**2023 - 2024**

**MISIÓN TFG**

**IES MELCHOR GASPAR DE JOVELLANOS**

**GRADO SUPERIOR:**

**DESARROLLO DE APLICACIONES MULTIPLATAFORMA**

**TUTOR TFG: JAVIER ORTIZ LAGUNA**

**AUTORES:**

* **ADRIÁN MORENO NOVILLO**
* **MARCOS MARTÍN GARCÍA**
* **SERGIO GONZÁLEZ MONTERO**

ÍNDICE

Contenido

[**1.** **Resumen** 1](#_Toc167615932)

[**2.** **Abstract** 2](#_Toc167615933)

[**3.** **Planteamiento del problema y justificación** 2](#_Toc167615934)

[**4.** **Desarrollo** 2](#_Toc167615935)

[**4.1.** **Datos técnicos** 2](#_Toc167615936)

[**4.1.1.** **Nombre de la aplicación** 2](#_Toc167615937)

[**4.1.2.** **Plataforma** 2](#_Toc167615938)

[**4.1.3.** **Lenguaje de programación** 2](#_Toc167615939)

[**4.1.4.** **Requisitos** 2](#_Toc167615940)

[**4.2.** **Funcionalidad** 2](#_Toc167615941)

[**4.3.** **Características.** 2](#_Toc167615942)

[**4.1.1.** **Diagramas E­R** 2](#_Toc167615943)

[**4.1.2.** **Diagramas de clases** 3](#_Toc167615944)

[**4.4.** **Usabilidad** 3](#_Toc167615945)

[**4.1.1.** **Diseño de interfaces.** 3](#_Toc167615946)

[**4.1.2.** **Casos de uso.** 3](#_Toc167615947)

[**4.5.** **Portabilidad.** 3](#_Toc167615948)

[**4.1.1.** **Plataformas.** 3](#_Toc167615949)

[**4.6.** **Rendimiento.** 3](#_Toc167615950)

[**4.1.1.** **Pruebas realizadas** 3](#_Toc167615951)

[**5.** **Conclusiones:** 3](#_Toc167615952)

[**6.** **Referencias bibliográficas:** 3](#_Toc167615953)

1. **Resumen**

Misión TFG es un proyecto de desarrollo de un videojuego en 2D, creado con la herramienta Unity. En este emocionante juego, tomamos el control de nuestro amigo Emilio, el protagonista, quien deberá, por todos los medios posibles, lograr llegar a clase para exponer su trabajo de fin de grado (TFG). Emilio se enfrentará a una serie de adversidades y deberá superar cada obstáculo para llegar a tiempo y así poder aprobar el curso.

El juego recrea, con escenas en pixel art, el instituto IES Melchor Gaspar de Jovellanos y sus alrededores. Además, se han añadido personajes emblemáticos al juego, como los profesores, para darle una inmersión y una personalidad aún mayores. Estos personajes no solo añaden profundidad al juego, sino que también representan figuras importantes en la vida académica de Emilio, proporcionando una experiencia más auténtica y envolvente.

¿Serás capaz de ayudar a Emilio a llegar a tiempo para exponer su trabajo, o te quedarás a las puertas de aprobar el Grado Superior de Desarrollo de Aplicaciones Multiplataformas? Todo depende de ti. ¡La aventura y el éxito de Emilio están en tus manos!

1. **Abstract**

**Mission TFG** is a project for developing a 2D video game created with the Unity tool. In this exciting game, we take control of our friend Emilio, the protagonist, who must, by all possible means, manage to get to class to present his final degree project (TFG). Emilio will face a series of adversities and must overcome each obstacle to arrive on time and thus be able to pass the course.

The game recreates, with pixel art scenes, the IES Melchor Gaspar de Jovellanos school and its surroundings. Additionally, iconic characters, such as the teachers, have been added to the game to give it even more immersion and personality. These characters not only add depth to the game but also represent important figures in Emilio's academic life, providing a more authentic and engaging experience.

