La Pluma del Tiempo

VIDEOJUEGO EN 2D Y 3D

Álvarez Álvarez, Enrique

Castro López, Miguel Ángel

Mosquera Pérez, Miguel

Trabazo Sardón, Diego

Contenido

1. Desarrollo artístico

1.1. Antecedentes

Los hechos narrados ocurren a principios del año 2017 en un centro universitario de una conocida ciudad gallega.

El alumno Lorem Ipsum se encuentra en la facultad de informática. Debería estar haciendo un examen de su asignatura fatal, la de los videojuegos. No era la primera vez que se presentaba y nunca conseguía que en el videojuego los enemigos se comportaran de forma inteligente, todos se caían en precipicios. Pero el profesor Arhashem Ajarsem no llegó a presentarse. Al parecer estaba enfermo y el examen se suspendió. Decide pasar por la cafetería para tomar un café y aguardar un poco a la espera de novedades, cuando de repente, le parece ver a ese mismo profesor entrando en un aula de la primera planta. Sorprendido, lo sigue… Allí encuentra al profesor… Parece que mal herido, lleva algo oculto en su abrigo. Lorem se acerca para saber que le ocurre. Parece estar en las últimas. – ¡He fallado! – Repetía una y otra vez. Lorem se prepara para pedir ayuda, pero el profesor Ajarsem lo agarra bruscamente. – Tienes que ayudarme, no hables con nadie, necesito que hagas algo. – Susurra. – Es importante, busca la puerta. En la planta 0, pasando los laboratorios, una puerta negra cerrada con código. Introduce el código 1010101111101011 para abrirla. Busca al Líder y cuéntale lo que ha pasado. Debes encontrarlo. No hables con nadie más. Está en el año… – Y de repente la voz del profesor se apagó…

Lorem Ipsum se queda paralizado. El profesor yace muerto delante de él. – ¿Qué debería hacer? – Se queda pensando. – No hables con nadie, encuentra al Líder… ¿A que venía todo eso? – Pero la parte que más lo inquietaba era – En el año… – ¿Acaso no sabía que día era? – De repente se percata de que lo que tenía oculto en su abrigo era una especie de huevo gigante. Lorem no era capaz de reconocer al animal al que pertenecía. Sacó su Smartphone para pedir ayuda, pero no marcó ningún número. Tras pensar en ello, decidió que primero iría a buscar la puerta de la que hablaba el profesor. Si no existía o no conseguía entrar, entonces pediría ayuda. Su curiosidad lo tenía en vilo.

Lorem se dirige a la zona donde supuestamente estaba la puerta que mencionó el profesor con sus últimas palabras. Efectivamente, allí había una puerta negra que jamás había visto en todo el tiempo que había sido alumno de esa facultad, algo que le sorprendió, conocía la facultad como la palma de su mano. Confuso, introduce el código y la puerta se abre. Allí se encuentra un largo pasillo iluminado que deja ver una serie de puertas. Se introduce en el pasillo e intenta abrir cada puerta con la que se encuentra, pero parecen estar todas cerradas. Cuando alcanza el final del pasillo y estaba a punto de darse por vencido, intenta abrir la última de las puertas y esta se abre. Lo que allí se encuentra lo deja sin palabras. Era una habitación esférica, totalmente blanca y vacía, a excepción de un gran arco de aspecto antiguo que desprendía un extraño brillo de su interior.

Sin previo aviso se ve sorprendido por una misteriosa voz a su espalda que le pregunta con autoridad: – ¿Quién coño eres? ¿Cómo has entrado aquí? – Lorem se gira sobresaltado. La voz provenía de un hombre mayor con una mirada agresiva. Con voz temblorosa, Lorem le responde dubitativo: – Yo… He traído esto – Mostrándole el extraño huevo. – Un hombre que acaba de morir me ha enviado aquí. ¿Es usted el Líder? – Le preguntó. El hombre le arranca el huevo de las manos y lo mira pensativo… Vuelve a fijar su mirada en Lorem y le responde: – Sí, soy yo. Cuéntame lo que ha pasado. Lorem le cuenta al Líder todo lo sucedido. Tras escuchar atentamente y después de una

contundente pausa, al Líder se le ilumina la cara y le dice a Lorem: – Ese hombre al que tú conoces como tu profesor, trabaja para mí. Para que lo entiendas, somos viajeros del tiempo. Nuestro objetivo es mantener el orden cronológico intacto. Sé que es un poco precipitado, pero ahora que Arhashem Ajarsem ya no puede trabajar con nosotros, me gustaría pedirte que continúes su labor, de alguna forma te ha elegido a ti… ¿Aceptas el reto? – Le pregunta el Líder. Nuestro intrépido aventurero no duda en contestarle inmediatamente: – Sí – Dice con firmeza. El Líder sonríe y le dice: – Aquí tienes tu primera misión.

1.1.1. Personajes

- **Arhashem Ajarsem**, el profesor. Es un hombre inteligente, reconocido investigador, aburrido de su trabajo como docente, motivo por el cuál buscó nuevas experiencias como viajero en el tiempo.
- **El Líder**. Es un hombre frío y serio, pero a la vez, amable. Dirige una sociedad secreta encargada de velar por la integridad de la historia.

1.2. Desarrollo videojuego 2D

1.2.1. Personajes

1.2.1.1. Personaje jugable

- **Lorem Ipsum**, nuestro protagonista. Es un joven aventurero, muy inteligente, cuyas aficiones se centran en la resolución de acertijos. Es muy curioso, siempre en busca de nuevas experiencias, y cae en el aburrimiento cuando no tiene ningún reto para afrontar.

1.2.1.2. Enemigos

- **El Hombre Misterioso**, el antagonista. Físicamente tiene características poco comunes en la raza humana. Es un hombre enigmático. Parece tener un plan para modificar la historia. Sus motivos no son claros. Su verdadera identidad inicialmente es desconocida.
- **Dinosaurios primera misión:**
  - **Velociraptors.** Veloces dinosaurios que muerden. Escapan del fuego.
  - **Pterodactylus.** Dinosaurios voladores que usualmente carga con diferentes objetos y los arroja al protagonista.
  - **Tricetatops.** Dinosaurios de gran envergadura y pacífico al que se ha de alimentar para poder avanzar. Si es atacado se enfurecerá.
  - **T-rex [Final].** Colosal dinosaurio al que hemos de aturdir para devolver su huevo al nido.
- **Las Piratas.** Violentas enemigas que intentan matar al protagonista. Van armadas con espadas.

1.2.2. Lugares

- **Cuartel de La Organización**. Base de operaciones de los viajeros del tiempo. Es el lugar secreto donde el protagonista inicia su aventura.
- **La Prehistoria**. Escenario jurásico con montañas cercanas y lejanas y cielo a diferentes velocidades, con árboles y otros elementos al frontal que tapan la escena trasera (y todos

sus elementos, como los personajes). En este escenario se desarrolla la primera aventura de Lorem, donde luchará contra los dinosaurios.

- **El barco**. Barco de madera en el que navega Colón, en el que Lorem tendrá que luchar contra las piratas que intentan abordarlo. Se apreciará oleaje con movimiento (Scroll vertical y horizontal repetitivo y constante). Al fondo se apreciará mar con barcos con movimientos leves para aportar realismo (se mueven horizontalmente con el scroll del fondo y levemente verticalmente simulando efecto de navegación). El personaje tendrá que esquivar obstáculos como barriles y subir y bajar escaleras para poder rescatar a Colón.

1.2.3. Objetos destacados

- **Misión 1:**
  - **El huevo misterioso**. Es el huevo con el que Lorem tendrá que viajar a la prehistoria para entregar al T-rex en su primera misión.
  - **Frutas**. Piezas de fruta que el protagonista se encontrará por la selva y puede comer para recuperar vida.
  - **Trampas naturales** en el terreno que pueden bajar la vida del protagonista.
- **Misión 2:**
  - **Botellas de ron.**
  - **Plataformas**: cuerdas, barriles, mástiles con plataforma…
  - **Barriles Explosivos** que explotan al dispararles con un trabuco.

1.2.4. Armas

- **Misión 1:**
  - **Piedras.** Aturden y ahuyentan a los dinosaurios (excepto a los Tricetatops, que se enfurecerán y atacarán al protagonista). Hay unidades limitadas, pero se pueden volver a recoger.
  - **Palo con fuego.** Palo de madera al que se le puede prender fuego (ha de encenderse) y se consume lentamente.
- **Misión 2:**
  - **Sable.** Arma blanca que el personaje usa con destreza para eliminar a sus enemigas piratas.
  - **Trabuco.** Arma de fuego de un solo disparo, escasa fiabilidad y lenta recarga.
  - **Bolas de cañón**. Salen de los barcos enemigos. Se observará una sombra cuando son lanzadas y posteriormente un impacto. Si el protagonista está cerca puede bajarle la vida, si recibe el impacto directo se quedará sin vida.

1.2.5. Guión

1.2.5.1. Primera misión: El huevo del Jurásico

El Líder le explica a Lorem que el huevo que portaba el profesor había sido robado a un dinosaurio de la época prehistórica. Lorem deberá viajar a la prehistoria para devolver el huevo al nido al que pertenece. Por el camino se encontrará con dinosaurios agresivos, que sorprendidos por su presencia, lo atacarán, por lo que Lorem deberá luchar contra ellos para cumplir su misión. Para concluir la misión, Lorem se encontrará con el dinosaurio final, al que tendrá que aturdirlo para poder devolverle su huevo. El dinosaurio que nacerá de él marcará el curso de la evolución, llegando a afectar a la genética humana en el futuro, entre otras cosas, en caso de no devolverlo, los humanos tendrán una piel distinta, detalle que caracteriza a nuestro enemigo.

1.2.5.2. Segunda misión: El rescate de Cristóbal Colón

En esta aventura, Lorem tendrá que luchar contra decenas de piratas que están abordando el barco de Cristóbal Colón, influenciadas por el Hombre Misterioso, para impedir el descubrimiento de América. Si Lorem consigue derrotar a las Piratas, tendrá que luchar contra el Hombre Misterioso. Tras ganar la batalla, mal herido el Hombre Misterioso, logra escapar por un portal espacio-temporal.

1.3. Desarrollo videojuego 3D

1.3.1. Personajes

1.3.1.1. Personaje jugable

- **Lorem Ipsum**, nuestro protagonista llega a esta nueva aventura tras haber seguido al enemigo por diferentes épocas. La experiencia adquirida lo ha convertido en un experto en temas de infiltración. Ahora, deberá infiltrarse en dos nuevas aventuras para conseguir descubrir la identidad del Hombre Misterioso.

1.3.1.2. Enemigos

- **El Hombre Misterioso**, continua con sus intentos de modificar la historia. Finalmente se descubrirá que este ser resultará ser el protagonista mismo pero de una realidad alternativa.
- **Los Nazis:**
  - **Soldados.** Militares integrantes del Partido Nacionalsocialista Obrero Alemán, que defienden la base militar. Podrían encarcelar a nuestro protagonista si llegan a descubrirlo.
  - **El General.** El dirigente de la base militar. Carácter duro. Es el que porta el código de acceso a la habitación donde se encuentra la máquina Enigma.
- **Ambiente de la discoteca:**
  - **Los clientes.** Personajes habituales de una noche de los 80.
  - **Empleados (camareros, guardias de seguridad, prostitutas…).** Trabajan para el villano.

1.3.2. Lugares

- **La base militar**:
  - **Zona exterior**. Cerrada con una verja. Se pueden apreciar coches militares, soldados haciendo guardia y el edificio.
  - **Sala de entrada**. Estancia de gran superficie en la que se puede apreciar una estatua gigante de Adolf Hitler y banderas con esvásticas del Partido Nazi a lo largo de toda la sala.
  - **El despacho del General**. Despacho donde el General guarda la clave de acceso a la sala secreta.
  - **Sala secreta.** Protegida por una puerta blindada con código de seguridad. En el interior se encuentran objetos de gran valor, entre ellos la máquina Enigma.
  - **Otros espacios**: sala de armas, habitaciones, un comedor, sala de tiro…
- **La discoteca**:
  - **Zona exterior**. Ambiente urbano con la fachada de la discoteca en el plano central y un callejón.
  - **El callejón**. Zona lateral de la discoteca levemente iluminada. Es a donde va a dar la salida de emergencia. Están los contenedores de basura, el cuadro eléctrico…
  - **La entrada**. Pequeña sala donde hay unos sofás a la derecha y a la izquierda el ropero y una máquina de tabaco. De frente se encuentra la puerta que da acceso al interior.
  - **El interior**. Una pista de baile ocupa todo el centro, con iluminación propia de una discoteca. Hay dos barras: una a la izquierda y otra a la derecha. Los baños se encuentran al fondo a la derecha, y en el fondo izquierdo la salida de emergencia. Al fondo de todo hay una puerta que da acceso a la zona privada, en la que se sospecha que está el Hombre Misterioso.
  - **La Sala del Tiempo**. Es una habitación subterránea en la que únicamente hay un portal. Allí encontraremos al fin al Hombre Misterioso. Es el último escenario del juego.

