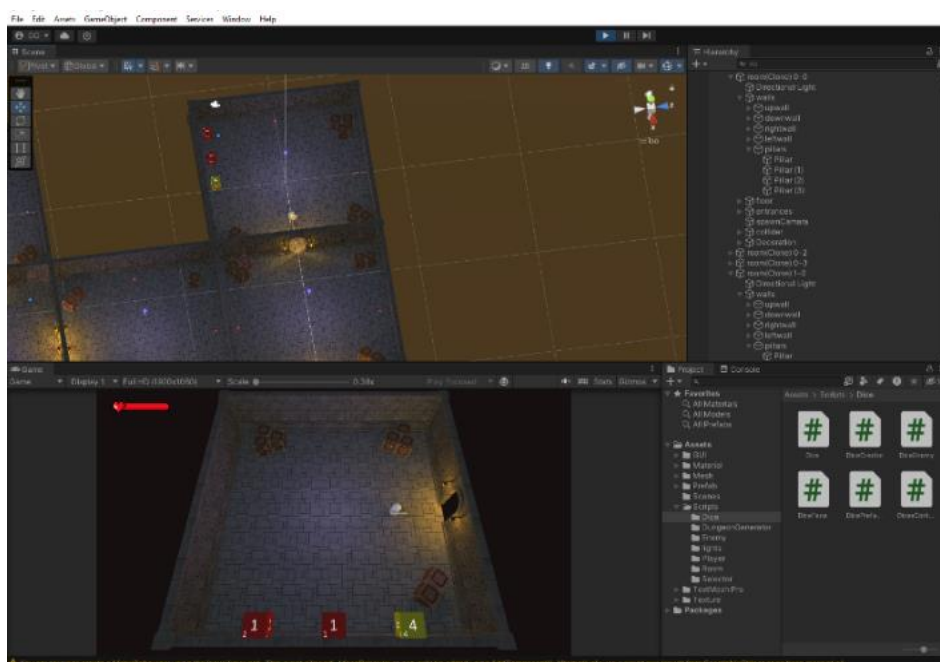
	Ciclo: Desarrollo de Aplicaciones Multiplataforma	Grupo: DAMT2
	Título: Requisitos PMDM	

1.3. Incluirá los elementos ya vistos en el módulo hasta el momento: lógica de juego, inclusión de assets (sprites o modelos 3D), sonido, efectos...


Para desarrollar este videojuego empezamos con assets básicos, investigamos los múltiples assets que soporta Unity y buscamos assets de libre uso para hacer una versión básica del juego.