Will you be able to help Emilio get to class on time to present his project, or will you fall short of passing the Advanced Degree in Multiplatform Application Development? It all depends on you. The adventure and Emilio's success are in your hands!

1. **Planteamiento del problema y justificación**

Texto

1. **Desarrollo**

Texto

* 1. **Datos técnicos**

Los datos técnicos de una aplicación son fundamentales para entender su funcionamiento, compatibilidad y requisitos. Estos datos ofrecen una visión detallada de los componentes tecnológicos que forman la aplicación y las condiciones necesarias para su correcto funcionamiento.

* + 1. **Nombre de la aplicación**

El nombre de nuestro video juego es: **Misión TFG**. Hemos elegido este título porque describe a la perfección el desempeño de la aplicación.

* + 1. **Plataforma**

El videojuego ha sido desarrollado principalmente para PC, específicamente en el sistema operativo Windows. Esta elección se debe a la amplia base de usuarios de Windows y a su robusta compatibilidad con una variedad de hardware de juego. Sin embargo, gracias a la flexibilidad de Unity, el juego podría implementarse sin mucho esfuerzo para otras plataformas como consolas (PlayStation, Xbox), Mac, e incluso dispositivos móviles (iOS, Android). Unity facilita la exportación y adaptación del juego a estas plataformas mediante su soporte multiplataforma.

* + 1. **Lenguaje de programación**

Para el desarrollo de Misión TFG, se ha utilizado Unity como herramienta de trabajo, y el lenguaje de programación principal empleado ha sido C#. Unity es uno de los motores de juegos más populares y versátiles del mercado, adoptando C# debido a sus múltiples ventajas que se alinean perfectamente con las necesidades del desarrollo de videojuegos.

C# se integra de manera nativa con Unity, facilitando la escritura de scripts para controlar el comportamiento de los objetos del juego, gestionar la lógica del juego y manipular gráficos y sonidos. Unity proporciona un entorno de desarrollo integrado (IDE) donde hemos podido escribir, probar y depurar los scripts en C#.

Los Scripts en C# se utilizan para programar el movimiento y las acciones de los personajes dentro del juego.

Hemos podido definir cómo los objetos del juego interactúan entre sí y con el jugador. Esto incluye la detección de colisiones, la manipulación de objetos, la implementación de mecánicas de juego complejas…

Además, permite crear y gestionar la interfaz de usuario del juego, como menús, puntuaciones y vidas, etc.

En resumen, el uso de C# en Unity no solo es una elección lógica debido a su integración y facilidades proporcionadas por el motor, sino que también potencia la creatividad y eficiencia en el desarrollo de videojuegos. Gracias a sus características, C# permite a los desarrolladores llevar sus ideas a la realidad de manera efectiva y con alto rendimiento, contribuyendo al éxito de innumerables proyectos en la industria del gaming.

* + 1. **Requisitos**

**Requisitos mínimos.**

Sistema Operativo: Windows 7 o superior

Procesador: CPU con soporte para SSE2

Memoria: 2 GB de RAM.

Gráficos: Compatible con DX10, DX11, y DX12

Almacenamiento: 1 GB de espacio disponible.

Sonido: Tarjeta de sonido compatible con DirectX.

Que sea de dos de que pueda andar patatin.........

* 1. **Funcionalidad**

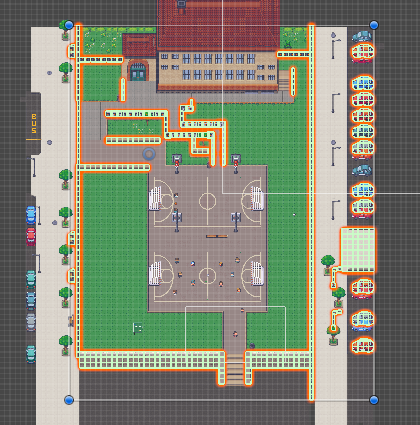
Trequisitos de usuario, interacciones, andar,etc conversaciones , cambios de mapas, npcs....