1.3.3. Objetos destacados

- **Misión 1:**
  - **La llave de entrada a la base.** Se encontrará perdida en las inmediaciones de la base.
  - **El papel con el código.** Se encuentra oculto en el despacho del General.
  - **La máquina Enigma.**
  - **El walkie-talkie.** Se encontrará extraviado en la base militar. Lorem lo necesitará para llevar a cabo su misión.
  - **El mechero.** Lo porta Lorem y lo necesita para incendiar el coche.
  - **Productos de limpieza.** Los porta uno de los soldados. Son inflamables.
  - **Cajas de madera.** Montones de cajas de madera que se encuentran en el exterior de la base militar.
  - **Coches.** Vehículos militares que se encuentran en el exterior de la base militar.
- **Misión 2:**
  - **Alicates.** Útiles para poder cortar el suministro eléctrico.
  - **Botellas.** Se pueden usar como arma.

1.3.4. Guión

1.3.4.1. Primera misión: En busca de Enigma

En la primera misión de esta nueva aventura nos encontraremos ante una base militar nazi. Nuestro objetivo es infiltrarnos para entrar en la sala secreta y llevar la máquina Enigma sin ser detectado, y así ayudar a los aliados a ganar la guerra, ya que nuestro enemigo ha evitado el asalto al submarino alemán U-110 donde se consiguió un ejemplar de Enigma, robando el informe que permitió a la Royal Navy localizar este submarino.

En el exterior nos encontraremos a un soldado que ha perdido la llave de la entrada. La encontraremos entre unas cajas a la izquierda de la base.

Una vez en el interior, todo pasa por conseguir que el General salga de su despacho para colarnos en él. Para ello, en primer lugar, debemos encontrar un walkie-talkie perdido y cogerlo sin que nadie nos vea. Una vez lo tengamos, debemos enviar una orden falsa al soldado que está limpiando la estatua de Hitler para que se vaya a otro lugar. Cuando el soldado se ausente deberemos robarle alguno de los productos de limpieza inflamables. Una vez en nuestro poder, deberemos incendiar uno de los vehículos que hay en el exterior y escondernos tras las cajas esperando a que salga el General. Una vez esté fuera deberemos entrar en su despacho, coger el papel con la clave y acceder a la sala secreta. Aquí terminaría nuestra misión. El nivel será imposible de pasar si no somos sigilosos.

1.3.4.2. Segunda misión: Bailando en los 80

En esta última misión deberemos descubrir a nuestro enemigo, que se encuentra oculto en una habitación secreta dentro de una discoteca.

En primer lugar, necesitamos en primer lugar entrar en la discoteca. Una vez dentro, deberemos encontrar una caja de herramientas que se encuentra oculta en una de las dos barras (no sabemos en cual), por lo que deberemos distraer al camarero para poder acceder al interior de la barra. Para ello, tenemos que manchar al camarero con una bebida para que tenga que ir a cambiarse de ropa y entrar a buscar en ese momento. Ahora, deberemos utilizar un alicate de la caja de herramientas para cortar

los cables del cuadro eléctrico del callejón y dejar al local sin suministro eléctrico. Los guardias de seguridad se separarán. Deberemos dejar inconsciente a uno de ellos para utilizar su vestimenta y poder acceder a la zona privada.

Al entrar en la habitación secreta descubriremos un túnel temporal por el cual viaja nuestro enemigo. Lo encontraremos y nos contará una historia diferente a la que conocemos. En realidad la línea temporal que conocemos está modificada. El Hombre Misterioso no intentaba cambiar el pasado, sino restaurarlo, al menos, así se lo cuenta a Lorem:

"Originalmente, Colón nunca descubrió América, EEUU nunca llegó a existir como tal y las grandes potencias mundiales eran China y Alemania. La Organización modificó ese hecho, con tu ayuda, para sus propios beneficios.

Enigma nunca fue descifrado. Los nazis ganaron la guerra y conquistaron todo el mundo. De nuevo, la organización se las ingenió para cambiar el rumbo de la historia según sus intereses.

¿Por qué? ¿Crees que sus acciones cambian la historia a mejor? No para el futuro. Su verdadera intención es mover los hilos del mundo, dominarlo por completo. Para ello han ido encauzando la historia para que pueda llegar su momento. Para hacerse con el poder. No les ha sido complicado conseguir puestos importantes en las distintas democracias del mundo, siendo EEUU su mayor baza. Teniendo en su mano el poder político, militar y económico, junto a la capacidad de viajar en el tiempo, el mundo se arrodillará ante ellos. Como ves, todo el pasado se ha modificado para llevar los EUA al poder. La derrota de los nazis era un hecho fundamental, pues en un tiempo anterior éstos llegaban a descubrir la organización y eliminarla casi por completo. Dos organizaciones intentando dominar el mundo… una tiene que perder.

Te preguntarás que tiene que tiene que ver el huevo del dinosaurio en esta historia. Bueno, eso sí es culpa mía. Necesitaba desenmascarar a La Organización ante el mundo y pensé que traer animales, objetos e incluso personas de otra época a la nuestra sería la forma de demostrar los viajes y pararles los pies a tiempo… Pero fracasé en mi primer intento, mi hombre fue descubierto y asesinado. Sí, Lorem, yo soy el líder. Todo este tiempo, has sido engañado.

¿Sabes? Casi me había dado por vencido, pero mira esto. -Dice señalando el portal-. *Llevo mucho tiempo buscando un segundo portal del que la Organización no tuviese ni idea, y al fin lo he encontrado. Puedo intentarlo de nuevo… Cambiar el futuro. Te aseguro que queda muy poco para el fin. ¿Qué me dices? Confía en mí, y te mostraré la verdad… para que la veas con tus propios ojos."*

En este momento deberemos tomar una decisión, de la que dependerá el final de la historia:

1. No creemos lo que nos ha contado el hombre misterioso y decidimos matarlo. Nos metemos en el portal. Se cierra la escena con el protagonista despertando en un futuro post-apocalíptico.
2. Esperamos unos segundos. No lo matamos y le preguntas al hombre: – ¿Quién eres? Y responderá, sonriendo: – Te lo contaré… a su debido tiempo. Entramos con el hombre en el portal y el juego termina ahí.

2. Desarrollo técnico videojuego 2D

2.1. Descripción del videojuego

2.1.1. Personajes

2.1.1.1. Personaje jugable

- **Lorem Ipsum**, nuestro protagonista. Es un joven aventurero, muy inteligente, cuyas aficiones se centran en la resolución de acertijos. Es muy curioso, siempre en busca de nuevas experiencias, y cae en el aburrimiento cuando no tiene ningún reto para afrontar. Su vestimenta y armas irán acordes al escenario en el que esté.

2.1.1.2. Enemigos

- **El Hombre Misterioso**, el antagonista. Físicamente tiene características poco comunes en la raza humana. Es un hombre enigmático. Parece tener un plan para modificar la historia. Su verdadera identidad inicialmente es desconocida. Aparecerá siempre al final de cada nivel y nuestra misión será derrotarlo.
- **Velociraptors:** Veloces dinosaurios que muerden. Atacarán al protagonista en la primera fase e intentarán impedir que llegue a encontrarse con el hombre misterioso.
- **Los Piratas.** Violentos enemigos que intentan matar al protagonista en la segunda fase. Van armados con espadas. Su poder y habilidad se distingue por su vestimenta, encontrando a nuestro paso desde el calmado pirata azul al desalmado rufián blanco.
- **Pirata interdimensional.** Un extraño pirata que posee capacidades sobrehumanas. Su piel brilla y cambia de color. Parece haber llegado desde otra dimensión. Solo parecerá en el modo arcade.

2.1.2. Objetos

- **Plataformas.** Para moverse por el escenario. Hay plataformas para la superficie del barco y del mundo jurásico, para las rampas y para las paredes. También en cuerdas y rocas.

Objetos disponibles en el modo arcade:

- **Corazones.** Usados para recuperar vida.
- **Espadas**. Aumenta el daño producido.
- **Botas**. Aumenta la velocidad del personaje.
- **PowerUp**. Aumenta la capacidad de salto.
- **Botella de ron**. Emborracha al personaje y le altera sus movimientos, confundiendo derecha e izquierda. La apariencia del personaje cambiará al entrar en estado de embriaguez.

2.1.3. Diseño

Partiendo del guión elaborado previamente, nos decantamos por diseñar un videojuego en 2D con perspectiva lateral, al considerar que era la opción más acertada y divertida. Inicialmente se había pensado desarrollar 2 fases diferentes, una en la cual visitásemos la época de los dinosaurios y otra un barco repleto de piratas. Finalmente, esa idea se fue desarrollando hacia algo más dinámico: Tenemos esas dos fases, pero se hacen más complicadas en cada partida y los enemigos no serán los mismos ni estarán en la misma zona. De este modo, el videojuego se hace muy rejugable.

Adicionalmente, se ha credo una nueva fase: 'modo arcade', en el cual el jugador debe resistir todo el tiempo que pueda una horda de enemigos que lo persiguen y atacan.

Inicialmente solo podremos jugar a la fase de los dinosaurios, al pasarla con éxito desbloquearemos la fase pirata, y al conseguir llegar al nivel 3 de dificultad (superando dichas fases en 3 ocasiones) se desbloqueará el modo arcade.

Otros elementos presentes son melodías diferentes para cada nivel y el menú, sonidos ambientales, animaciones a lo largo de los escenarios, como pueden ser olas en movimiento, lava volcánica o dinosaurios voladores que surcan un cielo que se va haciendo de día y de noche. Todos ellos en el afán de recrear las épocas en las que se ambienta y conseguir una mayor inmersión en el videojuego. Además, hemos ido incluyendo pequeños detalles para enriquecer la experiencia de juego, como pantallas de carga al inicio de cada fase, un fundido en negro con las letras "Game Over" cuando no superamos un nivel, y teñir de rojo la pantalla durante un breve momento cuando un enemigo nos ataca y quita un nivel de vida. Como ejemplo mostramos imágenes de estos dos últimos casos.

A continuación, se describirán las reglas y la mecánica de estos niveles.

2.1.3.1. Modo historia. Dinosaurios y piratas.

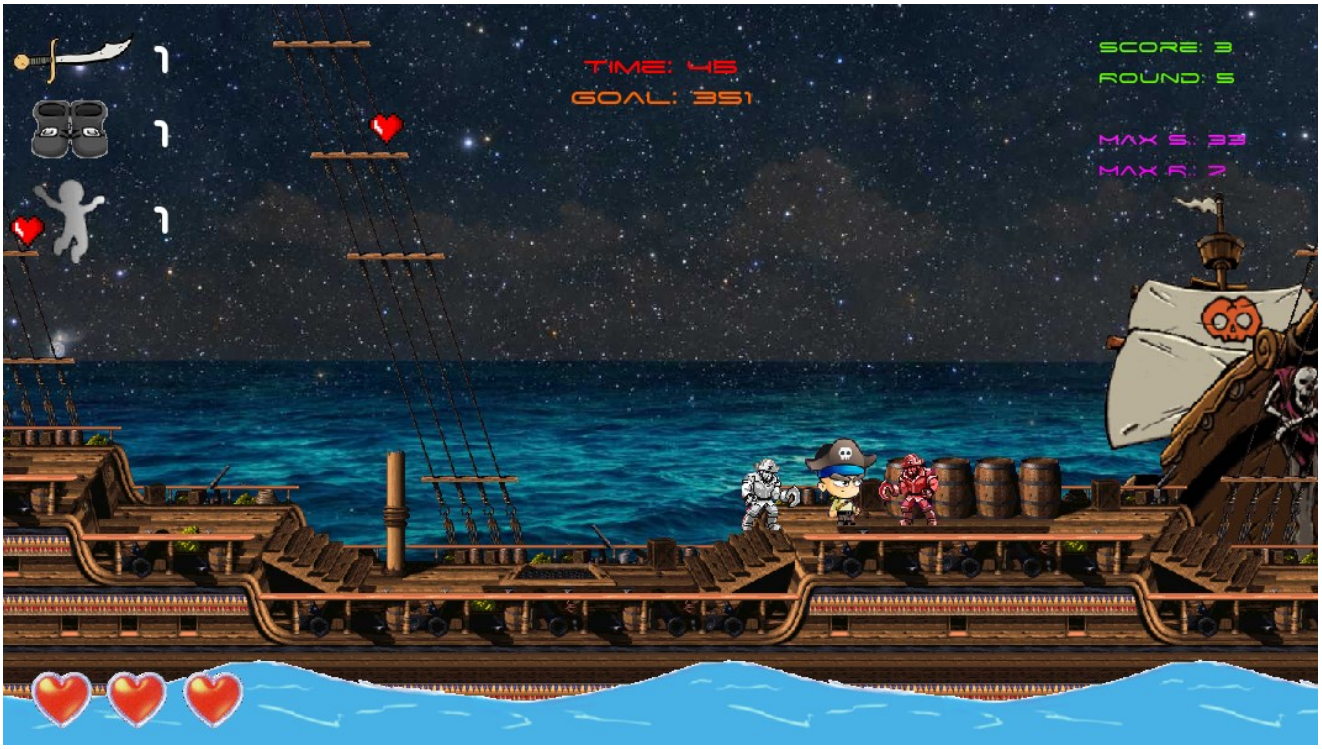
Las reglas y mecánica de juego son similares en estas dos fases. En ellas nos encontramos al principio de un escenario (una selva prehistórica y un barco respectivamente), por el cual debemos avanzar atacando hasta acabar con los enemigos o intentar esquivarlos mediante saltos y moviéndonos de forma inteligente. Nuestro personaje tendrá inicialmente 6 niveles de vida, representados por un panel con 6 flores situado en la parte superior izquierda de la pantalla. En el caso de que los perdamos todos, el juego se termina y debemos intentarlo de nuevo. Es importante remarcar que el nivel de vida perdido no se puede recuperar, por lo que el jugador debe gestionar bien la partida e intentar conservar el mayor nivel de vida posible. Al final de cada nivel nos encontraremos con 'el hombre misterioso', el villano y objetivo del juego, al cual debemos derrotar para superar la fase. A continuación, se muestran unas imágenes donde se puede apreciar los elementos descritos.