Software 3 – Assets Unity

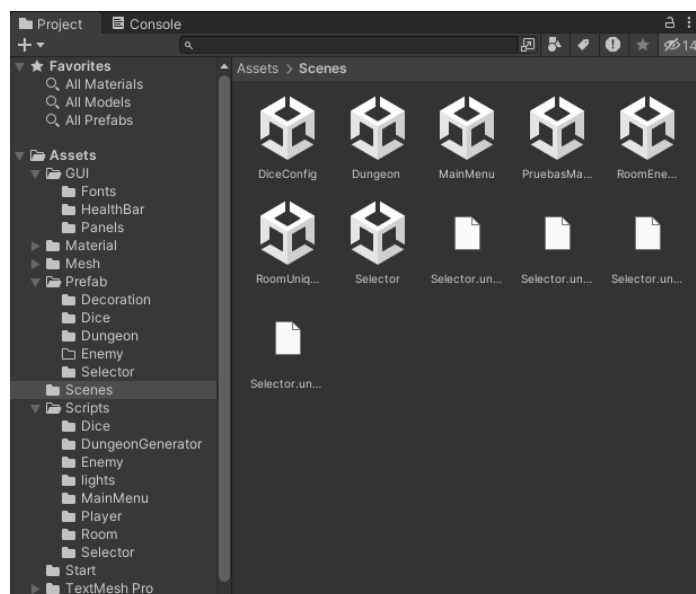


Software 4 – Escena con Assets Free Use

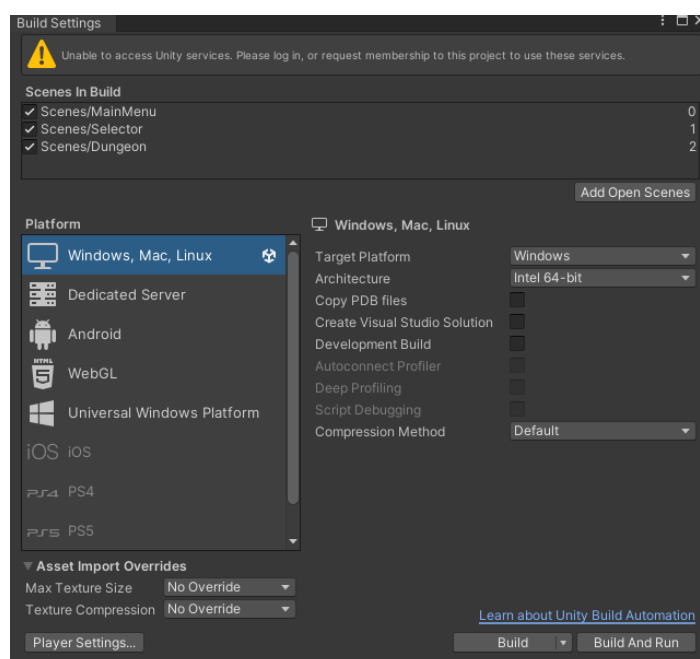
	Ciclo: Desarrollo de Aplicaciones Multiplataforma	Grupo: DAMT2
	Título: Requisitos PMDM	

1.4. Varias escenas correspondientes con: escena de inicio, varios niveles de juego (al menos uno por integrante del equipo), escena de fin.


Para el desarrollo del juego hemos usado múltiples escenas para programar las diferentes funciones, aunque en el juego final solo hay tres, el menú principal, el selector de datos para jugar, y la mazmorra aleatoria.



Software 5 – Explorador de Archivos en Unity

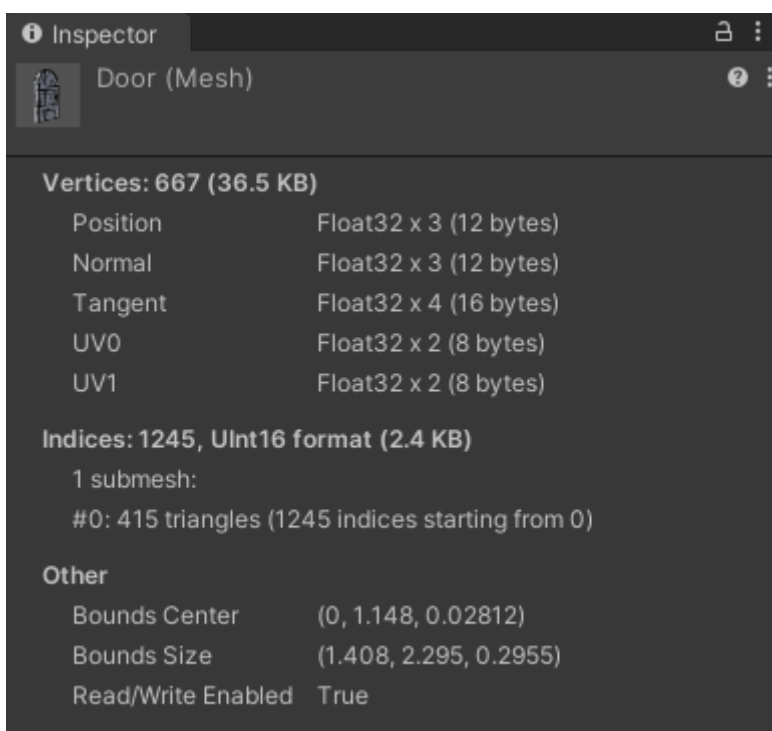


Software 6 – Build Settings del proyecto

	Ciclo: Desarrollo de Aplicaciones Multiplataforma	Grupo: DAMT2
	Título: Requisitos PMDM	