Aqui se explican las funciones de cada escena

La escena “3. Patio superior” contiene diferentes elementos desde el propio mapa “Grid” hasta elementos interactivos como puede ser el npc de “Sergio”, npcs simulando un partido de fútbol en el primer campo de fútbol y el balón o puntos de interés como la mesa de ping pong. A continuación una imagen de la escena en cuestión:



En primer lugar destacaríamos el procedimiento por el cual se ha “dibujado” la escena, ya que en este caso contamos con distintas capas las cuales son: Suelo, Decoración(dentro de esta Colisión, Delante, Detrás) y Paredes. La capa de “Suelo” comprende toda la superficie de la escena por la cual podemos movernos y también se encuentras todos los elementos de las otras capas. La capa “Decoración” pose distintos elementos del entorno como las porterías de fútbol, los bancos, el edificio, las canastas, farolas, etc. En esta capa encontramos diferentes “sub capas” las cuales ayudarán a nuestro personaje a diferenciar cuando se encuentra delante o detrás de los distintos elementos decorativos de la escena generalmente, como de la misma forma se hace en otras escenas del juego. Por último tendríamos la capa “Paredes” la cual en esta escena ha sido usada para los elementos como vallas y muros, los cuales serán los encargados de limitar el área donde el usuario pueda moverse, separando así el patio del instituto de la calle y para algunos de los coches del lado derecho de la escena, ya que así se colocaban por detrás entre ellos como se puede ver en la siguiente imagen.

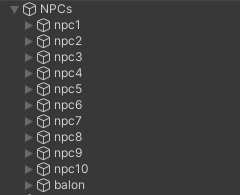


Profundizando más en la escena podemos encontrar al inicio de esta el primer elemento interactivo de la misma, este se trata de un npc (Sergio) el cual tiene un pequeño dialogo donde nos avisará de nuestra situación y advierte del peligro de más adelante, la implantación de este dialogo sigue la misma práctica que el resto de sistemas de diálogo que encontraremos en el juego. Usando así el script de gestion de diálogos creado en el proyecto y los diferentes elementos relacionados con dicha funcionalidad, tales como el objeto “Personaje” y “Conversación”.

Seguido de este npc podemos encontrar a la izquierda una mesa de pin-pong donde también podremos desbloquear un pequeño texto si nos acercamos lo suficiente a modo de “punto de interés” para el jugador, funcionando de la misma forma que los diálogos que encontramos en los npcs.

Siguiendo con el orden ascendente de elementos que encontramos en la escena ahora encontramos los npcs que simulan estar jugando un partido de fútbol. Todos los elementos de esta sección de la escena los podemos encontrar organizados en un objeto a modo de carpeta llamad “NPCs” ya que consta de once npcs incluyendo el balón. Npcs los cuales si nos acercamos lo suficiente hasta chocar con alguno de estos sufriremos cierta cantidad de daño que se verá reflejada en los corazones que representan la vida del personaje principal situados en nuestra interfaz de ususario (UI). Cada uno de estos incluyendo el balón contienen los mismos elementos, los cuales son; un Sprite Renderer para poder darle una imagen al objeto, Box Collider 2D para poder realizar colisiones con ellos, Animator que contendrá las animaciones correspondientes de su respectivo personaje y por último el script que se encargará del movimiento aleatorio y velocidad de los objetos (Random\_Mov).

En este script empieza con unas coordenadas preestablecidas las cuales harán que todos los npcs que estén “jugando al fútbol” tomen esas coordenadas como referencia para que su posición nunca se encuentre fuera de las mismas, creando así un área controlada donde transcurra todo el rato con seguridad el funcionamiento de este “minijuego”. Este comportamiento dentro del script se consigue a raiz de generar constantemente una nueva posición aleatoria, si esa posición no se encuentra dentro de las coordenadas anteriormente mencionadas rápidamente se vuelve a generar otra posición aleatoria así hasta que la nueva posición aleatoria generada exista dentro de las coordenadas establecidas. Una vez la posición aleatoria generada sea válida se usa una variable de velocidad (la cual se puede modificar en el inspector de unity) para mover a dicha velocidad el objeto hasta la dirección de la nueva coordenada y a su vez se analiza la dirección (derecha, arriba, izquierda, abajo) para colocar la animación correspondiente a esa dirección. Para intentar añadirle más realismo los npcs tienen distintas velocidades entre ellos y el balón (que es tratado de la misma forma) obtiene la velocidad más rápida y unas coordenadas más limitadas.