2.1.3.2. Modo Arcade.

Este nivel tiene sus propias reglas y la mecánica será diferente a las fases anteriores. En esta ocasión, nuestra vida está representada por corazones, situados en la parte inferior de la pantalla. A lo largo del escenario (que será el mismo de la fase pirata) tendremos corazones que van apareciendo de forma aleatoria y que podemos coger para recuperar la vida. Nos encontraremos también con otros objetos que pueden ayudarnos (espada, botas y powerUp) o perjudicarnos (botella de ron). El aumento de nuestras habilidades producida por dichos objetos se verá reflejado en la pantalla mediante un dibujo y un contador representando cada característica: poder de la espada, velocidad y capacidad de salto.

No comenzamos al principio del escenario, sino justo en medio. Además, tenemos varios contadores en la pantalla: en la zona central, un contador de tiempo, mientras a la derecha se encuentra la puntuación y la ronda en la que nos encontramos. La puntuación se incrementa con cada enemigo derrotado, y según el tipo de enemigo se sumará mayor o menor puntuación.

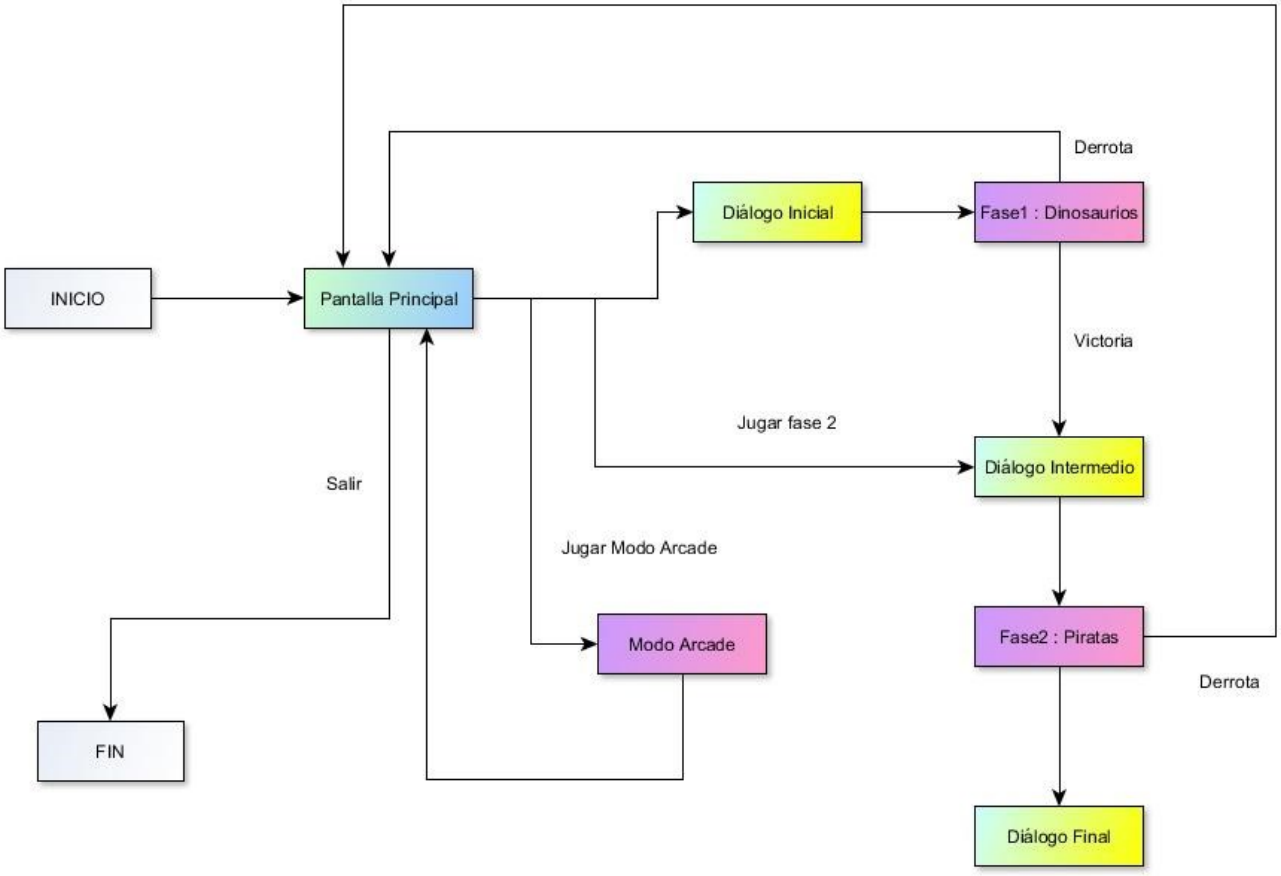
Nuestro objetivo principal será resistir el máximo tiempo posible con vida, esquivando o atacando a los enemigos que vienen a por nosotros, más numerosos y poderosos cuanto más avanza el tiempo.



El final de la partida llegará cuando nos quedemos sin corazones. Además, podemos ver, debajo de cada contador, nuestra mejor marca en todos los campos (mejor tiempo, score y round), para que así el jugador intente superarse a sí mismo en cada partida.

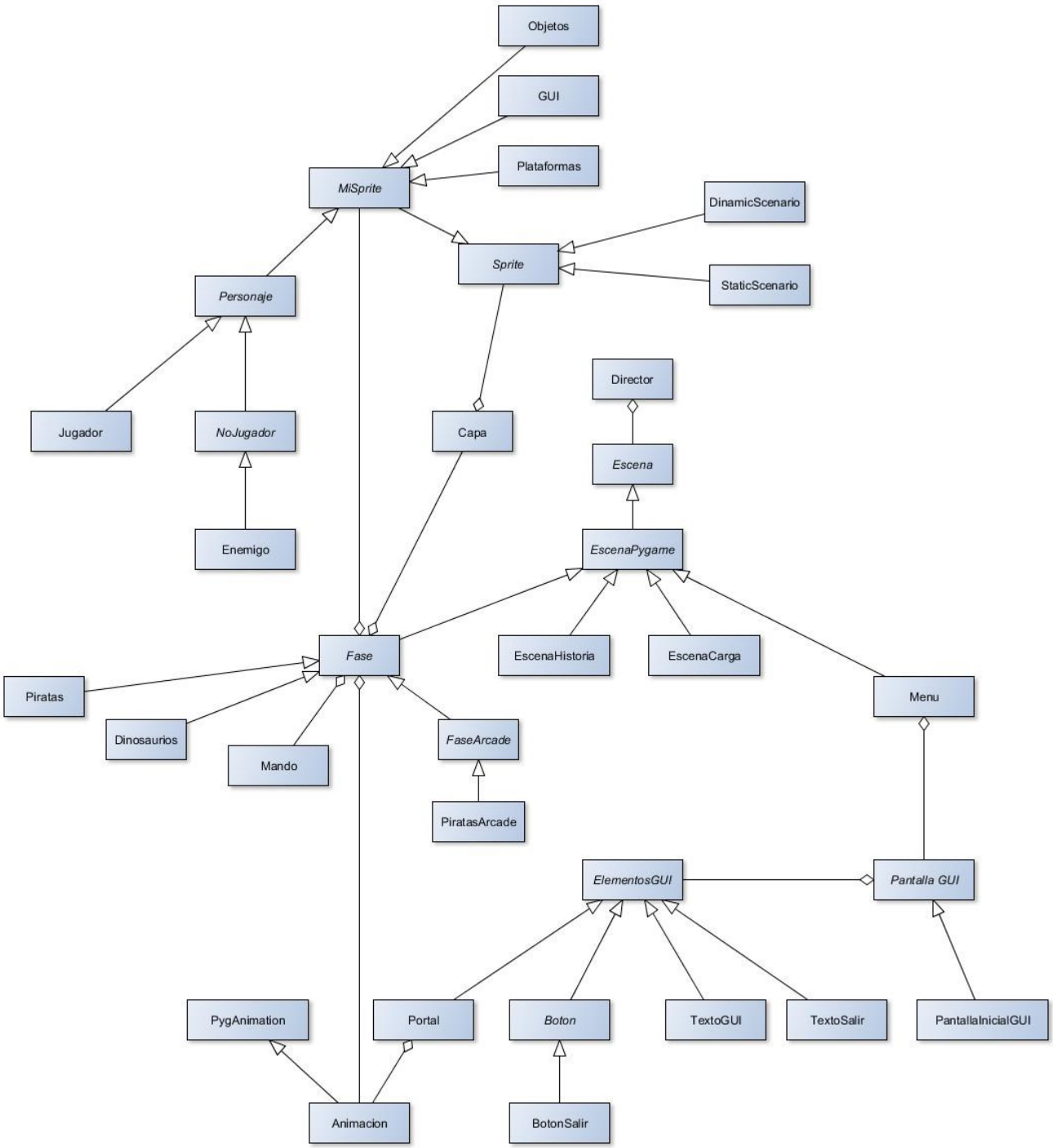
2.2. Flujo del videojuego

El flujo del juego sigue el esquema que se muestra a continuación.

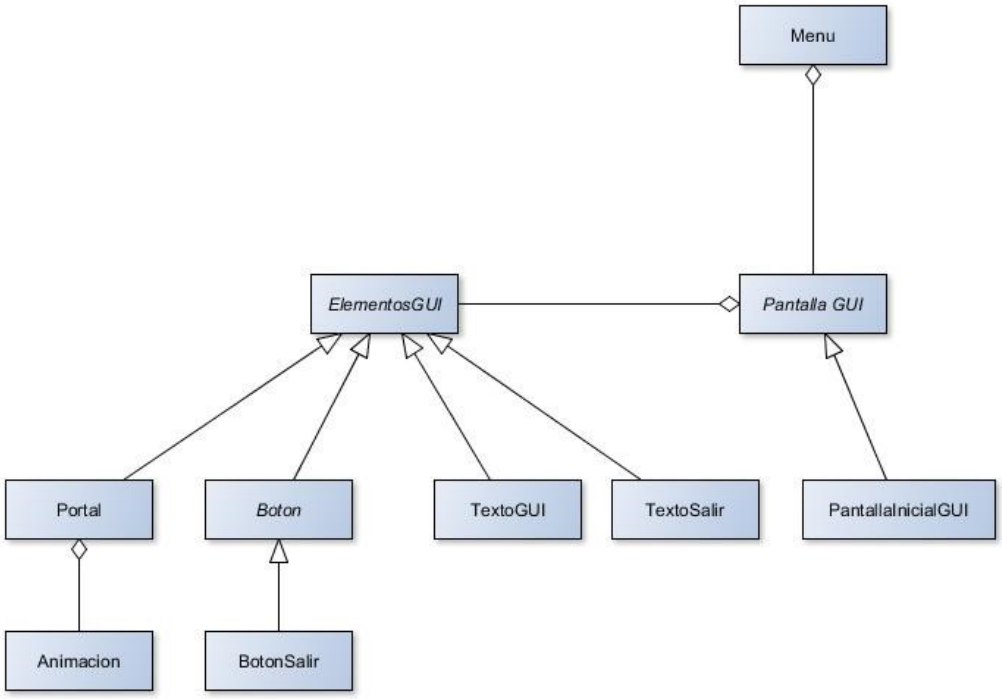


Cabe destacar que la Fase 2 sólo estará disponible desde la pantalla principal una vez superada la Fase 1 y, del mismo modo, el Modo Arcade queda desbloqueado al superar 2 niveles de la Fase 2.

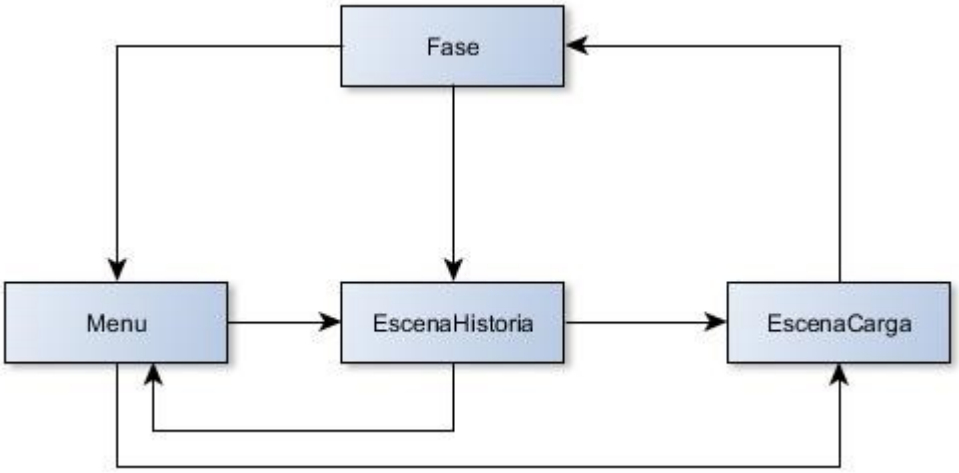
2.3. Diagramas de clases



2.4. Detalles de implementación



La implementación del menú no tiene grandes aspectos destacables, a excepción de la clase **Portal**, pues sus instancias serán las que nos lleven a las diferentes fases. Estas se representan por medio de animaciones, un método *focus* que activa dicha animación y el sonido asociado a la fase.