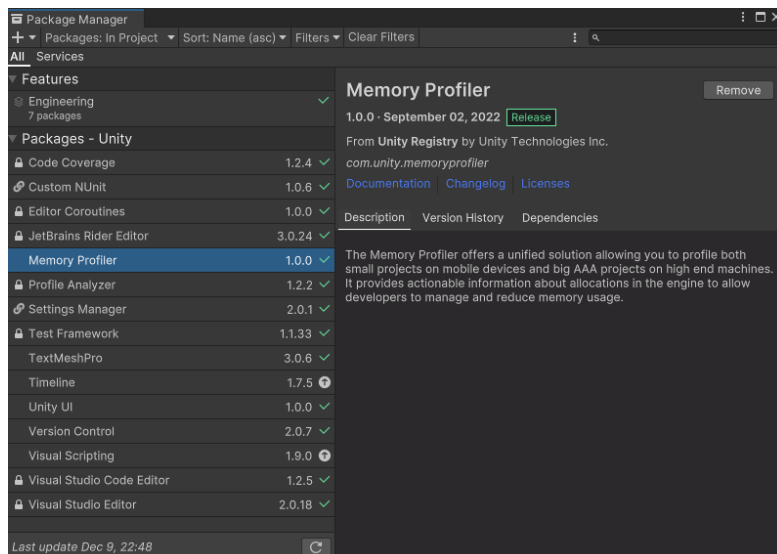
1.5. Pruebas y optimización: uso del profiler de Unity y realización de pruebas de usuario.

A lo largo de desarrollo a sido necesario ir optimizando el juego para que pueda ser ejecutado en cualquier ordenador, es comprobar el número de vértices de los assets 3D que nos descargamos, a menor número de vértices mejor rendimiento ya que el ordenador no tiene que procesar tantas coordenadas para mostrar un objeto.