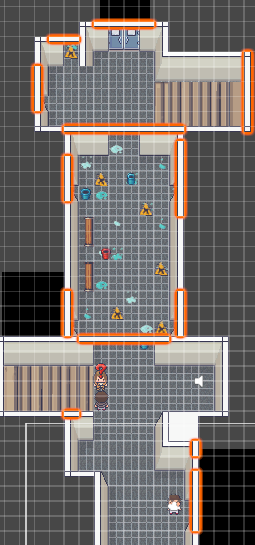
Siguiendo con los diferentes elementos de la escena en el siguiente campo después de pasar el “minijuego” del partido de fútbol encontramos dos npcs más a modo decorativo, los cuales simulan jugar al baloncesto pasándose un balón de baloncesto en bucle, el cual logra la animación en un mismo archivo .anim que combina la animación del balón en su propio eje (rotación) como la animación vertical del mismo (eje y). En este punto de la escena solo nos quedaría descubrir otro punto de interés como el que se menciono anteriormente con la mesa de pin-pong, pero esta vez lo encontramos si nos acercamos al pequeño jardín vallado de la esquina superior derecha de la escena. Donde una vez más si nos acercamos lo suficiente encontraremos el icono de interrogación dando la oportunidad de descubrir un pequeño diálogo.

Como elementos destacable de la escena ya solo quedaría mencionar los puntos de “Spawn” de la escena, los cuales podemos encontrar el spawn para ir al patio inferior y el spwan para ir al edificio y entrar en su interior, estos dos contienen un collider que sirve como detección para que el cambio se escena se produzca gracias al script de cambio de escena creado en el proyecto (Cambio\_Escena). También están establecidos en la escena la posición donde nuestro personaje principal aparecerá dependiendo de la escena de la que salgamos previamente.

La escena “4. Pasillo Inferior” contiene diferentes elementos desde el propio grid con sus diferentes capas usado para el dibujado y la creación de la escena (como en las demás escenas) hasta su propio “minijuego” y elementos interactivos.A continuación una imagen de la escena en cuestión:



En esta escena repetimos el procedimiento de capas con el cual conseguimos el “dibujado” deseado para la escena repartiendo los distintos elementos de la escena entra estas. Esta escena cuenta con las siguientes capas; Paredes, Suelo(dentro de esta encontramos otra capa llamada “agua”), Horizontal y por último ParedAlto. En la capa “Paredes” podremos observar como se compone de todas las paredes del espacio interior en el que nos encontramos, estableciendo estas como tal gracias a su “Tile Map Collider 2D”. En la capa “Suelo” encontramos una vez más toda la superficie por donde nuestro personaje principal podrá desplazarse, dentro de esta capa “Suelo” podemos ver como tenemos otra capa llamada “agua” la cual es usada para los charcos que se aprecian en el pasillo, ya que estos deben actuar en la misma capa sobre el suelo ya dibujado teniendo el mismo comportamiento que este y sin borrarlo, de esta manera conseguimos que los charcos de agua formen parte del suelo sin crear anomalías en el mismo. La capa “Horizontal” se encuentran unos elementos puntuales, ya que esta capa recibe ese nombre dado que al momento de crear la escena los sprites que se intuyen en la misma como escaleras y bancos se obtenían únicamente de forma vertical desde el Tile Palette. Por ello se creo esta capa la cual se roto para poder lograr que tanto las escaleras como los bancos estuvieran orientados de forma horizontal. Por último respecto a las capas de la escena encontramos la capa “ParedAlto” en la cual encontramos lo que podríamos identificar como el marco superior de todas las entradas de la escena, de esta forma se crea una visión área de la zona más intuitiva dando a entender rápidamente con que entradas y espacios contamos dentro de la escena, como se puede observar en la siguiente imagen.



Continuando con los elementos que componen la escena encontramos al inicio de esta dos npcs (los de Ricardo y AnaIsabel) los cuales una vez más como en el resto si nos acercamos lo suficiente saltará una interrogación roja la cual nos dará acceso presionando la tecla “E” a una pequeña conversación en la que están implicado tanto ellos como nuestro propio personaje principal, en esta nos indican precaución frente al obstáculo que se propone en la escena en una conversación entre ellos dos. Al igual que el resto de personajes con sistema de diálogos se sigue el mismo funcionamiento de objetos “Personaje” y “Conversación” gestionado por el script “S\_Dialogue Speaker”.

Si avanzamos un poco más en la escena con nuestro personaje principal entraremos en el “minijuego” de la escena, el cual consiste en chocar con las paredes de los objetos situados en el pasillo hasta encontrar la salida a este, descifrando un pequeño patrón para lograrlo. Afectando al personaje principal un comportamiento de movimiento distinto al que estábamos acostumbrados durante todo el juego, con la intención de dar a entender unas físicas que simulan un suelo mojado que por consecuencia resbala. Por esta misma premisa el jugador no podrá moverse mientras el personaje principal se encuentre en movimiento y solo será posible en el momento que el personaje principal logre chocar con alguna pared o elemento situado en el pasillo. El funcionamiento de esta mecánica de juego es dada gracias a dos scripts diferentes creados en el proyecto, los cuales son; Mecanica\_PI y Zona Activacion MPI.

En primer lugar el script de Mecanica\_PI es el encargado del movimiento que simula un suelo resbaladizo para el personaje principal, este script lo contiene el objeto del personaje principal junto al script de movimiento habitual que usamos en el resto del juego, En este srcript uno de los elementos protagonistas es una variable booleana que nos indicará si el personaje esta en movimiento o no, para esto se hace una comprobación de la velocidad del personaje, lo cual si es diferente de cero querrá decir que nuestro personaje se esta moviendo y por consecuencia esta variable será “false”, entonces no dejará cambiar la dirección de movimiento del personaje. La variable cambiará a “true” en el momento que esta comprobación sea “false”, y entonces el jugador ya podrá seleccionar una nueva dirección de movimiento para el personaje principal y así seguir desplazándose. La condición para que nuestra comprobación sobre la velocidad del personaje sea “false” la conseguiremos con un método OnCollisionEnter2D donde se comprobará si el personaje principal a colisionado con algún elemento que contenga el tag “obs” (de obstáculo), en caso de que esto se cumpla nuestra variable booleana pasara a “true” verificando que el personaje principal esta parado, y permitiendo al jugador la selección de una nueva dirección de movimiento. Este tag “obs” lo contienen todos los objetos que vemos posicionados estratégicamente en el pasillo.

En segundo lugar el script llamado Zona Activación MPI se encontrará situado en un objeto BoxCollider2D el cual cubrirá todo el largo y todo el ancho del pasillo, creando así una areá controlada donde queremos que ocurra nuestro evento del minijuego. Lo que hace este script es sencillo ya que con los métodos OnTriggerEnter2D y OnTriggerExit2D conseguimos detectar cuando nuestro personaje principal entra y sale del área controlada donde queremos que se produzca el evento. De esta forma cuando el personaje entra en nuestro BoxCollider2D se activa el método OnTriggerEnter2D el cual dejará activo en nuestro personaje el script Mecanica\_PI dando ese comportamiento resbaladizo al movimiento del personaje y desactivará de nuestro personaje el script de movimiento habitual (Per\_Movimiento), de la misma forma cuando nuestro personaje salga de este BoxCollider2D que conforma nuestro área del minijuego se activará OnTriggerExit2D que hará lo mismo que el OnTriggerEnter2D pero viceversa, dejando activado en nuestro personaje el script de movimiento normal y el de nuestra mecánica de movimiento resbaladizo desactivado. Enla siguiente imagen podemos ver como el BoxCollider2D delimita la zona donde se cumple el comportamiento mencionado anteriormente.



Para terminar con los elementos de la escena tendríamos que destacar los puntos de cambio de escena que contiene la escena, contiene tres. El primero para la entrada y salida desde la escena del patio superior, ubicado en la entrada, el segundo en la primera puerta del pasillo a la izquierda dando pie esta a una ubicación secreta del juego y por último ubicado en las escaleras del fondo de la escena, las cuales daría lugar a la escena del pasillo superior. En la siguiente imagen se muestra estos distintos puntos de “Spawn”.



# Calle

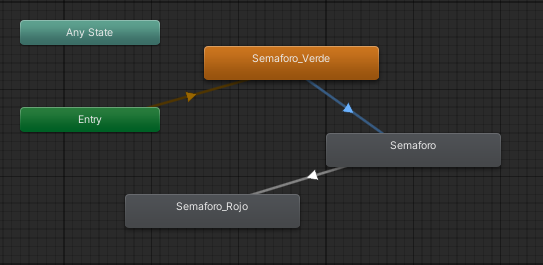


La escena de *Calle* es la primera escena del videojuego.

Esta escena contiene los siguientes elementos básicos: un tilemap, los NPCs, el personaje manejable, los elementos de aparición y cambio de escena y demás componentes de funcionamiento.

Además de esto, esta escena cuenta con unos semáforos y una serie de vehículos que supondrán un pequeño reto al usuario además de dar realismo y ambiente a la escena.

Los semáforos son objetos simples que no requieren de un script controlador, ya que unicamente necesitamos que tras unos segundos, se inicie la animacion de cambio de color.



Esto se puede conseguir con el Animator  estableciendo una serie de repeticiones en la animacion estatica y una unica repeticion en la de cambio de color

Para los coches sí es necesario un script. En concreto en esta escena hemos usado dos. Un tipo de coche recorrerá las calles a una velocidad concreta y tras salir del mapa volverá por el otro lado, dando así la impresión de múltiples coches pasando.

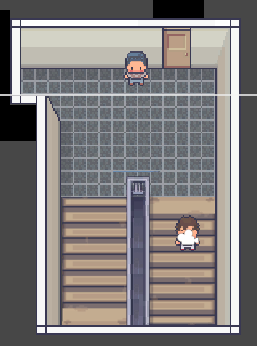
El otro tipo de coche será el responsable de aparecer en un tiempo concreto para cuando el semáforo se ponga en rojo, frenar delante del paso de cebra y dejar al jugador cruzar la calle.

Además de todo esto, esta escena es la única que no está rodeada completamente por paredes o muros, por lo que fue necesario tener también paredes invisibles, que se

comportan como paredes normales, pero las colocamos en la última capa, de forma que no se vean.

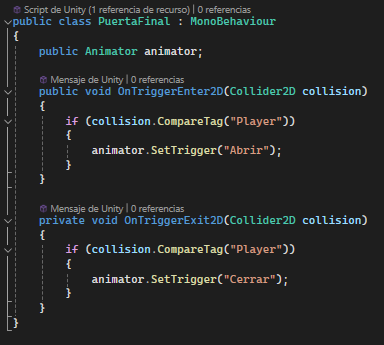
Por último está el autobús, que se comporta de forma semejante a los coches que frenan, lo único que este comienza estático y tras unos segundos arranca hasta desaparecer fuera del mapa.

Pasillo Superior

Esta es la escena más simple de todas, cuenta con su Tilemap, un diálogo y los elementos de cambio de escena.

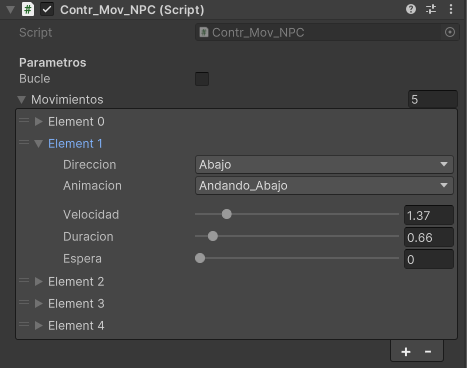
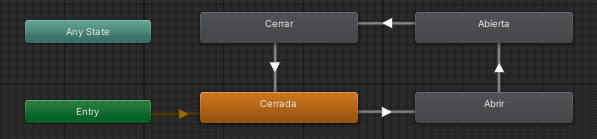
Cuenta también con paneles que permiten que el personaje no se vea a través de las paredes en algunos lugares. Estos paneles son necesarios en las escenas de interiores para evitar que al jugador se le vea por espacios en los que la perspectiva no permitiría su visualización.



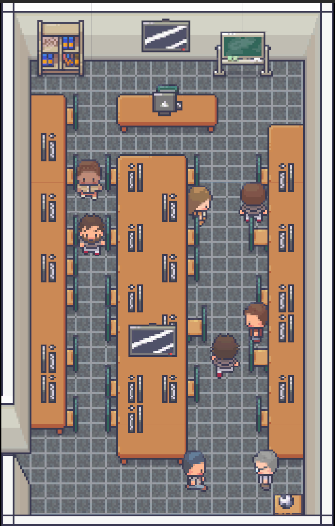


Por último está la puerta que tiene un comportamiento muy simple.

Cuando el jugador entra en el espacio cercano a ésta envía un trigger a su animator y lo mismo al salir, de forma que tras recibir el activador la animación se realiza una vez y queda a la espera en la siguiente animación del otro activador

# Escena final



Esta escena es una cinemática no jugable, por lo que no comparte la mayoría de elementos con las otras escenas.

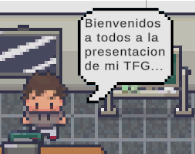
Sí requiere de un Tilemap, el cual es el que más capas necesita de todos, y unos NPCs.

Para el funcionamiento de los NPCs tenemos un script muy flexible que hemos reutilizado para el resto de NPCs en las demás escenas.

El script recibe una lista de *Movimientos*, que se realizan uno detrás de otro.

Cada Movimiento tiene una información concreta, y de esta forma podemos indicar a un objeto que se mueva de una determinada manera de una forma muy cómoda y ajustable.

Además, si todos los NPCs tienen los mismos nombres de animaciones, con un campo de elección podemos indicar cual se ejecuta en cada movimiento.

  
Por último están los elementos emergentes que aparecen al final de la cinemática, que son un bocadi llo de dialogo, un fundido en negro y la imagen de la nota final.

Estos elementos tienen un script que tras unos segundos de espera, comienzan a subir la opacidad del elemento poco a poco.

Para la imagen final, además de lo anterior, el script recibe una serie de sprites en un array y, dependiendo de la nota del jugador, se usa una u otra imagen.

* 1. **Características.**
     1. **Diagramas E­R**

**No tiene sentido aqui**

* + 1. **Diagramas de clases**

Texto

* 1. **Usabilidad**

Texto

* + 1. **Diseño de interfaces.**

Texto cada uno de los niveles,las imagnes

* + 1. **Casos de uso.**

Texto (Que puede hacer el usuario) que hace cuando se pulsan las teclas

* 1. **Portabilidad.**

Texto

* + 1. **Plataformas.**

Texto

* 1. **Rendimiento.**

Texto

* + 1. **Pruebas realizadas**

Tprobado en diferentes ordenadores

1. **Conclusiones:**

Texto

1. **Referencias bibliográficas:**

Texto.direcciones(asets)

1. Control de versiones: githubb
2. Software utilizado
3. Compra de assets