Desde los portales mencionados anteriormente se accederá a las distintas fases. Primero se carga sobre la pila del director la EscenaCarga que se encargará de cargar las imágenes de la fase de forma asíncrona para posteriormente lanzar la Fase. Pero antes de esto, y cargado por encima en la pila del Director, se muestra la EscenaHistoria en caso de haberla (la fase PirataArcade carece de ella). El flujo sería el siguiente:

- Menu -> EscenaHistoria: Pulsando sobre el Portal de la fase (Piratas/Dinosaurios).
- Menu -> EscenaCarga: Pulsando sobre el Portal de la Fase PirataArcade.
- EscenaHistoria -> EscenaCarga: Flujo normal en caso de haber EscenaHistoria (se puede omitir pulsando ESC).
- EscenaCarga -> Fase: Único flujo posible. Se realiza la transición una vez se cargan todos los elementos indicados.
- Fase -> Menu: Cuando cae derrotado en una de las fases o pulsando ESC.
- EscenaHistoria -> Menu: Después de la EscenaHistoria posterior al completar Piratas.
- Fase -> EscenaHistoria: Al completar una fase (excepto Fase Arcade).

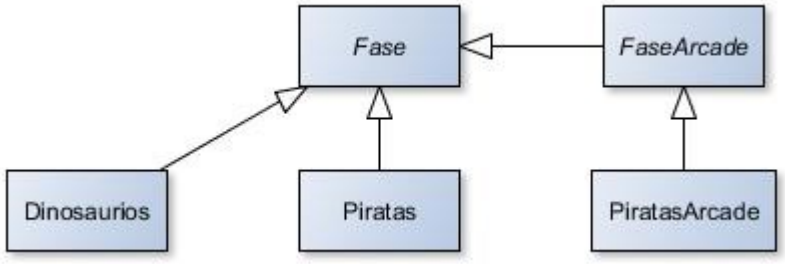
EscenaHistoria. Recalcar aquí que para mostrar una escena no jugable para contar la historia se han realizado tres implementaciones diferentes:

1. Con Pyglet. Aunque el acabado había sido bueno, se tenía el problema de que mostraba la escena en una ventana diferente, por lo que quedó descartada.
2. Creando una escena por cada imagen y apilando todas en el Director para su posterior muestreo. Esta implementación también se descartó debido a su baja funcionalidad.
3. Finalmente se ha implementado la EscenaHistoria, la cual, indicándole la fase que se va a reproducir después, muestra las imágenes correspondientes e inicia la carga de la fase, además de añadir las siguientes funcionalidades (carentes en las implementaciones anteriores): reproducción de audio y posibilidad de omitir toda la introducción a la fase, fundidos selectivos…

EscenaCarga. Dada la gran cantidad de elementos multimedia de cada fase, la carga inicial de todos esos assets llevada a cabo por el Gestor de Recursos tardaba mucho tiempo y el usuario no recibía ningún tipo de feedback durante ese periodo, más que una pantalla en negro. Para mejorar esta situación se ideó una solución que mejora la experiencia de usuario.

Cuando el usuario escoge qué fase quiere jugar, el director comienza una escena llamada escenaCarga que simplemente muestra un texto de espera al usuario. Se escribieron unos ficheros, con extensión "conf" en el directorio "others" quelistan los recursos que la escena jugable necesitará. escenaCarga, en cada ejecución de su método update llama al Gestor de Recursos con uno de los ítems listados.

Al acabar se apila la escena del juego para que el director comienze su ejecución. Se consigue así tener ya en memoria todos los recursos de esa escena. Cuando la escena los solicita al Gestor de Recursos sólo tiene que esperar a que se le devuelvan las referencias a los mismos con lo cual el tiempo de espera no es apreciable.



La clase abstracta ***Fase*** contiene toda la lógica de las diferentes fases, donde sus clases hijas sirven como parámetros para configurar la fase (fondo, sonidos, escenario, plataformas, jugador, enemigos…). ***FaseArcade*** es un caso especial que añade las funcionalidades propias del modo Arcade como son añadir enemigos y objetos aleatoriamente, así como redefine ciertos métodos de la clase ***Fase*** para adaptarla a este modo de juego. Así mismo, **PiratasArcade** sirve como configuración de la fase Arcade.

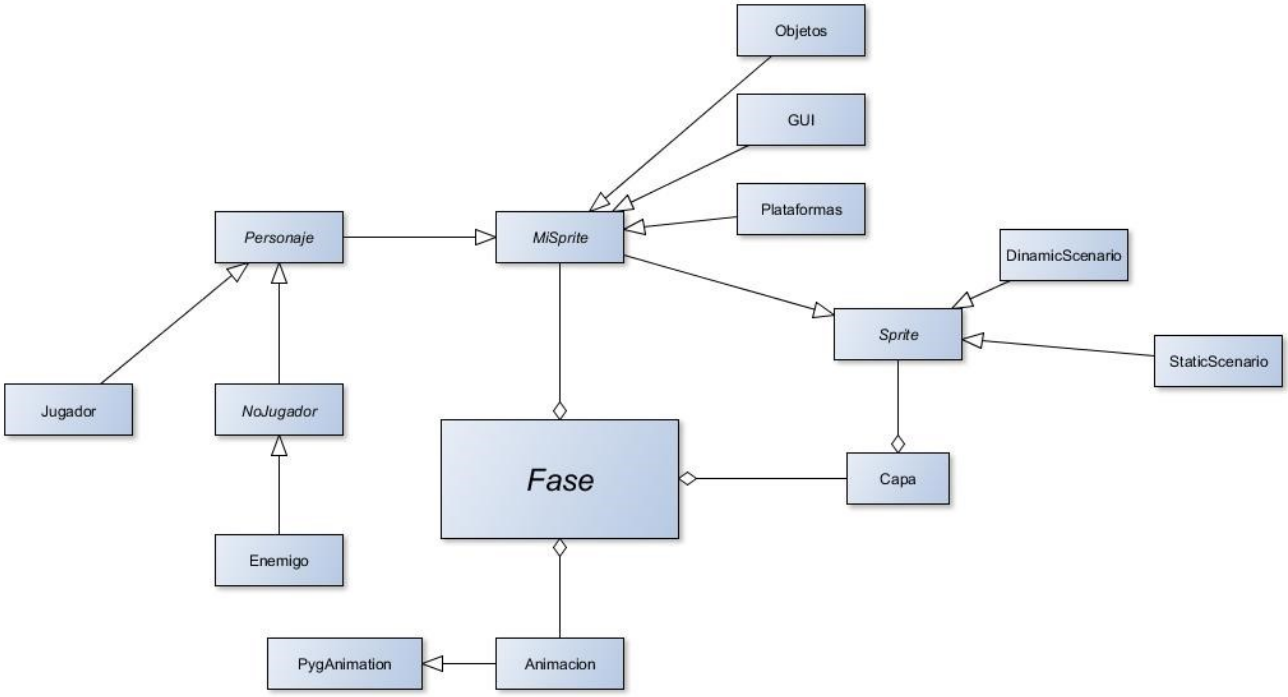
Al principio, se desarrolló todo sobre la clase **Piratas**, pero con el objetivo de modularizar y permitir la creación de fases de una forma más sencilla, abstrayendo el funcionamiento de esta fase como se comentó anteriormente. También se sopesó, en vez de utilizar clases hijas de fase, la creación de la fase desde un fichero de configuración, pero cierta variación entre la funcionalidad de las fases descartó esta idea (dinosaurios y piratas presentan diferencias como el oleaje, el decorado es diferente y tiene diferentes niveles …). Así que simplemente se hizo la carga desde fichero únicamente de las plataformas para no tener que incluirlo en el código.

Dos funcionalidades mencionables aquí son el fundido y el scroll en el eje y:

Para el primero se establece el atributo fade que indica cuando se está oscureciendo o aclarando la imagen, y este valor influirá notablemente en la lógica de la fase, pues el fundido a negro se inicia cuando se pierde (muere o pulsa ESC) o completa (matar al enemigo final) la fase. Final y fundido están ampliamente ligados, por eso se utiliza este atributo para determinar en varios puntos si la fase está terminando o aún en progreso.

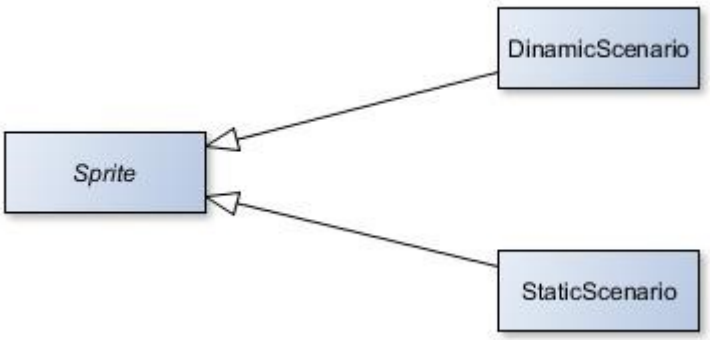
En cuanto al scroll en el eje Y, se ha creado el atributo virtual_scroll para crear el efecto del oleaje, y haciendo configurable este mediante las variables scrolly_speed y scrolly_amplitude. El uso de esta nueva variable se hace para desacoplar en cierta medida el movimiento del oleaje y el scroll en el eje vertical.

También en esta clase se encuentra el código que determina que hacer cuando un enemigo está en contacto con el jugador. Para determinar a quién quitar vida, se mira cuál de los dos está atacando, y quita la vida correspondiente al ataque del contrario. En caso de estar los dos atacando, no se quita vida y se reproduce un sonido de choque. En ambos casos se establece un periodo de vulnerabilidad al personaje que impide que le vuelvan a herir. Con esta implementación conseguimos un efecto que beneficia al jugador una vez encuentre la mecánica para herir a los enemigos sin salir herido, y radica en que el jugador debe entrar atacando, por lo que le quitará vida e este. Una vez empieza el enemigo a atacar, si el jugador aún está atacando se detectará un ataque mutuo y se establecerá el periodo de vulnerabilidad para el jugador, mientras el enemigo ya ha perdido la vida. De esta forma se consigue que el jugador mida bien los tiempos para atacar, ya que también tras atacar hay un retardo durante el que no podrá atacar (y así evitar que sea invulnerable, ya que atacaría todo el rato sin poder ser herido).

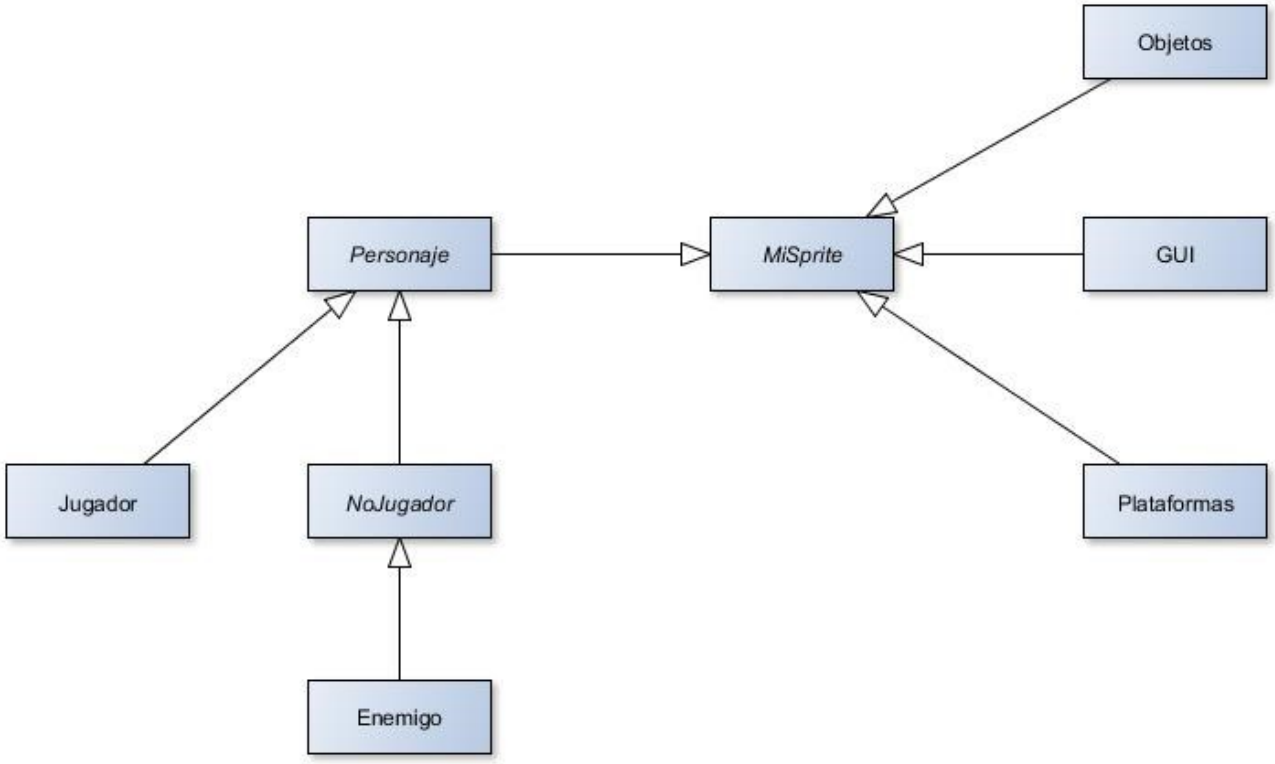


Los Sprites contenidos en la clase **Fase** se subdividen en diferentes grupos de Sprites con funcionalidad semejante (Jugadores, Enemigos, Objetos y Plataformas). Pero a la vez, los elementos del Escenario se agrupan en un objeto **Capa**. La clase **Capa** se diferencia de sprite.Group en que dibuja los Sprites que contiene en el orden en el que fueron añadidos. Se ha implementado de esta manera para que en conjunto con otros elementos, como los AutonomeSprite y diferentes decorados, den la sensación de profundidad. Es decir, se realiza así porque es necesario el dibujado ordenado.