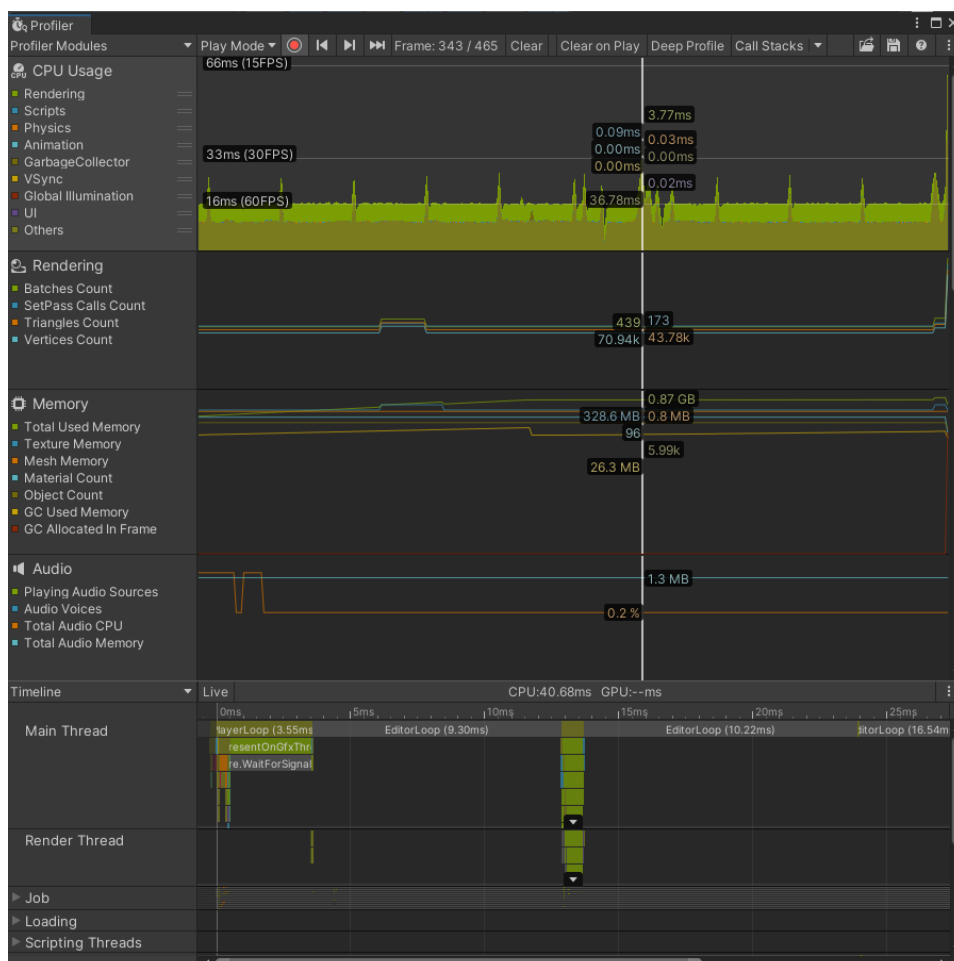


Software 7 – Inspector Unity (Mesh)


Pero muchas veces es necesario observar en que esta empleando el ordenador los recursos para optimizar las partes que se demoren mas de lo necesario, por ejemplo, que un script se quede bloqueado un par de fotogramas hace que el rendimiento empeore, para verlo de forma sencilla hay una paquete hecho por Unity llamado “Profiler”



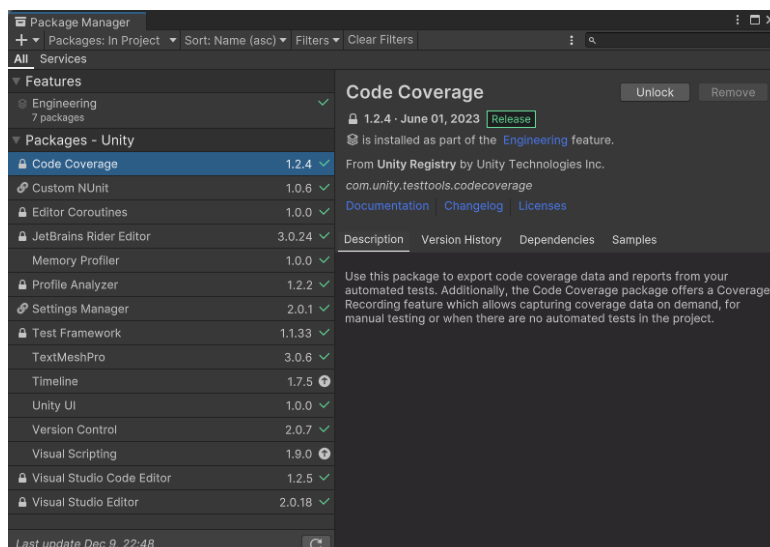
Software 8 – Package Manager(Profiler)