Los citados AutonomeSprite se han implementado como una clase especial de Sprite con un movimiento y escalado automático solo dependiente del scroll. Se ha implementado independiente de MiSprite para posibilitar un scroll a diferente velocidad y con la idea de crearlo con los parámetros deseados para que funcione de forma autónoma.



Se han creado dos nuevas clases con el objetivo de dibujar la imagen de fondo. La principal razón de esta implementación ha sido la de poder cambiar de resolución y de poder establecer diferente velocidad de scroll (diferente velocidad de desplazamiento de la imagen respecto a la pantalla). En el caso de **DinamicScenario** además se añade la funcionalidad de que cambie la imagen con el tiempo (necesario para hacer el efecto del transcurso del día).



Junto con Fase, la clase abstracta ***Personaje*** alberga la lógica más relevante para el videojuego, donde se efectúa el cálculo de colisiones, se establece y calcula el movimiento, etc.

Uno de los detalles más importantes de la implementación es que se ha condensado el funcionamiento de las clases hijas dada su similitud. Esto se ve reflejado en el atributo **tipo** con el que variará ligeramente la configuración y acción de esta fase (sonidos a reproducir, velocidad de movimiento y salto, vida, imágenes a mostrar, etc).

Otros dos detalles relevantes son las colisiones y el movimiento:

- Movimiento: para poder permitir diferentes movimientos al mismo tiempo se utiliza el atributo **movimientos** el cual es un diccionario con el par que relacionan los diferentes movimientos posibles con la intención de realizarlo(por ejemplo, la entrada por teclado). El atributo **posturas** se verá influenciado por esta intención de movimiento donde entrará por el medio la lógica de la clase implementada en el método **update.** Finalmente, y según la postura elegida (de haber varios movimientos realizándose, se escogerá una postura por su prioridad) se imprimirá el Sprite asociado a dicha postura.
- Colisiones: para las colisiones entre los personajes y las plataformas se han creado tres tipos de estas:

1. Suelo: rectángulo por el que pueden caminar los personajes sin caerse.
2. Pared: rectángulo vertical el cual impide el movimiento horizontal de un personaje en caso de que quiera atravesarlo.
3. Rampas: similar al suelo pero va variando ligeramente la altura del personaje según en qué punto de la misma se encuentre.

Esto se ha implementado así debido a como queríamos mover el personaje por nuestro escenario, el cual se escogió con anterioridad a la implementación. Así mismo, la carga y creación de estas plataformas se hace desde un fichero con el objetivo de hacer las fases más modulares (utiliza el Gestor de Recursos).

En cuanto a los enemigos, que en un principio iban a implementarse en clases diferentes, tras comprobar que sus comportamientos eran idénticos (solo varían ciertas configuraciones), se han implementado los diversos enemigos en una misma clase, especificando el tipo de enemigo por el parámetro **clase** con el que se creará uno u otro enemigo.

En cuanto la IA, de haberla, los enemigos únicamente se acercan al protagonista y lo atacan una vez están a su alcance. Algunos de los enemigos también podrán evitar obstáculos en la medida de sus posibilidades.

Otra clase que hereda de **MiSprite** es **Objeto**, que dado su simple funcionamiento no hereda de **Personaje** y su función es almacenar el tipo de objeto y el sonido asociado a cuando se recoge, así como su movimiento, que se limita a caer hasta que encuentra una plataforma. Esta clase así como la inclusión de su uso en la clase **Fase** (colisión de objetos con los jugadores así como su efecto) se ha implementado debido a la inclusión del modo Arcade, siendo innecesarias en caso de no incluir esta Fase.

Finalmente, tenemos la clase **GUI**, que aunque inicialmente su misión era mostrar un Sprite de vida, debido al modo Arcade, se ha ampliado su funcionalidad alejándola un poco de la clase MiSprite. Esto se ha realizado así, y no se ha separado en dos clases, ya que su objetivo es el mismo.

Gestor de recursos. Es la clase encargada de almacenar todos los recursos que va necesitando el videojuego, y al mismo tiempo, también gestiona la configuración de este. Esta última funcionalidad nos permite guardar y cargar parámetros (usando json y su librería para Python) como pueden ser las puntuaciones y niveles alcanzados, como una pendiente personalización de controles, resolución, sonido… que no se ha llegado a realizar, ya que no se ha priorizado su implementación. De todas formas, estos parámetros pueden cambiarse desde el fichero json en “others/game.conf”

En cuanto a la resolución, el juego está pensado para poder ser reescalable, aunque finalmente no se ha comprobado su completo buen funcionamiento por lo que queda como una función experimental.

Finalmente, y aunque no se tenía contemplado desde un principio, se ha creado el Modo Arcade debido a la escasa duración del juego y a una progresiva acumulación de ideas que acabó ideando este modo. Un detalle a comentar es que es posible situarse en una plataforma elevada donde los enemigos no pueden llegar. Esto se ha dejado así para poder posibilitar que los objetos queden situados en lugares de difícil acceso.

2.5. Aspectos destacables

- Las fases nunca se repiten porque la posición y tipo de enemigos se crean de forma aleatoria.
- Cada vez que se supera un nivel, automáticamente sube la dificultad del mismo. De esta forma se puede jugar, en principio, de forma indefinida. Para conseguir esto lo que se hace es aumentar el número de enemigos en cada nivel, que, además cada vez serán más poderosos. También se descubren otros cambios según avanzamos de nivel, como invertir las teclas de control superado el nivel 4, poniendo a prueba la coordinación del jugador.

2.6. Manual de usuario

2.6.1. Contexto y objetivo del juego

Ambientado en diferentes épocas y con viajes temporales como narrativa principal, el objetivo del juego será sobrevivir a los dinosaurios y piratas de las dos fases disponibles, con el fin de encontrar al misterioso hombre de negro e impedir así que cambie la historia.

2.6.2. Ejecutar el juego

Para poder jugar se proporciona un archivo makefile en la carpeta "Game" que contiene los archivos y datos del juego. Situándose en esa carpeta, se puede escribir desde el terminal "make" y el juego comenzará a ejecutarse, enviándonos directamente al menú de inicio. Se recuerda que es necesario tener instaladas las librerías pygame y pyglet.

cd LaPlumaDelTiempo/Game/

make

También se puede ejecutar directamente el archivo principal del juego, main.py, con la instrucción:

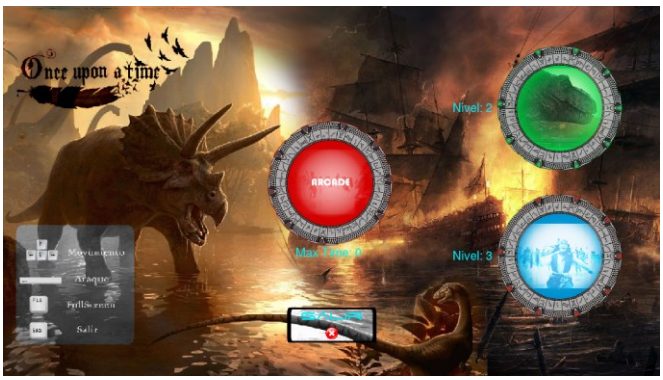
python main.py o, en el caso de Windows, doble click.

Adicionalmente, se incluyen en la carpeta raíz dos ficheros (start.bat y start.sh) para poder ejecutar el juego directamente.

2.6.3. Menú de inicio

Una vez iniciado el juego aparecerá el menú de inicio con las posibles acciones para el jugador.

Desde la pantalla de inicio el jugador puede iniciar una nueva partida, ver los controles del juego o salir de él. Según avance en el juego, se irán desbloqueando las diferentes fases y éstas serán también accesibles desde el menú, como se muestra en las siguientes imágenes:

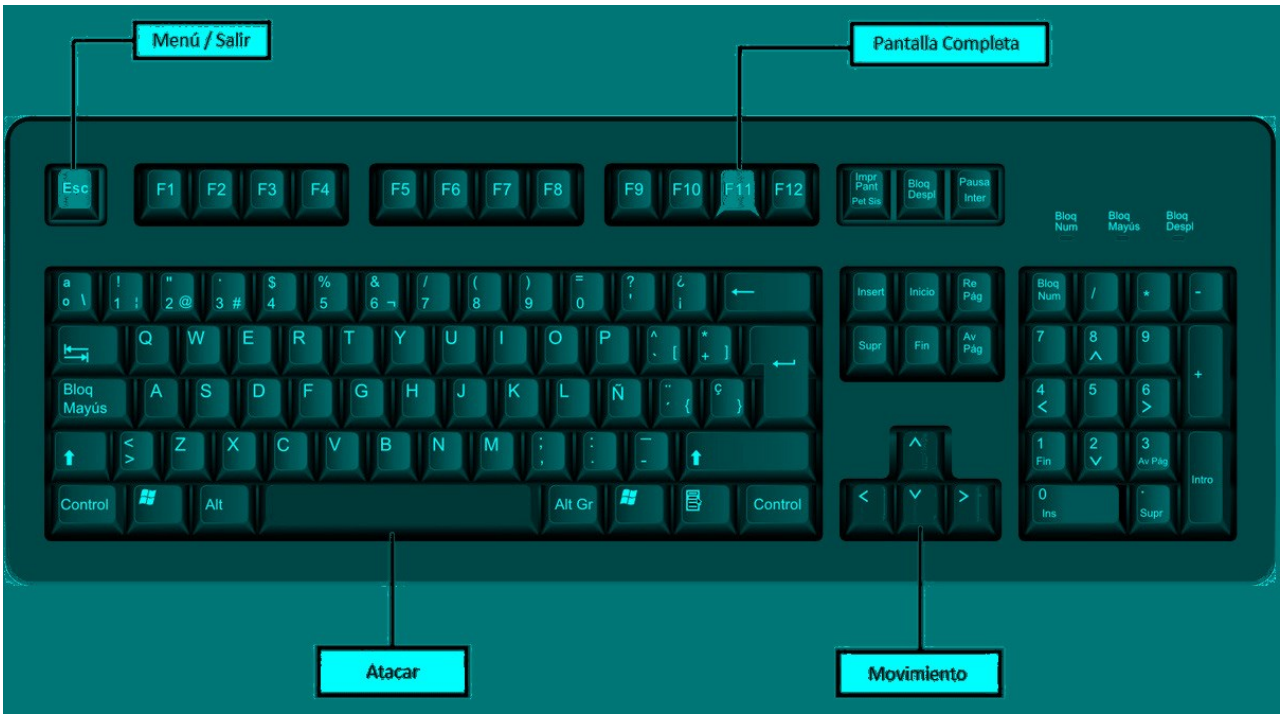


2.6.4. Controles

Durante la partida el jugador tiene las siguientes opciones:

- Movimiento: Mover el personaje por la fase con las flechas de dirección. Se puede mover a derecha e izquierda, así como saltar.
- Ataque: Atacar a los enemigos para poder derrotarlos usando la barra espaciadora.
- Volver al menú: Pulsando la tecla Esc.
- Ejecutar el juego en pantalla completa: Mediante la tecla F11.

Los controles se encuentran representados en la siguiente imagen:



2.6.4.1. Controles Joystick

Adicionalmente, se ha implementado de forma superficial el control mediante Joystick. Hemos usado el mando de la Xbox 360 con estos controles:

Al no poder comprobar dicha funcionalidad con otros Joysticks, se ha dejado como una funcionalidad latente.

Para activarse deberemos activar la constante booleana MANDO en la clase “Fase”. De este modo será posible el control tanto con el mando como con el teclado. Esta decisión se debe a que en un principio iba a estar siempre habilitada la opción de utilizar el mando, y que no se anularan entre sí (el teclado resetea el movimiento del personaje para establecer uno nuevo según qué teclas estén pulsadas, mientras el mando, que se comprueba posteriormente, establece movimientos de forma aditiva).

3. Desarrollo técnico videojuego 3D

3.1. Descripción del videojuego

Tras vencer al hombre misterioso durante los viajes a la prehistoria y al barco de Colón (videojuego 2D), el protagonista se encuentra esta vez en una base militar de la Alemania Nazi. Su objetivo es apropiarse de la máquina Enigma que poseen los enemigos.

3.1.1. Aspectos destacables

Con respecto al desarrollo artístico planteado inicialmente, el desarrollo del videojuego 3D se ha visto modificado en dos aspectos. Por una parte, se ha decidido implementar solamente uno de los niveles planteados inicialmente para poder ajustar los intervalos de tiempo del desarrollo, centrándose solo en una misión, y quedando el segundo de ellos pendiente para un posible trabajo futuro. En segundo lugar, el flujo del juego del nivel implementado se ha modificado ligeramente, con la idea de hacer el videojuego más entretenido y con más funcionalidades.

3.1.2. Personajes

3.1.2.1. Personaje jugable

- **Lorem Ipsum,** el protagonista. Ahora es un experimentado viajero temporal. Ha desarrollado sus dotes para pasar desapercibido en sus anteriores viajes temporales por distintas épocas, llegando a dominar el arte de la infiltración. En esta ocasión se infiltra travestido imitando la vestimenta de Amanda, la amante secreta de la general nazi.

3.1.2.2. Enemigos

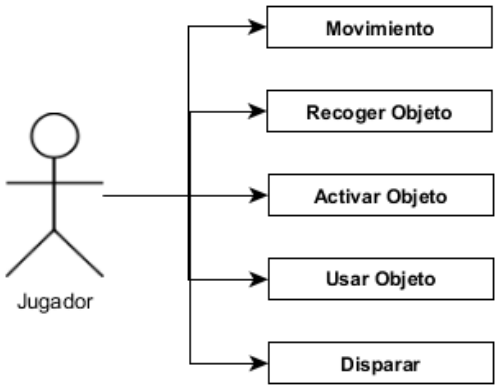
- **Soldados nazis**. Militares integrantes del Partido Nacionalsocialista Obrero Alemán que defienden la base militar. Si descubren al protagonista lo eliminarán.
- **La general**. La dirigente de la base militar. Carácter duro. Es la persona que porta la llave maestra que permite acceder a la habitación donde se encuentra la máquina Enigma y a la habitación de seguridad. Tiene una relación secreta con una mujer llamada Amanda.

3.2. Desarrollo del videojuego

3.2.1. Análisis

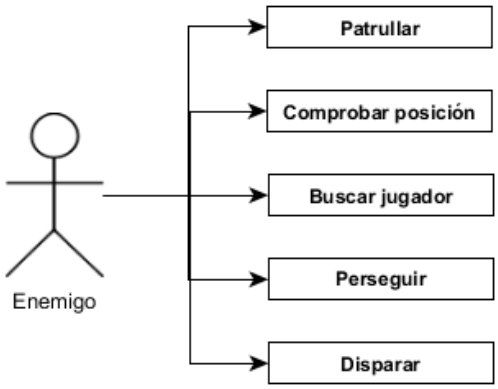
3.2.1.1. Personaje jugable

El jugador controlará al protagonista del videojuego, que puede desenvolver las acciones que se describen en el siguiente esquema:



3.2.1.2. Enemigos

Los enemigos son el principal obstáculo con el que se encontrará el jugador. Estos desenvolverán diferentes acciones en función del estado que les provoque el jugador.



3.2.1.3. Objetos

Existen distintos tipos de objetos en el videojuego:

a) Objetos de inventario:
   - Activables. El jugador puede activarlos para poder usar funcionalidades en el videojuego:
     - Rifle: podrá dispara a algunos objetos y enemigos.
     - Gafas de visión nocturna: podrá ver en la oscuridad.
   - Usables. Se pueden usar para alterar otros objetos:
     - Tenazas: sirven para cortar vallas.
     - Productos inflamables: el jugador puede rocíar vehículos para poder incendiarlos.
     - Mechero: el jugador puede incendiar vehículos que hayan sido rociados con productos inflamables.
   - Portables:
     - Llave maestra. Le permite al jugador abrir las puertas bloqueadas de la base militar.
     - Máquina Enigma. El objetivo del juego. El jugador necesita robarla y escapar con ella.

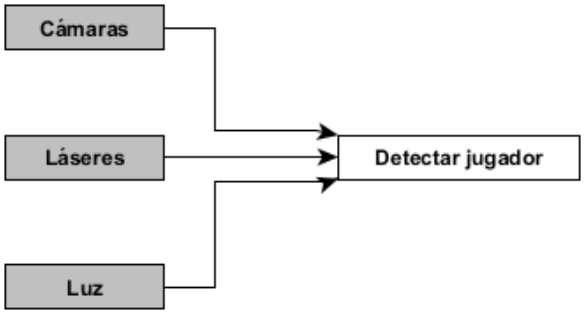
b) Objetos del entorno:
   - Puertas: Hay dos tipos:

- Bloqueadas. Solo se abren en caso de portar la llave maestra o ser un enemigo.
- Desbloqueadas. Se cada vez que un personaje se acerca a ellas.

- Generador eléctrico. Suministra energía a la iluminación exterior. Se puede apagar temporalmente y deshabilita dichas luces (y así reducir la visión de los enemigos).
- Panel de seguridad. Desactiva los láseres de seguridad temporalmente.

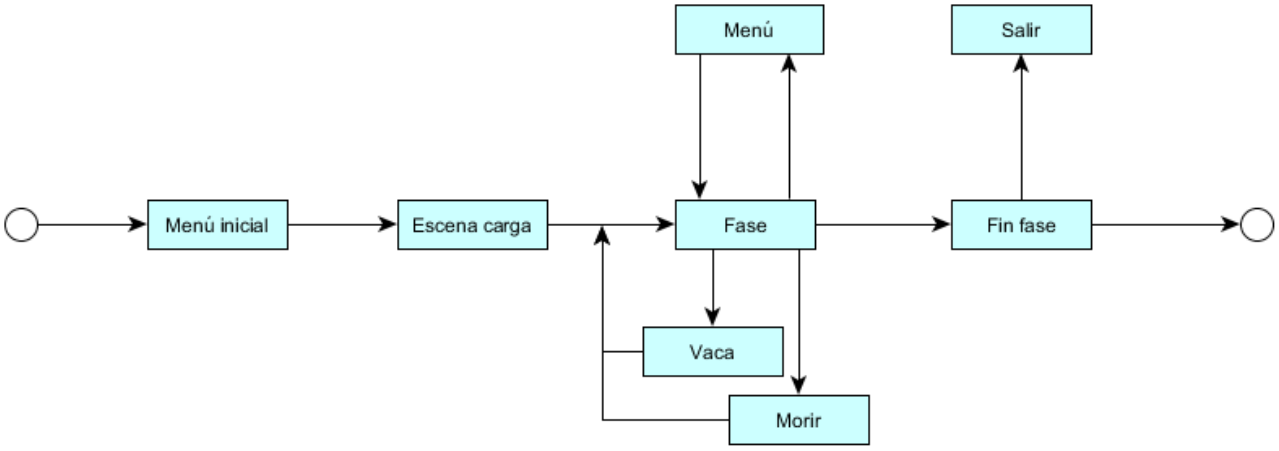
c) Objetos destruibles. Son objetos que el jugador puede destruir interactuando con ellos:
- Focos: disparando con el rifle.
- Cámaras: disparando con el rifle.
- Vallas: cortándolas con las tenazas.

d) Obstáculos: Son elementos de vigilancia que intentarán descubrir al jugador y dificultarán su misión (avisando a los guardias de la posición del jugador):



3.2.2. Diseño

3.2.2.1. Diagrama de flujo



3.2.2.2. Personaje jugable

El jugador puede controlar al protagonista tanto en primera como en tercera persona.

Si se encuentra con un objeto durante su exploración podrá (y deberá) recogerlo para poder usarlo con posterioridad.

Los objetos que tenga el jugador en posesión pueden ser activados o usados, dependiendo del objeto que sea.

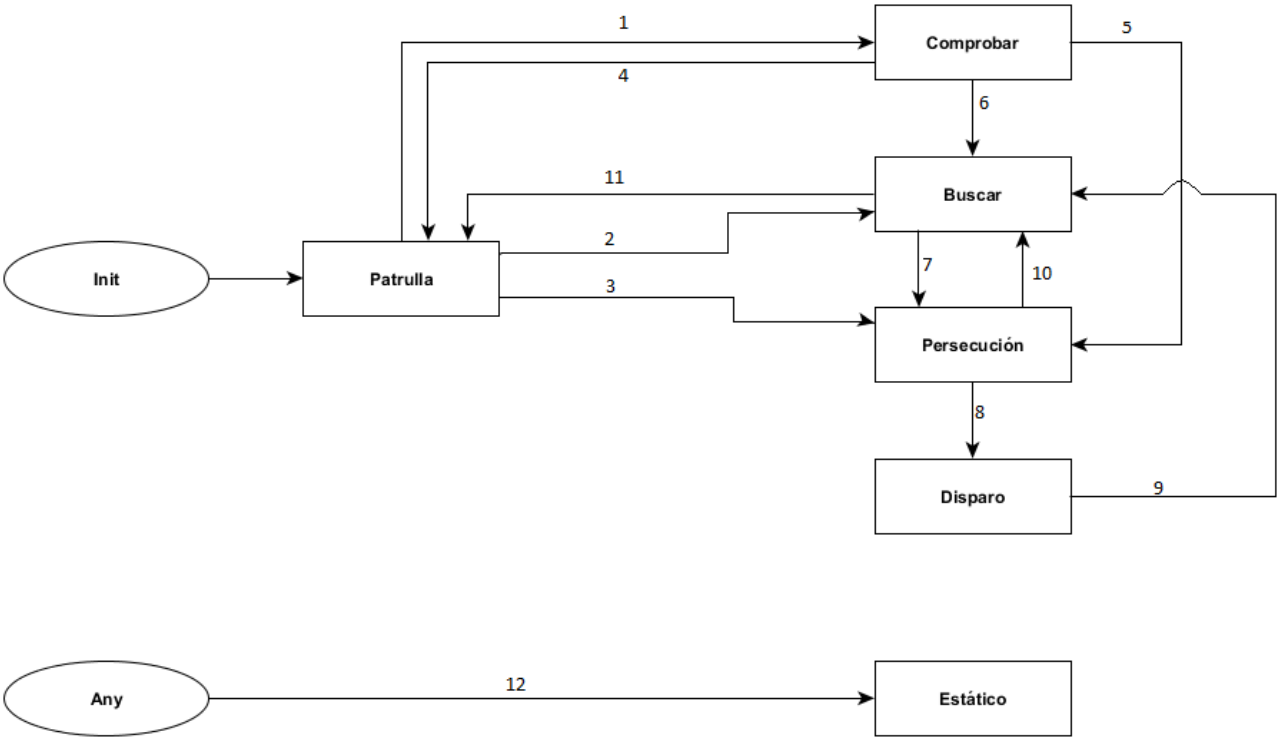
Si el jugador activa el arma podrá usarla para disparar.

3.2.2.3. Enemigos

Hay dos tipos de enemigos:

- La general.
- Los soldados.

Los enemigos están dotados de inteligencia artificial siguiendo el siguiente esquema de estados:



- Patrulla: Siguen una ruta de vigilancia.
- Comprobar posición: Acuden a un punto concreto.
- Buscar jugador: Siguen al jugador un determinado tiempo.
- Persecución: Persiguen al jugador mientras lo ven.
- Disparo: disparan con su arma al jugador.
- Estático:

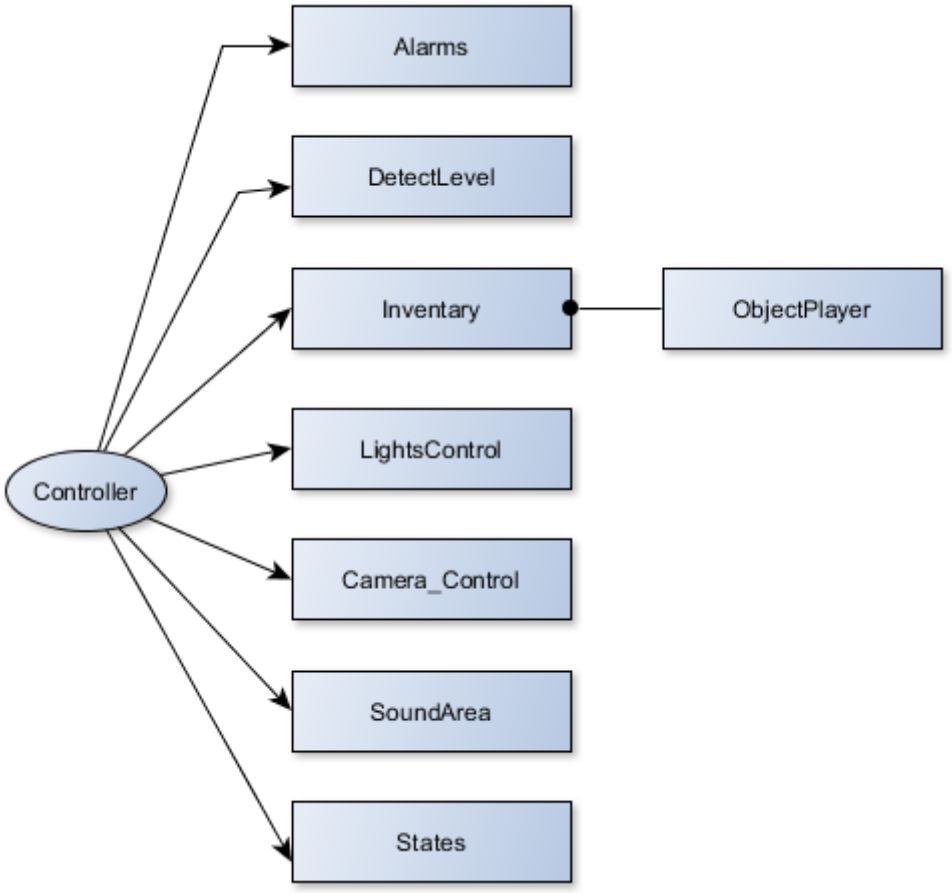
Transiciones:

1. Cuando se establece un punto de chequeo (activado por sonido del jugador).
2. Cuando se activa seguir al jugador (activado cuando saltan las alarmas) los enemigos cercanos buscan al jugador.
3. Cuando un enemigo ve al jugador, lo persigue.
4. Cuando el enemigo llega al punto de chequeo que lo llevo a este estado, si no encuentra nada, vuelve a patrullar.
5. Cuando el enemigo ve al jugador.
6. Si saltan las alarmas, el enemigo busca al jugador.