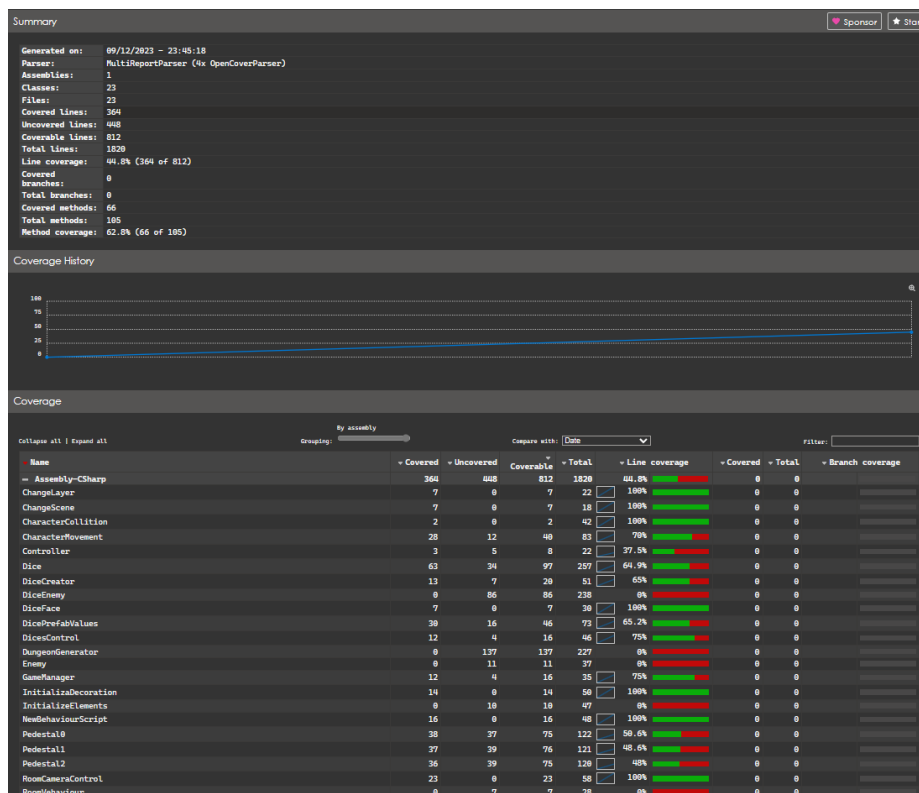
Software 9 – Modulo Profiler

	Ciclo: Desarrollo de Aplicaciones Multiplataforma	Grupo: DAMT2
	Título: Requisitos PMDM	

Otra forma de optimización es mirando que código está “muerto”, es decir que nunca es ejecutado, eliminar este código residual hace que el compilador tarde unos milisegundos menos en compilar, parece que no se obtiene una gran mejora ya que nuestro juego no tiene muchas líneas de código, pero en juegos mucho más grande esto puede suponer que el juego sea injugable. Para ello unity dispone del paquete llamado “Code Coverage”.



Software 10 – Package Manager (Code Coverage)



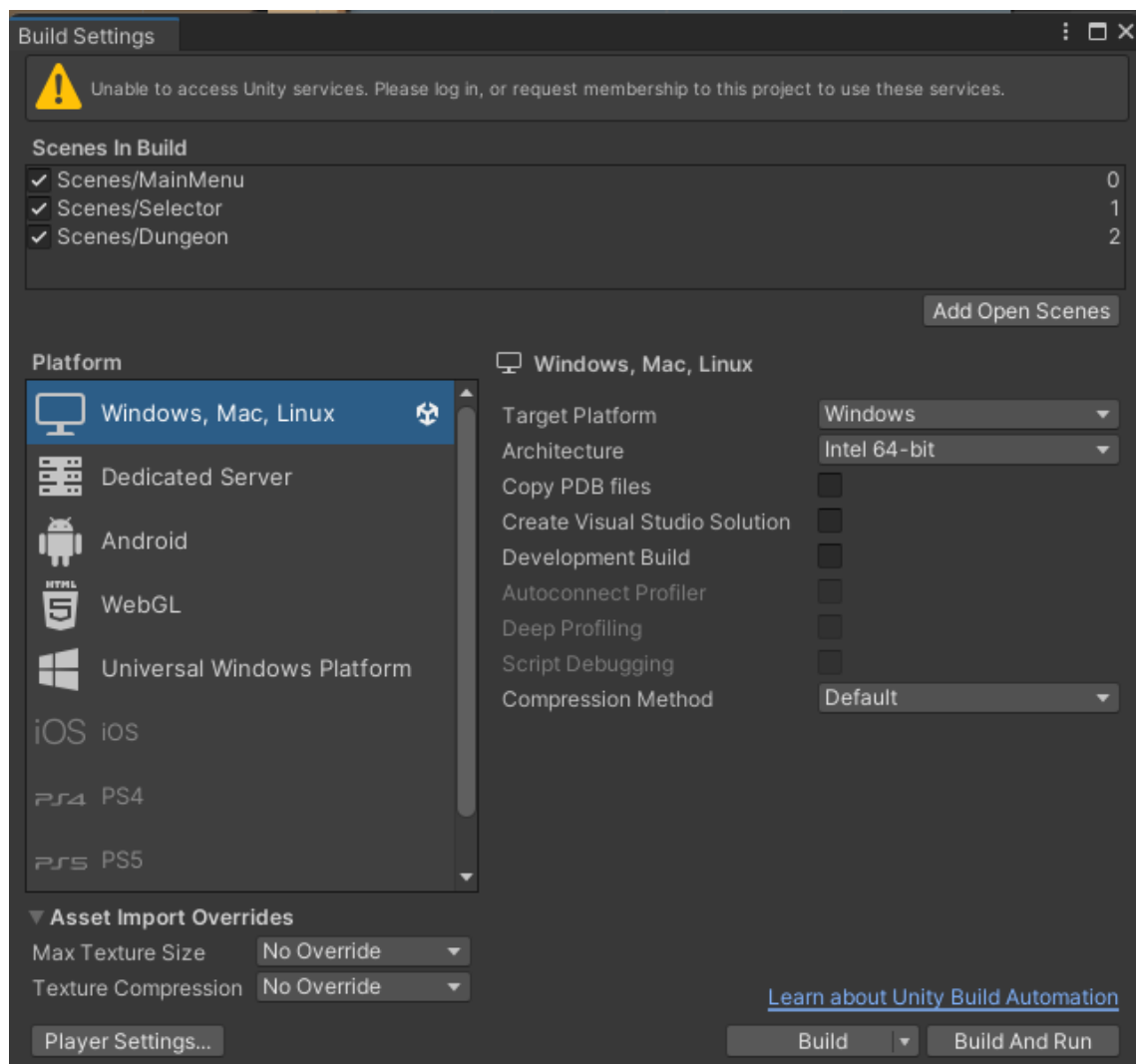
Web 2 – Test Code Coverage

	Ciclo: Desarrollo de Aplicaciones Multiplataforma	Grupo: DAMT2
	Título: Requisitos PMDM	

Además, otra buena forma de probar el juego es realizar pruebas con usuarios que no sean del desarrollo así puedes encontrar errores que los desarrolladores no encuentran por la visión sesgada.

1.6. Generación ejecutable: generar ejecutable para Windows (.exe).

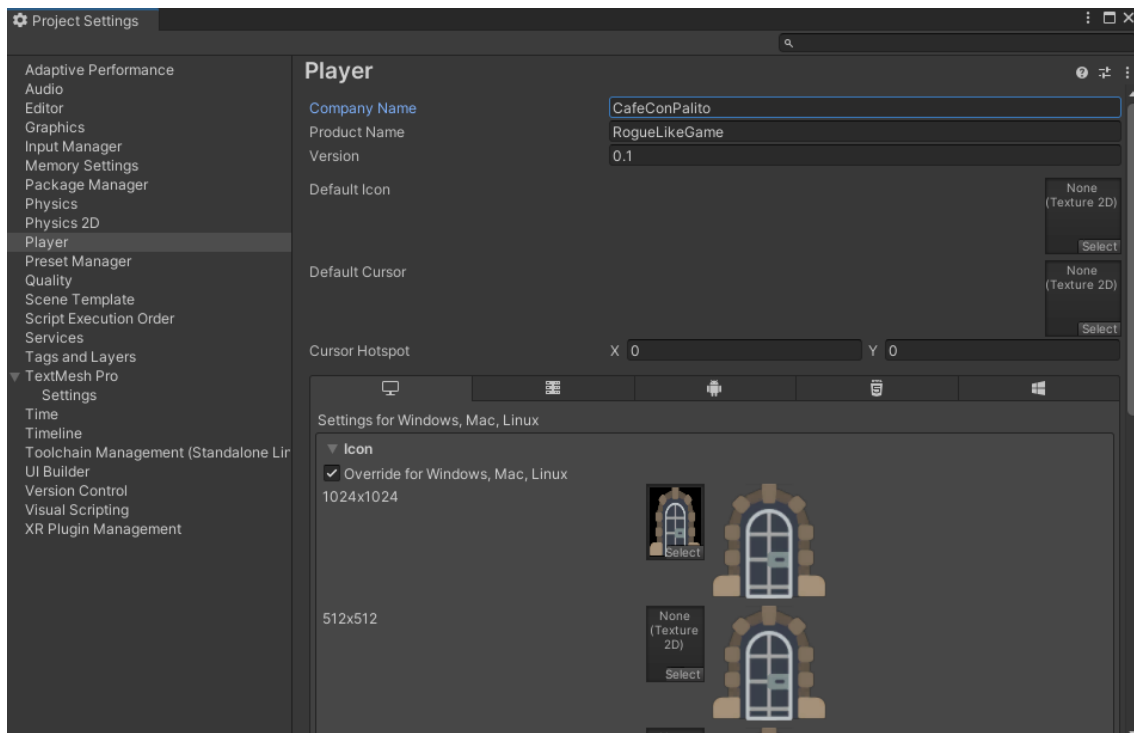
Tras finalizar el desarrollo se debe generar un ejecutable para probar que todo lo desarrollado se ejecuta correctamente, en nuestro caso lo hacemos sobre Windows y MacOs.




Software 11 – Build Settings (Windows)

	Ciclo: Desarrollo de Aplicaciones Multiplataforma	Grupo: DAMT2
	Título: Requisitos PMDM	

Unity permite modificar gran parte de los parámetros de ejecución. Como ejemplo modificar el splash inicial, que ejecute en modo ventana o el icono de la aplicación.

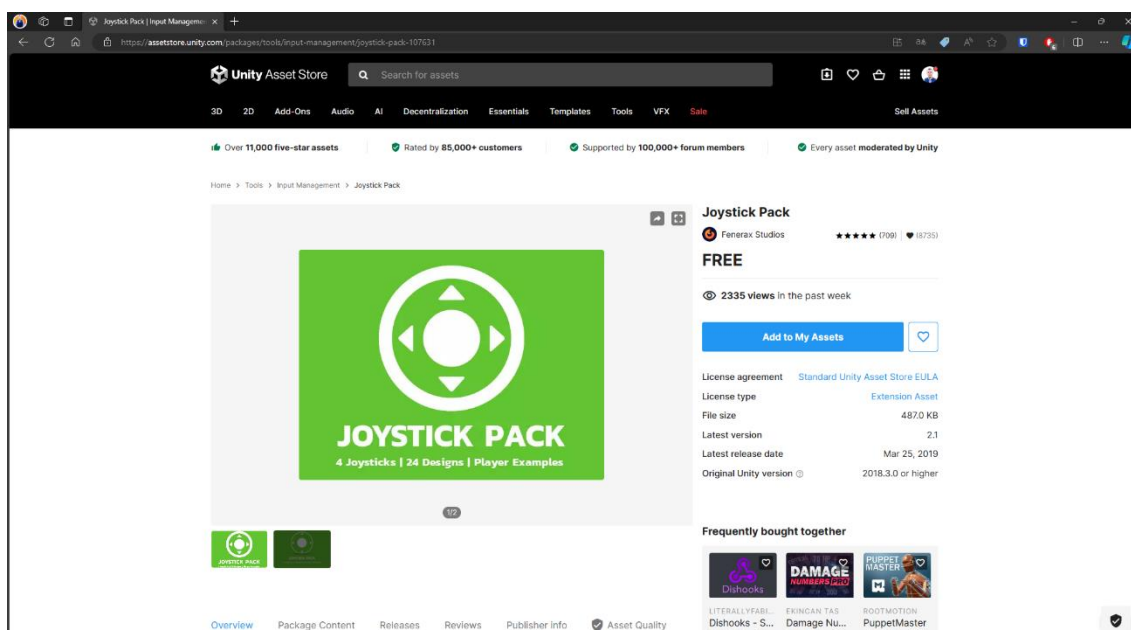


Software 12 – Project Settings

	Ciclo: Desarrollo de Aplicaciones Multiplataforma	Grupo: DAMT2
	Título: Requisitos PMDM	

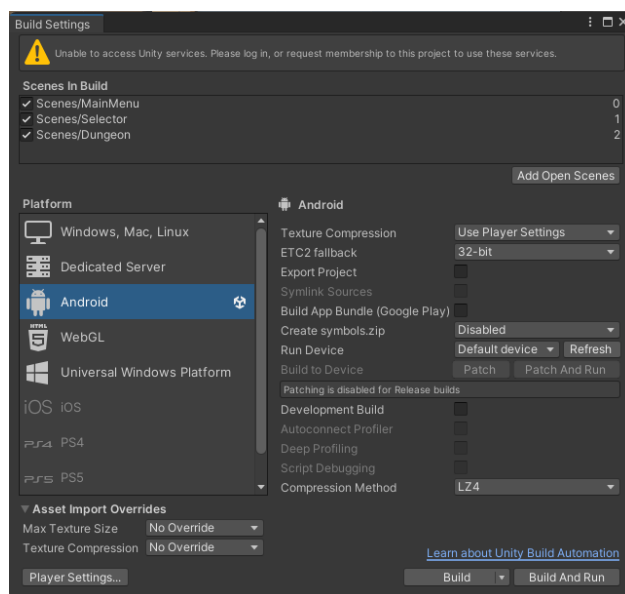
1.7. Generación APK: el proyecto debe incluir la posibilidad de generar el juego para dispositivos móviles Android, el control deberá ajustarse para utilizar entradas táctiles.

Si se quiere pasar el desarrollo a Dispositivos Móviles, es tan sencillo como adaptar los controles a Android, para ello nosotros pasamos el movimiento del personaje de las teclas WASD a un joystick virtual descargado de la Assets Store de Unity.




Web 3 - Unity Asset Store (Joystick Pack)

Tras ello solo queda generar la APK y probar el juego.



Software 13 - Build Settings (Android)

	Ciclo: Desarrollo de Aplicaciones Multiplataforma	Grupo: DAMT2
	Título: Requisitos PMDM	

1.8. Utilización de Assets: pueden utilizarse asset gratuitas del store de Unity o externas siempre de acuerdo con la licencia de uso que tengan.

Como ya dijimos en puntos anteriores, hemos usado assets de la Unity Assets Store para nuestro proyecto.

Si se desea adquirir algún asset de la tienda, hay que tener en cuenta que todas las compras realizadas en el store están ligadas a una cuenta de usuario, por lo que el usuario que hace la compra debe importar el asset al proyecto para que todos los desarrolladores puedan usarlo.


	Ciclo: Desarrollo de Aplicaciones Multiplataforma	Grupo: DAMT2
	Título: Requisitos PMDM	

Tabla de Ilustraciones

Código 1 - Script CharacterMovement.....	3
Web 1 – Repositorio Remoto GitLab.....	2
Web 2 – Test Code Coverage	9
Web 3 - Unity Asset Store (Joystick Pack)	12
Software 1 – SourceTree con múltiples remotos.....	2
Software 2 – Unity Animator View.....	4
Software 3 – Assets Unity	5
Software 4 – Escena con Assets Free Use	5
Software 5 – Explorador de Archivos en Unity	6
Software 6 – Build Settings del proyecto	6
Software 7 – Inspector Unity (Mesh)	7
Software 8 – Package Manager(Profiler)	8
Software 9 – Modulo Profiler	8
Software 10 – Package Manager (Code Coverage)	9
Software 11 – Build Settings (Windows)	10
Software 12 – Project Settings	11
Software 13 - Build Settings (Android).....	12