7. Cuando el enemigo ve al jugador.
8. Cuando el jugador entra en el área de disparo.
9. Si se pierde el contacto con el jugador y la animación de disparo ha terminado.
10. Cuando se deja de ver al jugador.
11. Si se alcanza el tiempo que se destinó a seguir el jugador
12. Si el jugador muere, todos los enemigos pasan a estado estático (independientemente del estado actual).

3.2.2.4. Diagramas de clases

[Controlador] Clases donde se encuentran la lógica, estado y control de la fase



-States: Basicamente se encarga de terminar la escena o reiniciarla, además de permitir el movimiento del personaje y también funciona como contenedor para referencias [Ver Implementacion]

-LightsControl: Se encarga de gestionar las luces exteriores así como de la versión nocturna.

-CameraControl: Gestina el cambio entre las cámaras entre primera y tercera persona, así como los efectos de estas [cambio entre ellas]

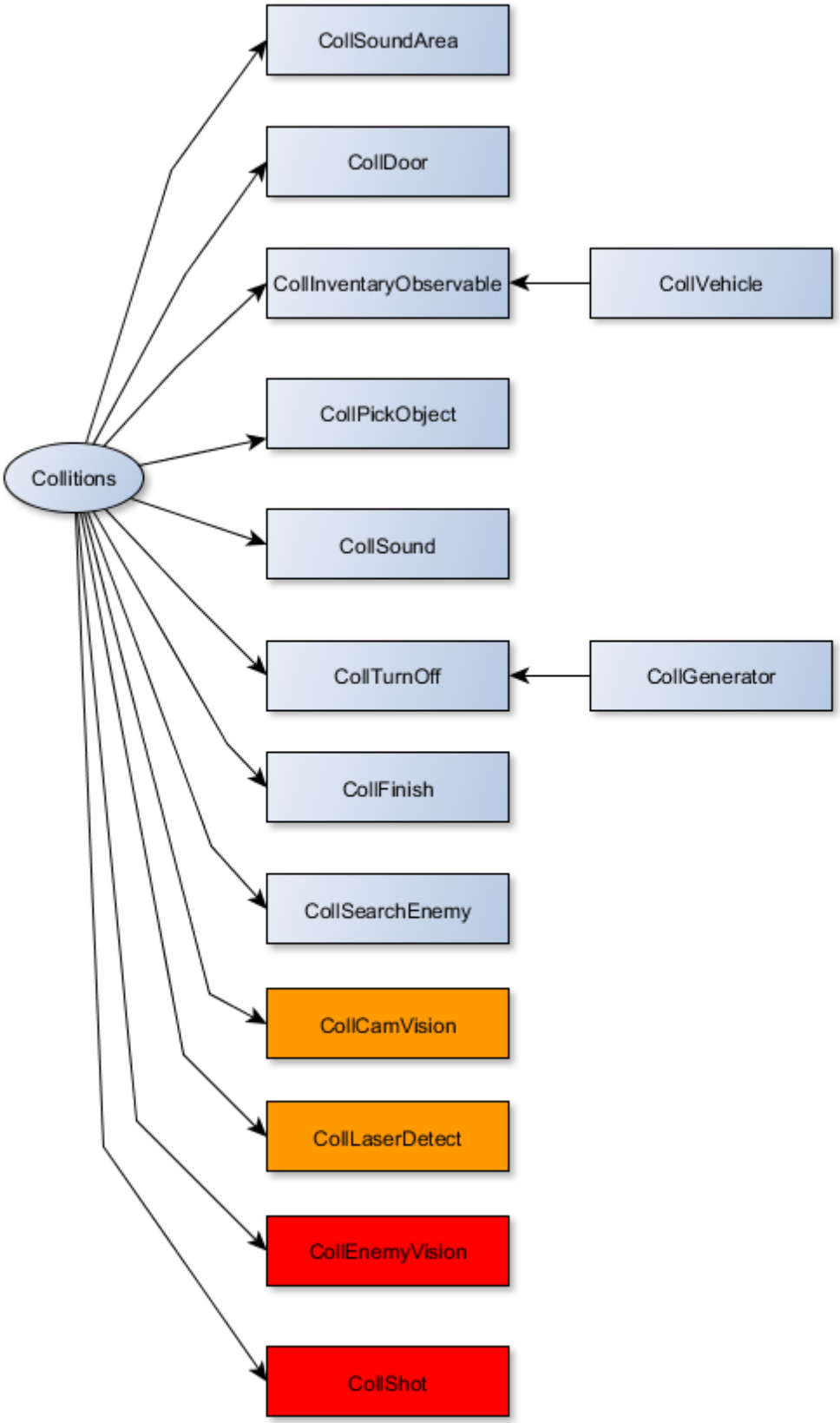
-DetectLevel: Gestiona el nivel de detección, así como las acciones que conlleva el cambio de dicho nivel [como por ejemplo avisar a los enemigos cuando se activa la alarma]

-SoundArea: Se encarga del área que produce el jugador al correr, andar, saltar o disparar y de su representación.

-Inventary: Contiene los objetos de inventario y se encarga de su comunicación con otros elementos.

-ObjectPlayer: Describe el tipo así como el comportamiento e interacción de los objetos de inventario.

[Collitions] Clases encargadas de gestionar las colisiones y lo que desencadenan



-CollSoundArea: Avisa a un enemigo está en el área donde se oye al jugador

-CoolDoor: Se encarga de gestionar la apertura de la puerta

-CollInventaryObservable: Indica al inventario con que objeto está en contacto con el jugador [por ejemplo que esta con una valla para poder activar las tenazas]

-CollPickObject: Encargado de detectar un objeto recogible y añadirlo al inventario.

-CollSound: Básicamente reproduce un sonido cuando el jugador entra en contacto con el collider

-CollTurnOff: Desactiva los objetos que se la han indicado cuando el jugador activa el objeto al que esta asociado el script [Por ejemplo los laseres]

-CollGenerator [Hereda de TurnOff]: Añade el funcionamiento de apagar el generador (apagar las luces exteriores)

-CollVehicle [Hereda de CollInventaryObservable]: Añade el funcionamiento para poder activar primero los productor inflamables y luego el mechero (en ese orden).

-CollFinish: Comprueba que el jugador esta en la zona para finalizar la fase y cumple las condiciones para completar la escena.

-CollSearchEnemy: Encargado de notificar a los enemigos contenidos en un collider para que vayan a una localización [Cuando se detecta al jugador en una cámara, DetectLevel crea un collider en tiempo de ejecución con este script]

[ColliderDetección]

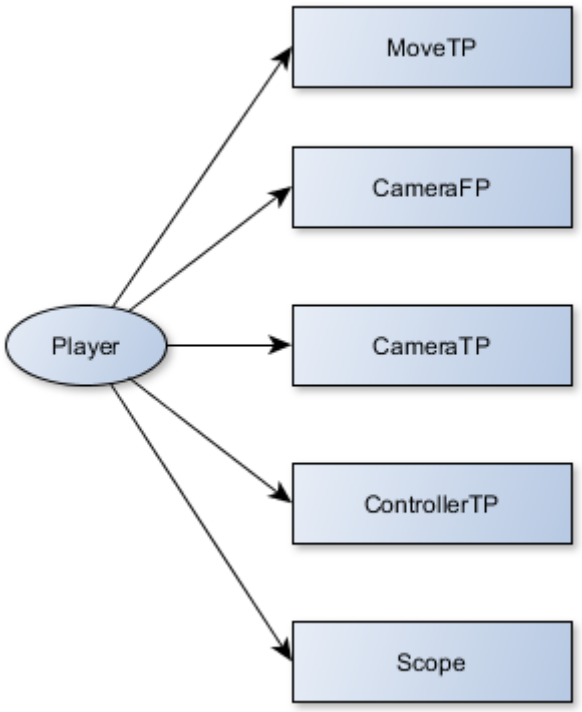
-CollCamVision: Detección del jugador con una cámara de seguridad [Collider + Raycast]

-CollLaserDetect: Deteccion del jugador con los laseres

-CollEnemyVision: Activa el modo Persecución del enemigo cuando este ve al jugador.

-CollShot: Activa el modo Disparo del enemigo cuando el jugador está lo suficientemente cerca de este

[Player] Clases encargadas de las cámaras en 1ra y 3ra persona, el movimiento del protagonista y la mira de la cámara



CameraTP Controla la cámara y control en primera persona

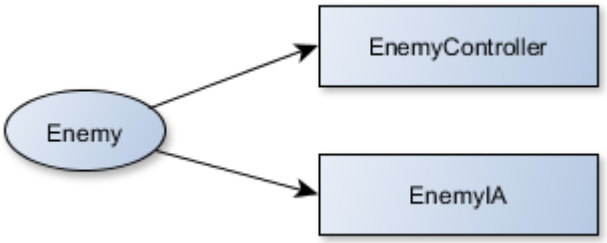
CameraTP Controla la cámara en tercera persona

MoveTP Gestiona el control del jugador en tercera persona

ControllerTP: Gestiona la animación del modelo del jugador en tercera persona

Scope: Funcionamiento del rifle

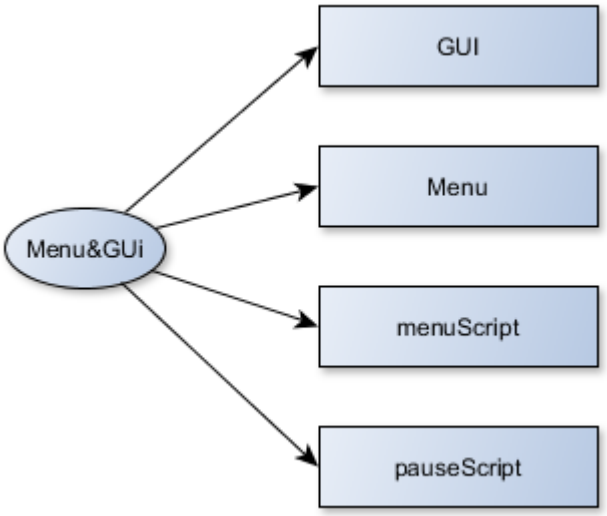
[Enemy] Clases encargadas del comportamiento de los enemigo



EnemyIa: Contiene la máquina de estados y la configuración del enemigo

EnemyController: Se encarga de gestionar la animación del enemigo

[Menu&Gui] Clases encargadas de los menus y de el HUD en pantalla



-Gui: Se encarga de gestionar la GUI en pantalla (mensajes, barras de proceso, actualizar la represetación del inventario, nivel de aletra, minimapa e etc…]

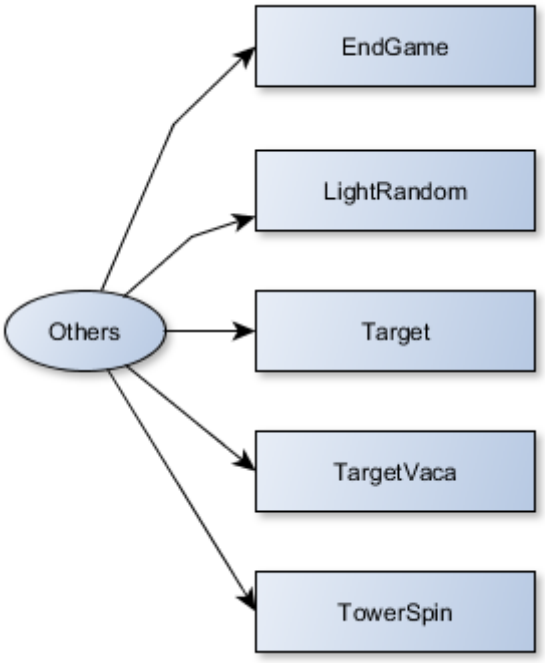
-Menu: Se encarga de gestionar la GUI cuando el jugador muere

-MenuScrip: Gestion Menu principal.

-PauseMenu: Contiene la gestión y funcionamiento del menú de pausa

[Animator] Simplemente actúan durante y para la animación

[Others]



-EndGame: Se encarga de controlar la gui al terminar la fase [Fundido y salir del juego]

-Target: Define un objeto al que se le puede disparar

-TargetVaca: Esto … ehm, mejor verlo que describirlo. Cambia la física de la vaca y hace el cambio de cámara.

-TowerSpin: Hace que gire la luz de la torre central

-LightRandom: Hace que una luz varia su tamaño e intensidad

3.3. Detalles de implementación

3.3.1. Personaje jugable

3.3.1.1. Cámara 1ª y 3ª persona

1ª y 3ª persona separadas, debido a las grandes diferencias y a que fueron implementadas de forma diferente y pensadas de forma diferente, lo que luego dio problemas a cosas como detectar al jugador, lo que terminó dando lugar al GameObject Player para contener cosas que ambos tipos de cámara comparten.

Debido a que inicialmente uno de los jugadores y cámara están deshabilitados (y aunque se deshabiliten por código alguno de los script no pueden escogerse), se da el problema de que no se pueden buscar por TAG, como se intentó hacer inicialmente, así que al final se introducen en el Camera_Control de forma manual mediante SerializeField. Se creó una clase aparte, dados los diferentes efectos de cámara y que se necesitaba deshabilitar la entrada en ciertos casos.

Además, se ha creado un objeto "Player" para hacer comprobaciones del jugador (como su posición) y poder hacerlas independiente de la cámara actual.

El juego de cámaras de cambio entre 1ª y 3ª persona es un poco complejo:

- Al cambiar de 3ª persona a 1ª se hace un zoom y una vez baje un umbral se cambia la cámara.
- En cambio, al cambiar de 1ª persona a 3ª, se cambia primero la cámara, y se aleja hasta el zoom que tenía la 3ª persona antes de cambiar (o a su valor por defecto la primera vez que se cambie).

3.3.1.2. Interacción con objetos

El jugador puede interactuar con diferentes objetos:

- Cortar vallas: se precisan unas tenazas y estar dentro del collider de la valla implementado para dicho fin.
- Incendiar vehículos: se precisa usar productos inflamables y mechero (en ese orden). Primero se pensó en crear el fuego en tiempo de ejecución, colocándolo en el objeto que se podría incendiar y centrando el prefab del fuego según el vehículo, aunque posteriormente se optó por una opción más sencilla y con un resultado mejor: modificar el prefab añadiéndole el fuego para adaptarlo a cada vehículo.

3.3.2. Enemigos

3.3.2.1. Navegación

Para que los enemigos pudiesen navegar por el mundo se ha creado un NavMesh. Esta estructura de datos describe las superficies caminables y permite encontrar el camino de una ubicación caminable a otra en el mundo del juego. Además, ha sido necesario marcar los objetos de la escena con la propiedad "Navigation Static", para que no se creara área de navegación a través de ellos.

Ha sido necesario probar diferentes opciones a la hora de crear el NavMesh (tamaño del Voxel, radios, alturas…), y ajustarlas minuciosamente hasta obtener la superficie navegable deseada, dado

que en muchas ocasiones se obtenían superficies navegables no deseadas y superficies no navegables que deberían de serlo.

El algoritmo que los enemigos usan para encontrar caminos es A*, el cual Unity es el que usa por defecto.

3.3.2.2. Detección

En cuanto a la visión, tanto de cámaras como de los enemigos, en una primera aproximación se realizaba un múltiple trazado de rayos en forma de cóno simulando la visión. Finalmente esta solución se descartó debido a su carga computacional y su complejidad, y se substituyó por un collider en conjunto de un simple rayo. Esto es, primero se comprueba que se está en el collider comprobando que entra en su área de visión, y a continuación se lanza un rayo para comprobar que no hay nada de por medio (que puede ver al jugador). Para esta implementación se han jugado con múltiples layers, en especial la IgnoreRaycast.

También en cuanto a la representación de dicha visión, se usó inicialmente un Mesh Render, aunque finalmente, y para poder señalizar que la visión de los enemigos no puede atravesar objetos, se hace mediante una luz con HardSadows con lo que la correspondencia entre la visión real y su representación es bastante fidedigna. Además, para no sobrecarga la escena con demasiada luz, se ha aplicado una textura de dicha forma.

3.3.2.3. Inteligencia artificial

Inicialmente simplemente se ha implementado un estado de patrulla y otro de persecución, pero posteriormente se implementó una máquina de estados y transiciones entre ellas bien definidas y claramente identificables en el juego.

Durante la implementación de dicha máquina se decidió implementarla como tal, con una primera etapa para decidir las transiciones y después ejecutar el estado, aunque podría ahorrarse código implementándolo de otra forma (cambiando los estados desde la acción de un estado, o simplemente independientemente del estado actual ir a uno determinado), finalmente, para mantener la idea inicial y por claridad y tener cada cosa en su sitio se implementó de esta forma.

Cómo funciona cada estado (código según en qué estado se encuentra el enemigo) se puso en funciones diferentes, aunque en su mayoría coinciden, se ha decidido dejarlo así para posibles futuras modificaciones en el comportamiento de las acciones.

3.3.3. Objetos

- Objetos de inventario: Aunque inicialmente se pensaron para implementar con herencia, pero para poder elegir la “Clase” de los objetos contenidos en el Inventario [array Inventary.cs] y editarlos desde el Inspector, se ha simulado el comportamiento de herencia por medio de un Switch y un enumerado que indica la clase.
- Vallas. Se han usado dos colliders en cada valla: uno para evitar pasar y otro con trigger para poder realizar la acción de corte, dado que se ha intentado con colisión, pero muchas veces no detecta.

- Rifle: Rifle añadido durante la implementación, ya que la idea gusto y se le vio una mejora en cuanto a jugabilidad, aunque pierde algo de su esencia como juego de infiltración, se penaliza bastante su uso, pero así el jugador no siente la impotencia de no poder defenderse del enemigo.

En cuanto a la activación de los objetos del entono, se ha usado un Collider diferente al del jugador (PlayerAction) para así solo poder activarlos o realizar una acción cuando se mira hacia ellos, y no poder activarlos de espaldas.

3.3.4. Cámaras

- Cámara rifle: Inicialmente se usaba la misma cámara cambiando el FOV para hacer el aumento, pero para una sensación de mayor inmersión se ha hecho uso de dos cámaras cuando se apunta con el rifle, la normal no se altera (en primera persona) mientras que en el área de la mira se coloca una imagen, la cual utiliza un render producido por una segunda cámara (en la misma posición y rotación que la original, pero con el FOV cambiando). Todo se realiza en el HUD.

  Se han encontrado grandes dificultades a la hora de contener el rectángulo, solucionándolo con el uso de máscara para que lo renderice solo en el circo.

  Se ha cambiado el funcionamiento de la cámara cuando se hace Scope. Tiene amortiguación del ratón como diferente sensibilidad, y cambio de FOV (solo mira del rifle) con el mousewheel.

3.3.5. Iluminación

En 3ª persona se deshabilitan, puesto que es necesario para el posicionamiento de la cámara (raycast y visión). Esto hace que la iluminación penetre en zonas interiores cuando se activa dicha cámara. Se probó tanto con layers (no renderizar tejados desde la cámara en 3ra persona), como deshabilitando el material y/o mesh render del tejado para que se pudiera ver a través de los tejados pero crear las HardShados con las luces, todo sin resultado. Finalmente, y por no resultar un gran problema, se dejó sin solucionar.

El control de la visión nocturna y la luces exteriores están en la misma clase, ya que inicialmente se altearnan solas y no había un objeto para activar la visión nocturna [y por ende no se activa directamente desde el propio objeto.

3.3.6. Ambientación

Se ha intentado agregar niebla pensando que conjugaría bien con la temática del videojuego, pero entre la carga que esta supone y que apenas se ha observado mejora en ambientación se ha tomado la decisión de descartar su inclusión.

También se descartó el uso de hierba, ya que su carga gráfica no compensaba en cuanto a mejor en ambientación suponía.

3.4. Aspectos destacables

Aunque inicialmente se tenía bastante clara la idea de cómo iba a ser el videojuego, y por ende, su diseño (se hizo un primer diseño de la implementación), dos factores hicieron que a la hora de implementarlo el resultado se alejara de esta idea inicial.

Primero, el empezar a utilizar Unity y no conocer bien los recursos de los que se puede hacer uso, dio resultado a tener que cambiar varias cosas del diseño inicial, así como tras haber conseguido cierta soltura con el programa, se rehizo trabajo donde se ha visto adecuado una mejor implementación de una forma diferente. Este caso fue, por ejemplo, el inventario y los objetos de este.

Segundo, y aunque inicialmente se pensó en diseñar y luego implementar, finalmente el ciclo de desarrollo fue derivando en un modelo más cercano al prototipado. Esto fue debido, además de lo anteriormente comentado, a que a medida que se implementaba lo ya diseñado se veían opciones o alternativas que mejorarían diferentes aspectos del juego. O simplemente, por mera curiosidad de comprobar el resultado de nuevas ideas, se implementaban dichas ideas y finalmente se añadían una vez visto el resultado aunque no fueran contempladas en un principio.

Esto, por ejemplo, se dio en el caso del rifle. Aunque inicialmente era un juego de infiltración, sin armas, se ha visto que el hecho de que nuestro personaje no se pudiera defender podría frustrar al jugador, así que se ha decidido añadir el rifle. Una vez añadido, se ha visto un recurso de jugabilidad muy valioso, así que se explotó este apartado (la mira, los disparos, el apuntado…). El hecho de añadir el arma repercutió en otros elementos, como fue los enemigos (ahora podrían morir, escuchar el rifle, etc). El cambio a tercera persona y un largo etcétera, con lo que hubo que cambiar varias partes del código, parámetros y demás elementos para que el juego conjugara bien con este nuevo elemento.

Otros ejemplos fueron el añadir cámaras, la visión de los enemigos…

3.5. Manual de usuario

3.5.1. Contexto y objetivo del juego

El jugador deberá controlar al protagonista para poder cumplir el objetivo final: apropiarse de la máquina Enigma. Para ello, tendrá que seguir correctamente unos pasos concretos.

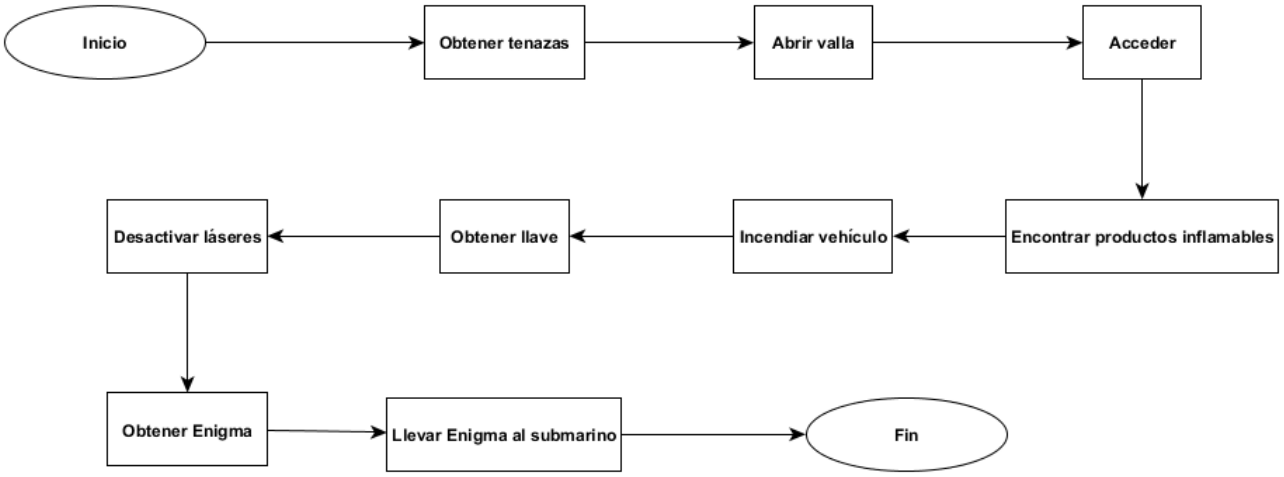
Al inicio del juego, el protagonista se encuentra a las afueras de la base militar y deberá ingeniárselas para acceder a ella. Es necesario encontrar unas tenazas para poder cortar una de las vallas y poder acceder. También es muy recomendable apagar el generador que suministra electricidad al exterior de la base, puesto que así se reducirá el campo de visión de los enemigos y se podrá acceder más discretamente.

Una vez dentro, hay que llamar la atención de la general para poder acceder a su despacho y robar la llave maestra. Para ello, el jugador deberá encontrar productos de limpieza inflamables en la base y provocar un incendio en uno de los vehículos que hay en el exterior. De este modo la general saldrá de su despacho y se podrá acceder a por la llave maestra.

Con la llave maestra en posesión, el siguiente objetivo es apropiarse de Enigma. En primer lugar, se debe acceder a la habitación de seguridad para desactivar los láseres de seguridad que rodean a la máquina. Se dispone de un tiempo limitado para poder acceder sin que suene la alarma.

Por último, se debe llevar la máquina enigma al submarino y escapar. Una vez hecho esto se completaría el objetivo del juego.

A continuación se muestra un diagrama de flujo donde se reflejan las transiciones que tiene que seguir el jugador para completar el nivel:



3.5.2. Controles

En las siguientes imágenes se representan los controles por defecto de teclado/ratón: