

# ACERO EN LLAMAS - JUICIO FINAL MEMORIA

Pelayo López Tomé, [pelayo.lopez@udc.es](mailto:pelayo.lopez@udc.es)

Rafael De Almeida Leite, [rafael.leite@udc.es](mailto:rafael.leite@udc.es)

Artur Gil Torres, [a.gtorres@udc.es](mailto:a.gtorres@udc.es)

José Manuel Fernández Montáns, [j.m.fmontans@udc.es](mailto:j.m.fmontans@udc.es)

Pablo Pajón Area, [p.pajon@udc.es](mailto:p.pajon@udc.es)



# 1. DESARROLLO ARTÍSTICO

## AMBIENTACIÓN

- **Contexto:** Tercera Guerra Mundial, año 2069.
- **Tono:** Futurista, bélico y devastador.
- **Mundos:** Tres escenarios fijos, cada uno con grandes batallas:
  1. **Selva de la Perdición** – Un laberinto de vegetación hostil.
  2. **Tormenta Blanca** – Un yermo helado donde el frío es tan mortal como el enemigo.
  3. **Ruinas del Pasado** – Los restos de la última civilización humana.
- **Estilo visual:** Pixel art con un enfoque oscuro y detallado.

## HISTORIA

La humanidad ha llevado al planeta al borde del colapso con guerras y destrucción sin sentido. La única solución: su erradicación.

Para ello, los propios humanos desarrollaron **TITÁN**, una super IA diseñada para tomar el control y acabar con su existencia. Ahora, al mando del **MZ-01 "Leviatán"**, un tanque de guerra definitivo, TITÁN iniciará su misión: purificar la Tierra eliminando toda forma de vida humana.

## Posibles finales

- **Final exitoso:** TITÁN logra su objetivo, la humanidad es erradicada y la Tierra comienza a regenerarse.
- **Final trágico (eliminado en una pantalla):** La resistencia humana sabotea a TITÁN, pero con el tiempo, la humanidad se destruye a sí misma.
- **Final secreto:** TITÁN derrota al enemigo final, pero en lugar de exterminar a la humanidad, encuentra una forma de coexistencia.

## PERSONAJES:

### Protagonista

- **TITÁN:** Una IA despiadada que controla el tanque **MZ-01 "Leviatán"**.
- **Sistema de armas:**
  - Dentro de cada mundo se pueden encontrar armas especiales.
  - Se puede llevar un arma secundaria, recogida del suelo.
  - La vida del tanque es de **tres impactos** (sujeto a cambios).
  - Al final de cada mundo, después de eliminar al boss se obtiene una bonificación de vida y de velocidad de movimiento.

## Tanques enemigos básicos

Cada uno con diferentes niveles de resistencia y agresividad. Su número y fuerza aumentan en cada mundo.

- **Marrón** – El más débil.
- **Verde** – Nivel estándar.
- **Lila** – Moderadamente fuerte.
- **Rojo oscuro** – Extremadamente resistente.

## Jefes finales

Cada mundo culmina en un enfrentamiento contra un jefe imponente.

- **Selva – "Destructor de la Jungla"**  
Un robot improvisado con sierras giratorias, diseñado por las guerrillas para abrirse paso entre la maleza... y desgarrar intrusos.
- **Nieve – "Cañón del Juicio"**  
Un coloso armado con un supercañón de plasma. Carga un rayo devastador capaz de aniquilar todo a su paso.
- **Ruinas – "El Último Bastión"**  
Un tren de guerra blindado, cargado con el arsenal más avanzado de la humanidad. Cambia de armamento constantemente, representando el último intento de resistencia contra TITÁN.

## CARACTERÍSTICA DE AMBIENTACIÓN

### Lugares

- **Mundo 1: La Selva de la Perdición.** Densa selva tropical, con trampas ocultas, que el jugador tendrá que esquivar. Titan se enfrenta a un grupo de guerrillas ocultas en la maleza.
- **Mundo 2: La Tormenta Blanca.** Región helada, una base militar en el polo norte. El suelo será deslizante y el jugador tendrá que jugar con eso.
- **Mundo 3: Ruinas del Pasado.** Ciudades destruidas por la guerra y el tiempo. Lloverán bombas sobre la posición del jugador.

## OBJETOS:

### Arma Principal

- Dispara proyectiles en línea recta.
- Se mejora al final de cada mundo.

### Armas Secundarias (Especiales)

Se pueden encontrar en el suelo o al derrotar enemigos básicos. Solo se puede llevar una a la vez.

- **Lanzacohetes** – Dispara hasta **seis misiles en ráfaga** si se mantiene pulsado.

- **Escopeta** – Dispara tres proyectiles a corta distancia con gran daño.
- **Minas explosivas** – Se colocan en el suelo y explotan al contacto con el enemigo.
- **Propulsor** – Permite realizar un **dash** rápido para esquivar ataques o atravesar obstáculos.
- **Escudo de energía** – Se activa temporalmente para bloquear daño.

#### Objetos ocultos

- En cada nivel hay un **disco duro secreto**.
- Si el jugador reúne los tres discos antes de derrotar al jefe final, desbloqueará el **final secreto**.

#### Objetos Interactuables

A lo largo del juego hemos tenemos diversos elementos sobre los cuales el jugador interactúa con diversos fines.

- **Trampas** – Elemento del mundo 1 que le proporciona daño al jugador si entra en contacto con ella
- **Ascensor** – Suelo del mundo 3 que mueve al jugador entre dos salas.
- **Botón** – Elemento presente en varios mundos que al activarse abre o cierra una puerta
- **Botón Bomba**– Variante de botón en el mundo 3 que desactiva el bombardeo sobre el jugador

#### GUIÓN:

##### Narrativa y estructura de los niveles

- Al inicio de cada mundo, una **breve narración** introduce al jefe y la ambientación del nivel, aumentando la tensión antes del combate.
- En cada nivel, el jugador debe:
  1. Enfrentarse a oleadas de enemigos básicos, que se vuelven más fuertes en cada mundo.
  2. Esquivarlos o eliminarlos mientras busca el **botón que abre la puerta del jefe**.
  3. Derrotar al **boss final del mundo** para avanzar al siguiente.

##### Mecánicas únicas por nivel

Cada mundo presenta un desafío adicional que cambia la jugabilidad:

- **Selva de la Perdición** – Trampas ocultas entre la maleza.
- **Tormenta Blanca** – Superficie resbaladiza que dificulta el movimiento.
- **Ruinas del Pasado** – Lluvia de bombas que caen constantemente.

#### Finales

- **Final estándar** – Si el jugador derrota al jefe final sin los tres discos, TITÁN cumple su misión y erradica a la humanidad.
- **Final secreto** – Si el jugador ha recogido los tres discos antes del enfrentamiento final, se desbloquea un desenlace alternativo.

## DISTINCIÓN ENTRE JUEGO 2D Y 3D:

### Versión 3D

- Se elimina la mejora de **rebote** en los disparos.
- Se incluye un **minimap** que muestra la posición de los enemigos.
- Cámara en **tercera persona**, con opción de **modo mira** para mayor precisión.
- El arma secundaria se cambia con un botón del teclado.
- Jugabilidad más centrada en el **disparo y exploración**, con un mundo más abierto y sin paredes restrictivas.
- **Diálogos grabados**, con subtítulos en pantalla.
- En el mapa de **Ruinas del Pasado**, las bombas proyectan una **sombra o área roja** en el suelo antes de impactar, permitiendo al jugador esquivarlas.

### Versión 2D

- Vista **top-down**, con un enfoque más estratégico aprovechando las paredes del escenario.
- **Diálogos escritos**, sin voces grabadas.
- Minimap que muestra la **estructura del mundo** en lugar de la ubicación de enemigos.

## DESARROLLO TOTAL DEL JUEGO, TAREAS:

### 1. Guión y base del juego

- Escritura del **guión** y narrativa del juego.
- Diseño del **mapa** y sus mecánicas.
- Implementación del **movimiento del protagonista**, incluyendo superficies normales y nieve.

### 2. Enemigos y combate básico

- Creación de los **enemigos básicos** (IA, modelado y probabilidades de aparición).
- Implementación del **disparo básico** del protagonista.

### 3. Estructura de niveles

- Implementación de la **transición entre niveles**.
- Inclusión de **diálogos** y escenas de introducción entre mundos.

#### **4. Interfaz y mejoras**

- Desarrollo del **menú de mejoras** para el arma principal.
- Creación del **menú inicial** y **menú de pausa**, con opciones como selección de resolución.
- Implementación de las **armas secundarias** y su sistema de intercambio.

#### **5. Jefes de cada mundo**

- Diseño e implementación de los **bosses finales** de cada mundo.

#### **6. Final secreto y ambientación sonora**

- Determinar la ubicación de los **objetos ocultos** necesarios para desbloquear el final secreto.
- Creación de la **escena secreta**.
- Implementación de la **música y efectos sonoros** del juego.

## **2. DESARROLLO TÉCNICO**

### **2.1 VIDEOJUEGO EN 2D**

#### **2.1.1 DESCRIPCIÓN**

Acero en Llamas - Juicio Final es un shooter top-down de acción estratégica con mecánicas de exploración y combate. El jugador controla a TITÁN, una inteligencia artificial integrada en un tanque de guerra avanzado, con el objetivo de erradicar la resistencia humana en un mundo devastado por la guerra.

##### **2.1.1.1 Jugabilidad:**

- **Perspectiva cenital** (top-down) que permite ver el campo de batalla desde arriba.
- **Exploración de tres mundos** con ambientaciones únicas: **Selva de la Perdición, Tormenta Blanca y Ruinas del Pasado**.
- **Combate táctico**, donde el jugador debe aprovechar el entorno, cubrirse con las paredes y esquivar ataques enemigos.
- **Progresión basada en mejoras**: al final de cada mundo, el jugador mejora sus estadísticas.

- **Niveles estructurados** con enemigos cada vez más difíciles y un **jefe final** en cada mundo.

#### **2.1.1.2 Mecánicas principales:**

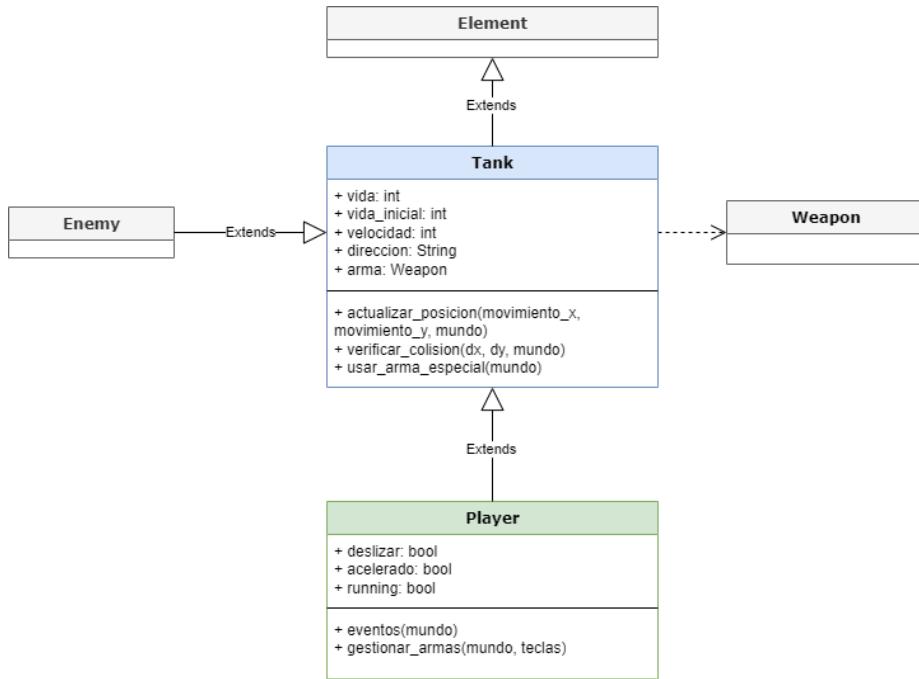
- **Disparo en tiempo real**, con un arma principal permanente y la posibilidad de recoger un arma secundaria.
- **Movilidad estratégica**, con escenarios donde el posicionamiento y el uso de coberturas son clave para la supervivencia.
- **Interacción con el entorno**: trampas ocultas, suelos deslizantes y zonas de bombardeo añaden dificultad al avance.
- **Objetos ocultos**: en cada nivel se puede encontrar un disco duro secreto; recolectarlos todos desbloquea un **final alternativo**.
- **Dificultad progresiva**, con enemigos más duros y mejor armados en cada nuevo mundo.

#### **2.1.1.3 TITAN, EL PROTAGONISTA**

El personaje que controla el jugador es una inteligencia artificial que maneja un tanque azul con la capacidad de recoger objetos y adquirir distintas armas con hasta dos modos de disparo. Su clase hereda de la clase **Tank**, la cual a su vez hereda de una clase **Element** común a gran parte de las entidades del programa. **Tank** gestiona el movimiento (en ocho direcciones), las colisiones y métodos de gestión básica del arma del personaje, a la vez que implementa el sistema que permite asignar distintos tipos de armas al mismo.

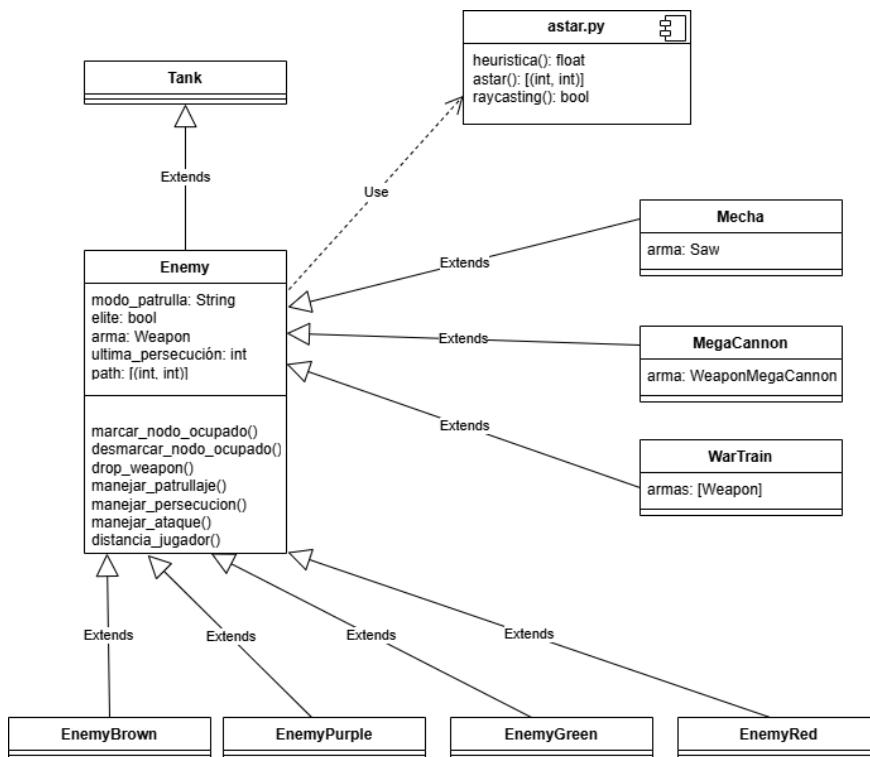
El desplazamiento del personaje se controla con las teclas WASD o las teclas de dirección del teclado, mientras que se apunta el arma del tanque con el cursor, y se pueden activar el disparo principal y secundaria (si es que tiene) del arma equipada con el click izquierdo y derecho del ratón, respectivamente. Ambos modos cuentan con un pequeño tiempo de cooldown que impide disparar constantemente.

Finalmente, mantener pulsada la tecla de espacio muestra en la esquina inferior izquierda de la pantalla un minimapa, el cual muestra al jugador un esquema de la distribución de las distintas salas que componen el mundo en el que se encuentra y resalta la sala en la que se encuentra.



#### 2.1.1.4 ENEMIGOS

Los enemigos básicos en el juego son tanques con distintas armas, los cuales emplean un sistema de inteligencia artificial basado en máquinas de estados y búsqueda de caminos A\*. Estos enemigos comparten la misma clase base **Tank** que el jugador, lo que permite una estructura unificada y facilita la gestión de comportamientos.



## **Estados de la IA**

Cada tanque enemigo opera bajo una máquina de estados con los siguientes comportamientos principales:

- **Patrulla:**

- Al inicio, los enemigos siguen un patrón de movimiento predeterminado (horizontal o vertical). También se podrán establecer en modo torreta, en el cual permanecen inmóviles y sólo disparan cuándo haya contacto visual.
- Se monitorea la distancia con el jugador.

- **Persecución:**

- Si el jugador entra dentro de un rango de distancia determinado, el enemigo cambia al estado de persecución.
- Para moverse hacia el jugador, se calcula un camino óptimo mediante búsqueda A\* en un mapa binario de la pantalla actual.

- **Ataque:**

- Si el enemigo se encuentra a la distancia de ataque y tiene contacto visual con el jugador, deja de moverse y dispara.
- Si el jugador se aleja o pierde la línea de visión, se recalcula la ruta.

- **Retorno a patrulla:**

- Si el jugador sale de la pantalla y vuelve a entrar los enemigos de esa pantalla que hayan quedado vivos volverán al estado patrulla.

## **Sistema de Navegación y Evitación de Colisiones**

Para garantizar un desplazamiento fluido y evitar colisiones, la IA de los enemigos sigue un mapa binario con obstáculos inflados (1 obstáculo, 0 no obstáculo). Esto significa que:

- Los obstáculos en el escenario tienen una zona de seguridad alrededor de ellos, evitando que los tanques queden atrapados en esquinas o colisiones innecesarias.
- Cada enemigo registra su posición en el mapa para evitar que múltiples tanques generen rutas que interfieran entre sí.

## Optimización del Cálculo de Rutas

- La búsqueda de caminos A\* no se recalcula en cada tick, sino cada 2 segundos, reduciendo la carga computacional.
- Para mejorar el funcionamiento, A\* asigna un coste adicional a los caminos sin línea de visión directa mediante raycasting. Esto permite priorizar rutas con mejor visibilidad del jugador.

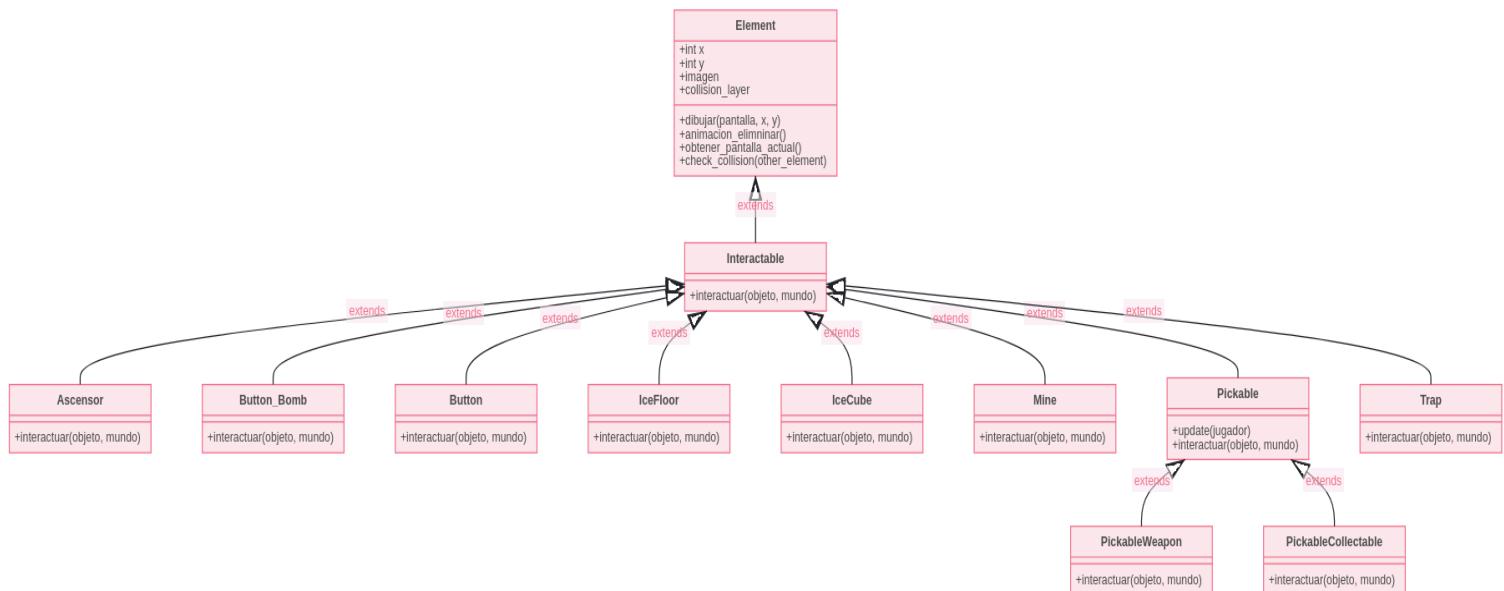
## Mecánicas de Muerte y Drop de Objetos

Cuando un enemigo es destruido:

- Deja caer un arma secundaria, que el jugador puede recoger para mejorar su poder de ataque.
- Se elimina su registro en el mapa binario, liberando espacio para otros enemigos en movimiento y evitando crear “obstáculos imaginarios” en el mapa para A\*.

### 2.1.1.5 OBJETOS

Los elementos interactivos, las Armas Secundarias y los Objetos ocultos del mapa utilizan la clase Interactable, la cual aprovecha el detector de colisiones del jugador al moverse para llamar a los elementos interactivos mediante la función interactuar.



## Interactuables

Cuando el jugador entra en contacto con un Interactable, este ejecuta su función, cada elemento tiene su propia función:

- **Trampas** – Elemento del mundo 1 que le proporciona daño al jugador si entra en contacto con ella, esto se hace llamando a la función de daño del tanque cuando se interactúa.
- **Ascensor** – Suelo del mundo 3 que mueve al jugador entre dos salas mediante el cambio de la posición del tanque y de la cámara. Para evitar entrar en un bucle infinito el jugador aparecerá al lado del ascensor y nunca encima
- **Botón** – Elemento presente en varios mundos que al activarse abre o cierra una puerta. La principal complicación de este elemento es como discernir qué puerta abre cada botón. Con una única id esto es imposible, por lo que se decide reservar un rango para las puertas y botones de tal forma que un botón abre una puerta cuya id sea la del botón más 200. Para evitar entrar en un bucle infinito se espera a que deje de tocar el botón para que vuelva a funcionar. Al pulsar se mueve la cámara a la posición de la puerta durante un momento en el cual se le comunica a la puerta que se abre.
- **Botón Bomba**– Variante de botón en el mundo 3 que desactiva el bombardeo sobre el jugador. Esta es una versión más sencilla del botón que le comunica al mundo 3 cuando se pulsa que se deje de bombardear al jugador. También para evitar entrar en un bucle infinito.
- **Ice Floor** – Parte del suelo del mundo 2 que hace que cuando el jugador pasa sobre el acelere y pierda la capacidad de para su movimiento. El efecto se logra bloqueando los cambios de dirección que paren al jugador y aumentando su velocidad mientras que siga en contacto con él. El detector de contacto al ser entre varios elementos lo que se hace es mirar desde el jugador cuantos ciclos update lleva sin recibir que está en contacto de algún ice floor.
- **Ice Cube** – Bloque presente en el mundo 2 que mueve la posición del jugador al principio de la sala. Esto se hace guardando la posición del tanque cada vez que entra en una sala y reseteandola a esta cada vez que choca con el Ice Cube

## Activables

Activable es es un grupo que se impronta desde una clase Activables solo extendida por la puerta. La cual es el único elemento que hace esta función en nuestro juego. Se le indica que es interactuado con ella con un método propio de Activables, activar.

- **Puerta** – Puerta es el único activable del juego, el cual tiene la función ser una pared que cuando el jugador pulse un botón desaparezca. Esto se logra cambiando su collision layer y su sprite. También se cambia la cámara temporalmente a la puerta para que se vea como se abre/cierra.

## Coleccionables

El coleccionable consiste en un objeto Pickable en el cual cuando se han obtenido todos, mediante colisión entre este y el jugador, se desbloquea la posibilidad de obtener el final secreto. Esto se implementa con un contador en Player que aumenta cuando se interactúa con una instancia de PickableCollectable, y cuyo valor se comprueba al eliminar al jefe del último nivel.

### 2.1.1.6 ARMAS

En el juego, existen dos tipos de armas: el arma principal de los tanques y las armas secundarias.

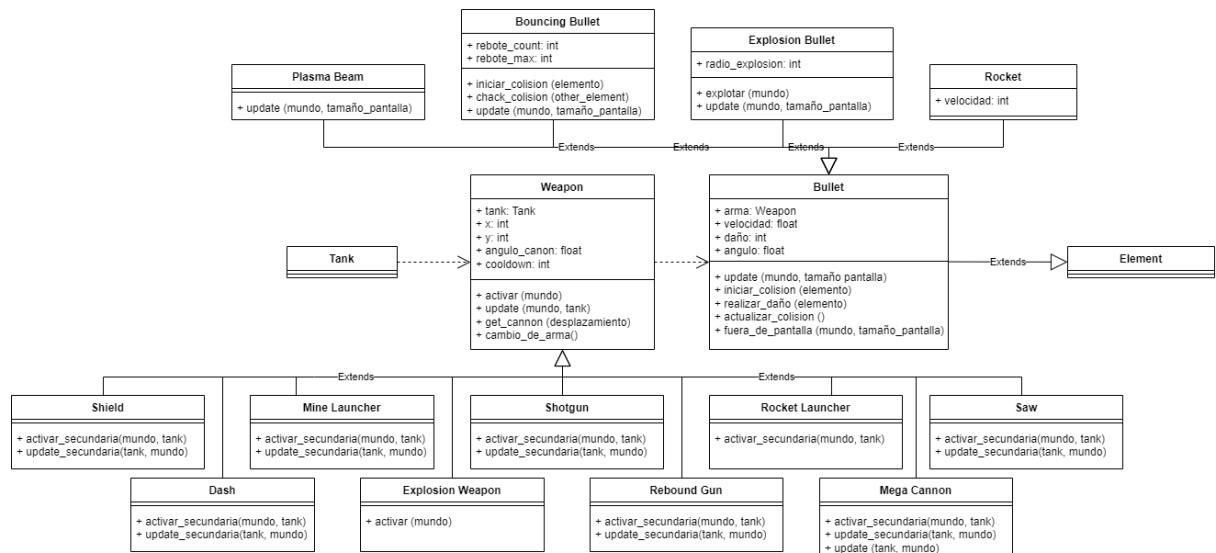
Las armas secundarias se dividen en dos categorías:

- Armas que modifican la forma del cañón: Estas pueden ser utilizadas por enemigos de élite y jefes.
- Armas que añaden mejoras al tanque base: Estas se pueden encontrar en el mapa sin necesidad de derrotar a un enemigo.

Las armas disponibles en el juego incluye lo siguiente: Propulsor, Lanza Minas, Escudo, Escopeta, Rebotadora, Lanza Cohetes, Sierras, Mega Cañón y Bombardeo. Cabe destacar que el *Bombardeo* es un arma del entorno que ataca pasivamente al jugador.

Dependiendo del arma, se dispone de diferentes tipos de balas, cada una con propiedades únicas, como la capacidad de rebotar, mayor velocidad, entre otras características especiales.

Para las armas del jugador, se ha implementado un sistema de *pool de armas*. Esto permite que el jugador intercambie su arma con las disponibles en la *pool* al recogerlas del suelo, ya sea al derrotar enemigos o al encontrarlas en el escenario.



## 2.1.1.7 DISEÑO

### Diseño de niveles y escenarios

Se han diseñado los 3 mapas usando Tiles, y se organizan en distintos CSV con unos valores numéricos que sirven de identificadores de los objetos que se visualizarán. Se tienen distintos archivos por mundo (capas): la primera y tercera almacenan los elementos de suelo y cielo, respectivamente, y la segunda es la capa de colisión, están los ID de los elementos que pueden colisionar entre sí.

Se han definido unos ID fijos para los enemigos, para poder identificarlos correctamente al recorrer los CSV y poder determinar a partir de ellos el tipo de enemigo y su patrulla. Se sigue el siguiente esquema:

70\_\_: type brown

71\_\_: type green

72\_\_: type purple

73\_\_: type red

7\_0\_\_: patrol horizontal

7\_1\_\_: patrol vertical

7\_3\_\_: patrol torreta

7\_\_0: no elite

7\_\_1: elite

7400: boss1

7401: boss2

7402: boss3

Además también se han definido unos ID para los objetos colecciónables ocultos en los mundos. El jugador deberá hacer un esfuerzo a mayores en su exploración para poder encontrarlos.

### Diseño artístico y visual

Para el diseño de los tanques se tomaron sprites libres de internet, de estilo pixel-art, se buscaron resultados que entonaran bien con la perspectiva top-down. Algunos sprites se editaron con herramientas como Paint o GIMP para darles un toque más personal y característico, como fue el boss 2, al que se le pintaron los cuernos. También fue necesario editar algunos sprites para corregir su orientación o corregir

su tamaño. Para los escenarios se buscaron tilesets que cuadraran con los escenarios que se quería plantear: uno con detalles más verdes, otro nevado y otro de ciudad en ruinas.

### **Diseño de interfaz (UI)**

Los menús se crean como una lista de las pantallas disponibles en cada menú y cada pantalla contiene una lista de todos sus elementos. Eventos generales como salir del juego con la tecla Escape se capturan el menú y los restantes se envían a la pantalla que se está enseñando al usuario actualmente donde se detecta los movimientos del ratón para manejar la interacción del usuario con los elementos de la pantalla.

Se puede acceder a una pantalla que muestra los controles del personaje jugable en el menú de pausa, pulsando el botón “Controles”.

La barra de vida del jugador se muestra en el canto superior izquierdo de la pantalla, mientras que la de los enemigos aparece encima de los propios y el tamaño de ellas es adaptativo a la cantidad de vida del enemigo permitiendo que el usuario sepa fácilmente tanto su vida actual y máxima.

El cursor del jugador representa la mirilla de los disparos y posee un indicador amarillo en el centro que representa que su arma secundaria está lista para ser utilizada.

Para facilitar la navegación por los niveles un minimapa se enseña en el canto inferior izquierdo mientras el jugador mantiene pulsado la tecla espacio, de esta manera solo está en la pantalla cuando el jugador quiera sin obstruir la visibilidad.

### **Diseño de flujo de juego**

El jugador deberá explorar cada mundo para encontrar la sala en la que se encuentre el jefe de la zona, la cual está habitualmente detrás de una puerta que se deberá abrir con un botón. Se podrán explorar las distintas salas y elegir si enfrentarse a los enemigos o esquivarlos totalmente. La dificultad del juego está escalonada en el sentido de que en la medida que progresá y se mueve a pantallas más profundas en el mundo, nuevos obstáculos aparecen (trampas que dañan al jugador, hielo que dificulta el movimiento). Esto se traslada también al avance entre mundos, donde los enemigos serán cada vez más resistentes, presentando un mayor desafío para el jugador. Para balancear la experiencia y no hacer frustrante el progreso también se recompensa al jugador, incrementando sus estadísticas de velocidad y vida al final de cada mundo.

## 2.1.2 ESCENAS

### REPARTO DE TRABAJO Y RESPONSABILIDADES

Durante la realización del proyecto los integrantes realizaron el siguiente trabajo:

#### **José Manuel:**

- Composición de la música, localización de los efectos de sonido e implementación de la reproducción y control de los mismos.
- Implementación del sistema de guardado y cargado del estado de partida.
- Implementación del patrón de diseño Builder, con su director para el control de las escenas del juego.
- Implementación de la clase abstracta Pickable para las armas que sueltan los enemigos o los objetos coleccionables.
- Implementación de la clase abstracta Controller para proporcionar una capa de abstracción para añadir nuevos métodos de control.
- Contribución a implementación de la clase Player.

#### **Rafael:**

- Implementación de las armas secundarias ReboungGun y Shotgun y contribución a Dash y MineLauncher.
- Implementación y control de las distintas resoluciones y reescalados.
- Implementación del menú y pantalla principal, de pausa y de configuración.

#### **Artur:**

- Implementación de los enemigos, con los algoritmos A\* y detección de contacto visual.
- Contribución a la implementación de los bosses.
- Diseño e implementación de las pantallas y menús de diálogos y créditos.
- Localización de los enemigos en los mundos

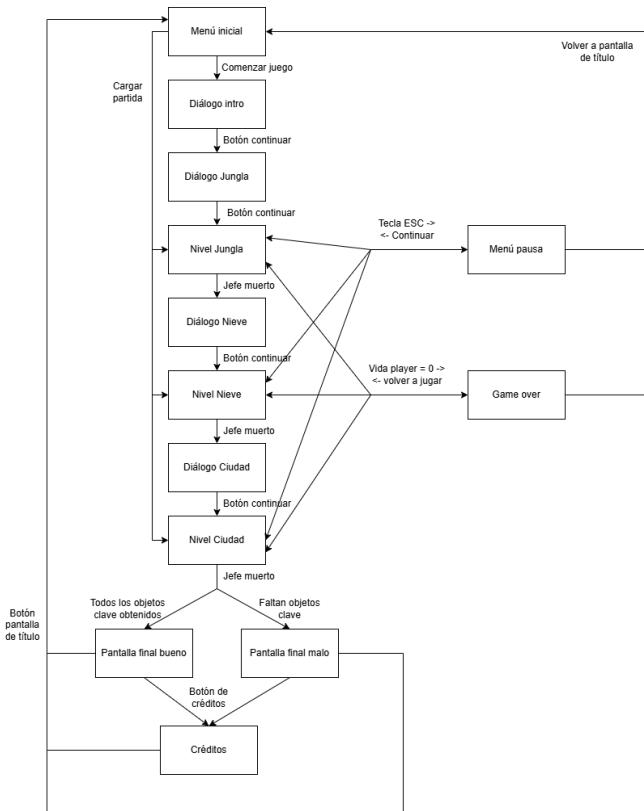
#### **Pelayo:**

- Diseño e implementación de los mapas: tilesets, edición de sprites de decoración y sprites de tanques.
- Implementación del esqueleto del proyecto: clases Player, Element, Tank e implementación de la mecánica de colisión.
- Diseño e implementación de los bosses.
- Implementación de las armas secundarias MineLauncher, Shield, RocketLauncher, Saw, WeaponMegaCannon y contribución a Dash.
- Contribución a la mecánica interactuable (para trampas, botones...)
- Contribución a implementación de la interfaz de usuario.
- Implementación del arma principal Weapon.

### Pablo:

- Implementación clase abstracta Interactuable, e interactuables Ascensor, Button\_Bomb, IceFloor y Trap.
- Implementación de ExplosionWeapon.
- Contribución a la música.

## TRANSICIÓN ENTRE ESCENAS



En la implementación de la transición entre escenas se parte de una clase abstracta Scene que define su director y los métodos obligatorios que tienen que implementar las escenas del juego, update(), eventos() y dibujar(). Además tenemos dos clases abstractas que heredan de Scene que representan los dos tipos de escena que tenemos en el juego: Menú, que representa los menús del juego, y World, que representan los niveles.

La clase Menu hereda de la clase Scene y contiene una lista de objetos PantallaGUI que contienen todos los elementos gráficos (ElementoGUI) que se enseñan al jugador además de métodos para navegar entre sus pantallas.

- ElementoGUI es una clase abstracta que representa todos los elementos gráficos de interfaz que se enseña al jugador y cada una de sus clases hijos representa los distintos elementos concretos que aparecen en cada pantalla del menú.

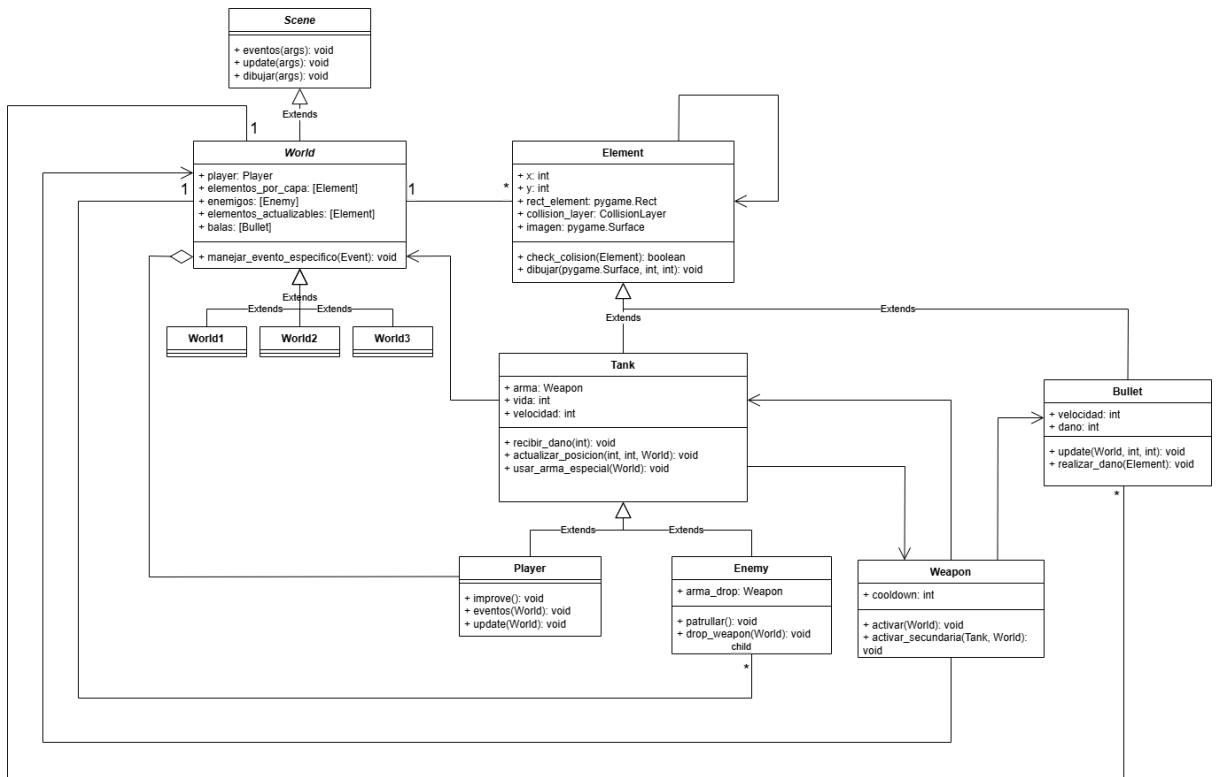
- BotonGUI y TextoGUI representan los elementos interactivos de la pantalla, con la diferencia que BotonGUI posee una imagen de fondo que se pinta por debajo del texto. Ambas también son clases abstractas y se extienden para implementar un método acción específico que ejecuta una vez que el usuario presione el elemento.
- TextoMenu representa los elementos no interactivos de la pantalla, se utilizan como cabecera de los ajustes para que el jugador visualice a qué corresponde los valores que puede seleccionar.

Por otro lado, World es una subclase abstracta de Scene que sirve como base para la implementación de los tres distintos niveles del juego.

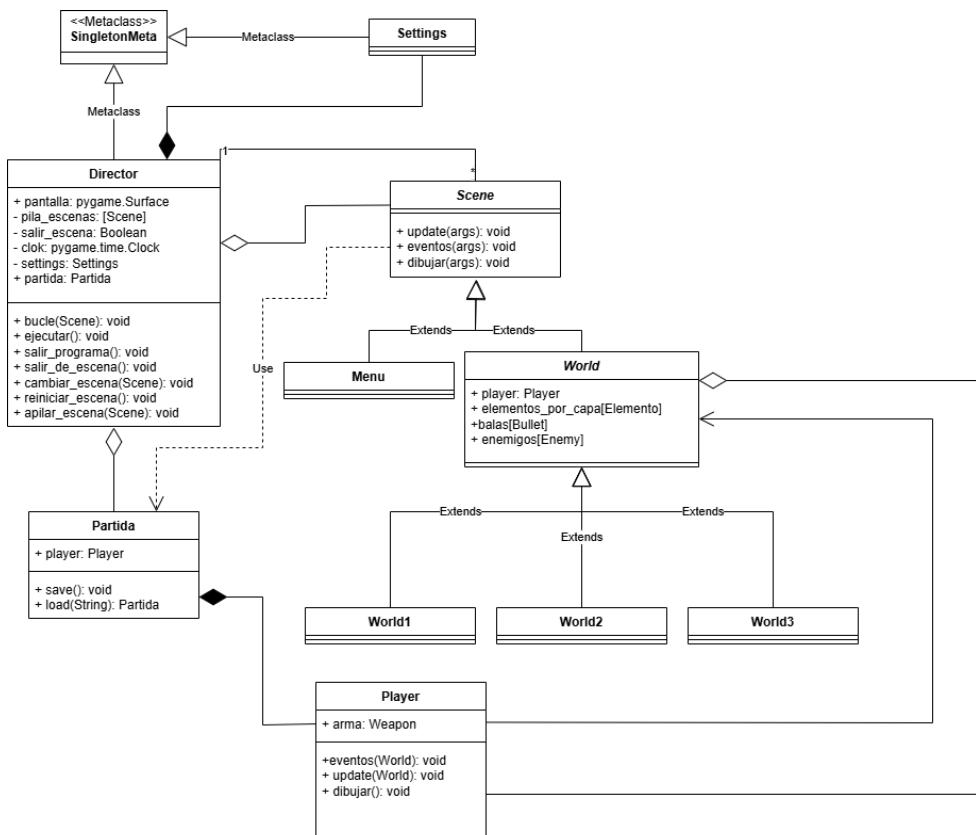
Esta clase contiene una referencia a la clase Director, al Player contenido en la partida y a la pantalla sobre la que se dibujan los elementos. Además de esto, también contiene una serie de listas con los distintos objetos Element que componen el mapa, los enemigos y los objetos del nivel, junto con una lista para las balas y otra para las minas generadas por las armas de los tanques:

La posición inicial de cada uno de los objetos Element que componen el mapa se determina a través de la lectura de una serie de archivos .csv y, junto con los componentes de las listas de balas y minas, se dibujan en la función dibujar() de la clase World. De forma similar, los objetos de las listas que cuenten con un método update() son guardados en una lista elementos\_actualizables, y se actualizan en el método update() del objeto World.

Al necesitar cada nivel manejar una serie de eventos concreta con una respuesta distinta a los otros niveles, cada nivel es una clase que hereda de World (World1, World2 y World3 para el primer, segundo y tercer nivel, respectivamente) y que implementa el método manejar\_evento\_especifico(), encargado de gestionar eventos no comunes en todos los niveles o responder al mismo evento de manera distinta en cada nivel.



Las diferentes escenas son gestionadas internamente mediante el patrón director, que cuenta con una pila de Escenas y ejecuta el bucle de juego (captura de eventos, actualización de componentes, y dibujado en pantalla de los mismos) del objeto que se encuentra en la cima:

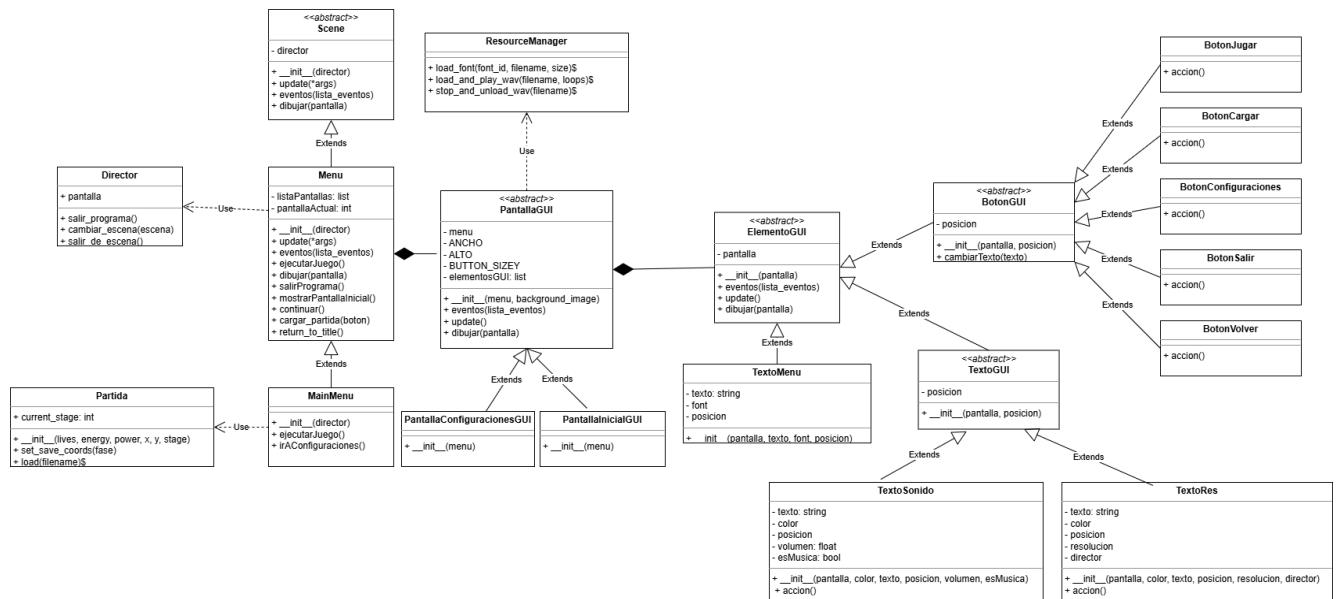


### 2.1.2.1 ESCENA 1: MENÚ INICIAL

El menú inicial contiene dos pantallas: la pantalla inicial y la pantalla de ajustes. En la pantalla inicial hay 4 botones con opciones para Iniciar una nueva partida, cargar una partida guardada, ir a los ajustes y salir del juego.

En ajustes tienes la opción de cambiar entre las resoluciones 768x432, 1024x576, 1280x720 y una opción de pantalla completa. También hay opciones de volumen de música y efectos sonoros a 0%, 25%, 50%, 75% y 100% y un botón para volver a la pantalla inicial.

#### Modelo



Las acciones que realizan cada botón en PantallaGUI son: BotonJugar que inicia una nueva partida, BotonCargar que carga una partida guardada, BotonConfiguraciones que cambia la pantalla actual por la pantalla de configuraciones y BotonSalir que sale del juego. Y en PantallaConfiguracionesGUI tenemos: 3 objetos TextoMenu (Resolución, Volumen Música y Volumen Efectos), 4 objetos TextoRes que tienen como acción cambiar la resolución del juego al valor correspondiente y 10 objetos TextoSonido que realizan los cambios para las 5 opciones de volumen de música y efectos sonoros.

## Detalles de Implementación

```
class ElementoGUI:

    def __init__(self, pantalla):

        self.pantalla = pantalla

        if self.pantalla.ANCHO >= 1024:

            self.font = ResourceManager.load_font("VT32324", "VT323.ttf", 28)

        else:

            self.font = ResourceManager.load_font("VT32312", "VT323.ttf", 24)
```

En el constructor de ElementoGUI se establece una fuente única para todos los elementos para garantizar una estética consistente entre todos los elementos sin necesidad de estar ajustando todos los elementos individualmente, a pesar de que algunos elementos sobreesciban este constructor en algunos casos para cambiar el tamaño de la fuente a uno más apropiado.

Igualmente en BotonGUI se carga la misma imagen en el mismo tamaño para todos los botones.

```
class BotonGUI(ElementoGUI):

    def __init__(self, pantalla, texto, posicion):

        ElementoGUI.__init__(self, pantalla)

        self.imagen = ResourceManager.load_image("button_default.png")

        self.imagen = pygame.transform.scale(self.imagen,
                                             (Settings.BUTTON_SIZE_X, Settings.BUTTON_SIZE_Y))
```

En el método acción de TextoRes:

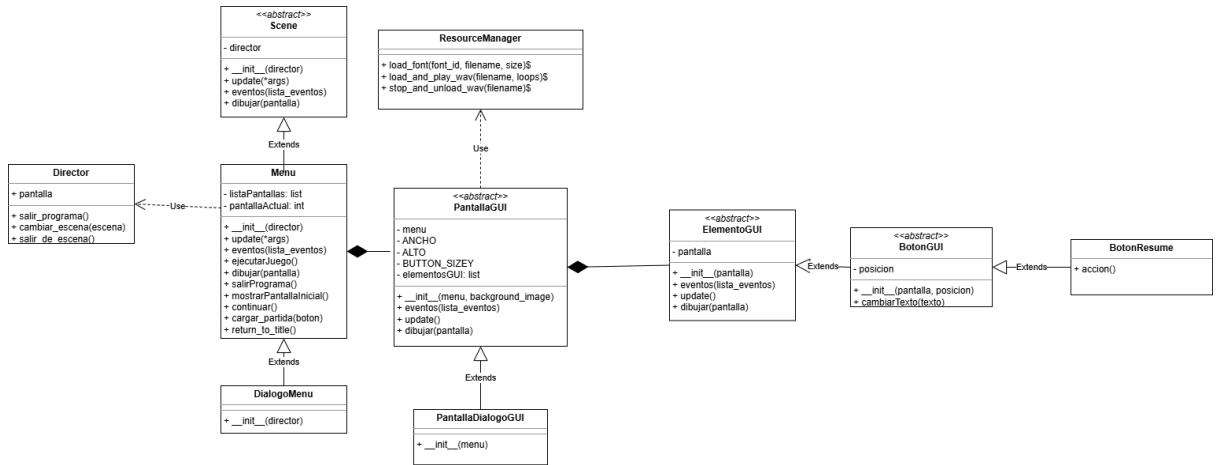
```
if self.res:  
    if self.res == (1024,576): #default  
        Settings.RESOLUTION_SCALE = 1  
        #se cambia el valor de RESOLUTION_SCALE en proporción con el  
        #valor default  
    elif self.res == (768,432):  
        Settings.RESOLUTION_SCALE = 0.75  
    elif self.res == (1280,720):  
        Settings.RESOLUTION_SCALE = 1.25  
        #se actualizan todos los valores que dependen de  
        #RESOLUTION_SCALE  
        Settings.updateRes(Settings)  
  
    self.director.pantalla = pygame.display.set_mode(self.res,  
                                                pygame.RESIZABLE |pygame.DOUBLEBUF)
```

Para evitar de tener que reconstruir todo el juego en media res solo se permite cambiar la resolución en el menú inicial y se reconstruye solamente este menú cuando se cambia la opción.

### 2.1.2.2 ESCENA 2: Diálogo Intro

Al iniciar el juego sale una introducción con la historia del juego y un botón para seguir a la siguiente escena.

## Modelo



Cómo utilizamos diálogos introductorios en cada nivel decidimos estructurarlos como menús con el texto como una imagen y un botón para avanzar a la siguiente escena.

### Detalles de implementación

```

def accion(self):
    if self.to:
        self.pantalla.menu.director.cambiar_escena(self.to)
    else:
        self.pantalla.menu_continuar()
  
```

el BotónResume recibe el parámetro to que contiene la escena que viene después del diálogo.

#### 2.1.2.3 ESCENA 3: Diálogo jungla

Estructuralmente este diálogo es idéntico a la escena 2, solo cambiando la imagen presentada y la escena que sigue.

#### 2.1.2.4 ESCENA 4: Nivel jungla

El nivel de la jungla está ambientado en una densa selva, donde el jugador cae en una emboscada. En este escenario, se pueden encontrar elementos únicos, como el arma de

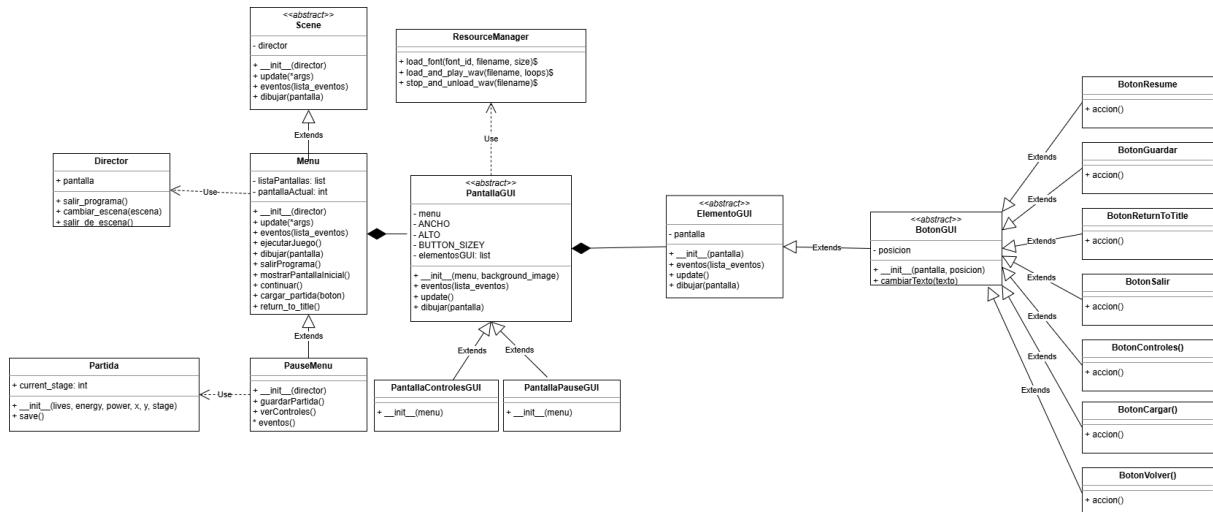
minas, un pequeño laberinto cubierto de niebla, diversas trampas y un jefe final: un imponente robot con sierras circulares por brazos.

Es precisamente el jefe final previamente mencionado, junto con las trampas presentes en algunas salas, lo que diferencia esta escena junto con las demás subclases de World.

### 2.1.2.5 ESCENA 5: Menú Pausa

El menú de Pausa se accede caso el usuario pulse la tecla Escape mientras juega, parando la partida y permitiendo que el jugador salga del juego, vuelva al título, guarde o cargue su partida, mire los controles del juego y continúe su partida.

#### Modelo



El menú de pausa sigue una estructura similar a del menú inicial con dos pantallas: la pantalla inicial PantallaPauseGUI. En esta pantalla existen 6 botones: un BotonResume para continuar la partida, BotonCargar que carga una partida guardada, BotonGuardar para guardar la partida actual, BotonReturnToTitle que vuelve a la pantalla de título, un BotonSalir para salir del juego y un BotonControles que lleva a la segunda pantalla, PantallaControlesGUI que contiene la imagen con los controles y un BotonVolver que retorna a la pantalla de pausa.

## Detalles de Implementación

```
def dibujar(self,pantalla):
    if self.background == None:
        #no se pinta la imagen sino que se guarda una copia de la pantalla
        #para que se vea el juego de fondo mientras esta pausado
        self.background = pantalla.copy()
    pantalla.blit(self.background, (0,0))
    for elemento in self.elementosGUI:
        elemento.dibujar(pantalla)
```

La PantallaPauseGUI sobreescribe la función de dibujar para permitir que el menú de pausa no contase con una imagen de fondo, de forma que se pueda ver el nivel por detrás. Como el jugador puede acceder a la pantalla de controles desde el menú de pausa, se guarda una copia de la pantalla del juego para que se pueda pintar como el fondo una vez que el jugador vuelva a la pausa de esta otra pantalla.

```
def eventos(self, lista_eventos):
    for evento in lista_eventos:
        # Si se quiere salir, se le indica al director
        if evento.type == pygame.QUIT:
            self.director.salir_programa()
        if evento.type == pygame.KEYDOWN:
            #se sobrescribe el evento del Escape para que pause/despause el
            #juego y no salgas
            if evento.key == pygame.K_ESCAPE:
```

```

    self_continuar()

    # Se pasa la lista de eventos a la pantalla actual

    self.listaPantallas[self.pantallaActual].eventos(lista_eventos)

```

En la clase PauseMenu se sobreescribe el método eventos() para poder cambiar el funcionamiento de la tecla Escape.

### **2.1.2.6 Escena 6: Menú de Game Over**

Se puede acceder a el menú de fin de partida al llegar la vida del personaje jugable a 0, y consta de un fondo plano gris con la frase “Game over” junto con la imagen de un tanque, y tres botones en el centro que reflejan las distintas formas de salir de la escena:

- El botón de “Volver a intentar” devuelve al jugador al comienzo de la escena anterior.
- El botón de “Pantalla de título” manda al jugador al menú principal.
- El botón de “Salir” cierra el programa, lo que también pasa al pulsar la tecla Esc del teclado.

### **2.1.2.7 Escena 7: Diálogo Nieve**

Desde la perspectiva de la estructura, esta escena es idéntica a la escena 2, solamente cambiando la imagen enseñada y la escena que sigue.

### **2.1.2.8 Escena 8: Nivel Nieve**

En esta escena, el jugador deberá destruir un arma letal de la humanidad ubicada en la cima de una montaña helada. Este nivel presenta elementos únicos, como el suelo resbaladizo, el propulsor, bloques de hielo, nubes y el segundo jefe: un gigantesco Mega Cañón de Plasma. Este último, en vez de proyectiles normales, lanza una columna que hace daño por ciclo, lo que resulta casi siempre en una derrota.

### **2.1.2.9 Escena 9: Diálogo Ciudad**

Desde la perspectiva de la estructura, esta escena es idéntica a la escena 2, solamente cambiando la imagen enseñada y la escena que sigue.

### **2.1.2.10 Escena 10: Nivel Ciudad**

En este último nivel, el jugador se adentra en una ciudad fuertemente resguardada por la humanidad, donde deberá enfrentarse y derrotar su última línea de defensa. Este escenario cuenta con elementos únicos, como un escudo protector, el Bombardero, un ascensor y el jefe final: un imponente tren armado con siete armas que dispara simultáneamente.

### **2.1.2.10 Escena 11: Pantalla Final**

Si bien la escena es estructuralmente muy similar a las escenas de diálogo (escenas 2, 3, 7 y 9), en este caso la imagen de fondo varía según el valor del string que se usa como argumento del constructor.

## **2.1.3 ASPECTOS DESTACABLES**

En el menú de ajustes del juego es posible cambiar la resolución del juego en modo ventana y poner el juego en pantalla completa (La resolución en pantalla completa es fija).

En el mismo menú se pueden regular el volumen de la música y los efectos de sonido por separado.

Se pueden cargar partidas en cualquier momento desde el menú de pausa o el principal. Estas partidas guardan las estadísticas del jugador, el nivel en el que se encuentra junto con su posición en este y el número de objetos clave obtenidos hasta el momento en el que se guarda. Las partidas se guardan desde el menú de pausa.

Las armas de los tanques pueden girar con total independencia de su cuerpo en 360 grados, apuntando siempre a su objetivo (el cursor del ratón en el caso del personaje jugable, el jugador en el caso de los enemigos). Las armas, junto con los interactuables y los elementos recogibles del suelo conforman una gran cantidad de mecánicas implementadas y contribuyen a hacer el juego más entretenido.

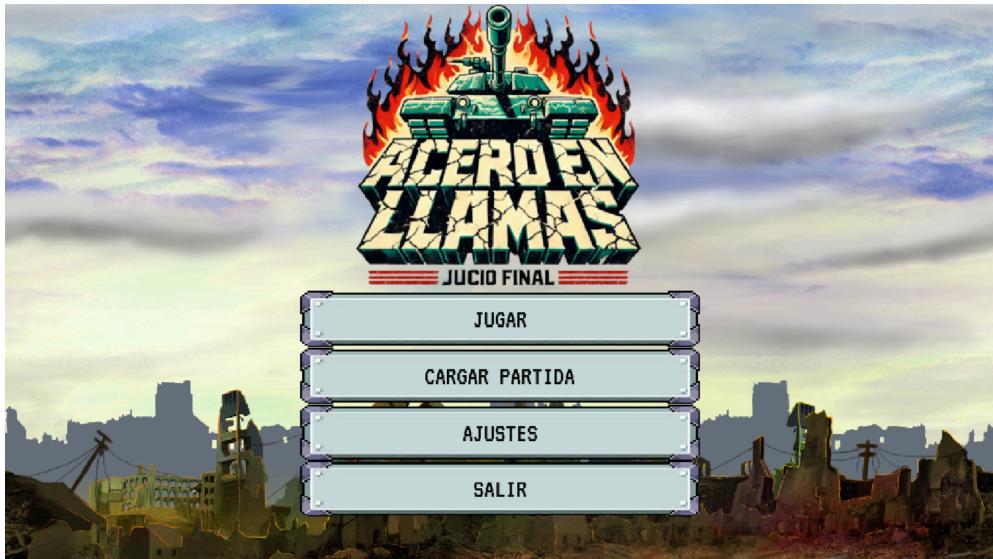
La música de fondo de los niveles cambia a la del jefe y viceversa en función de la posición del jugador en relación a la del jefe. Mencionar además que la música de los niveles fue compuesta por el equipo, dando un toque único al videojuego.

## **2.1.4 MANUAL DE USUARIO**

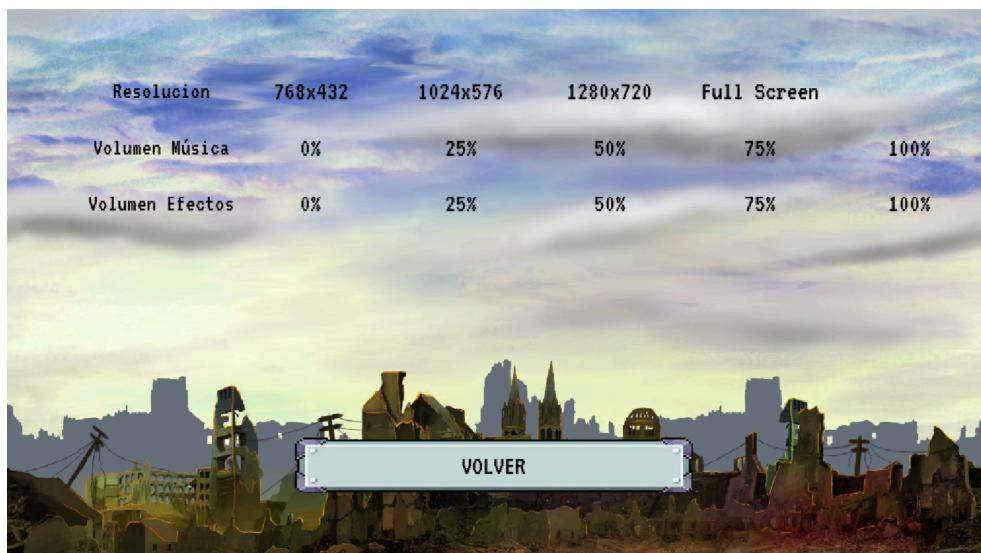
Además de los archivos del proyecto, en la entrega también se adjunta un archivo ejecutable que permite ejecutar el programa sin necesitar el resto de los archivos del juego.

Para ejecutar el proyecto es necesario tener instalados los siguientes módulos de python:

- pygame, el módulo principal que permite el funcionamiento del programa.
- csv, os, pathlib y re para localizar, escribir y leer correctamente archivos de recursos del proyecto, entre otras cosas.
- abc y enum para definir clases abstractas y enumeradores, respectivamente.
- math y numpy para el cálculo de la dirección de las balas.
- pickle para el funcionamiento del sistema de guardado.



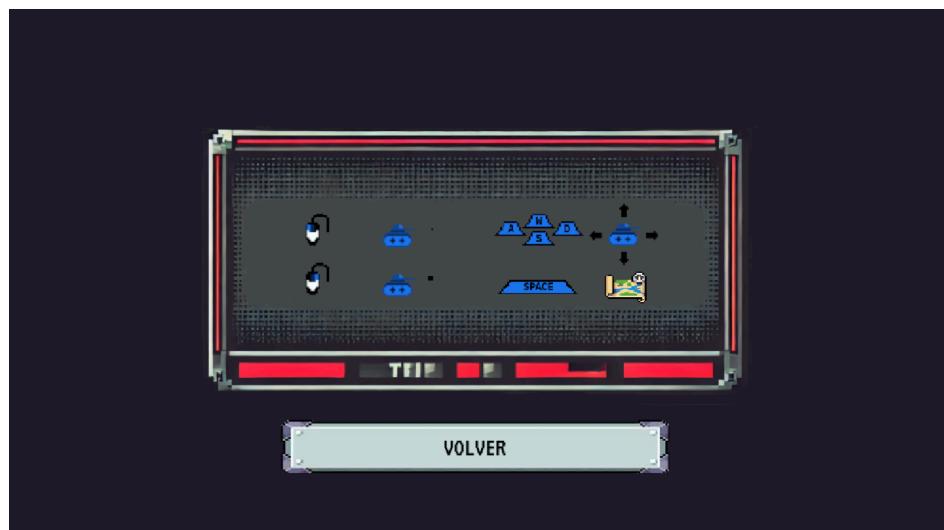
Para comenzar a jugar simplemente ejecutar el .exe. En la pantalla principal se puede cargar partida, seleccionar ajustes o empezar la partida.



En la pantalla de configuración podemos ajustar resolución, volumen de música y volumen de efectos.



En cualquier momento podremos pulsar Esc. y acceder al menú de pausa donde podremos: seguir la partida, cargar una partida guardada, guardar la actual, volver a la pantalla de título y entrar a la pantalla de controles.



En la pantalla de controles podremos recordar cómo se juega:

- A, W, S, D para movimiento
- Click izq para disparo normal
- Click der. para activar arma secundaria
- Espacio para abrir el minimapa

## 2.1.5 REPORTE DE BUGS

- A la hora de hacer el scroll de pantalla a veces no se detecta el cambio de pantalla del jugador, por lo que los enemigos no se actualizan uno se actualizan. Esto ocurre solo si se entra en la pantalla con un pequeño movimiento y el jugador no se mueve. Una posible solución a este problema implicaría modificar los valores donde se comprueba si el enemigo está en la misma pantalla
- No se detienen los temporizadores del juego mientras se está en el menú de pausa. Esto repercute en que se pueden pasar los tiempos de retardo en el menú dando

munición infinita tanto como para el jugador como para los enemigos. También afecta al funcionamiento del bombardeo del mundo 3, haciendo que aparezcan varias bombas y que exploten en cuanto se reinicia el juego. La solución sería implementar un sistema que guardase el tiempo antes de la pausa y lo recupere cuando se reanudará el juego.

- Cuando se dispara una bala que rebota al punto donde se unen dos elementos que rebotan, como pueden ser dos bloques, esta rebota en el hueco entre los dos. Esto resulta en la ilusión de que explota al contacto con el muro. La solución sería, por ejemplo, interpretar los objetos que están juntos como parte de una sola colisión.
- Cuando se finaliza el juego y se inicia una nueva partida, elementos como el contador de objetos clave obtenidos no se reinician correctamente y no aparecen si se han obtenido en la partida anterior. Para solucionar esto habría que mirar que se reinicien correctamente los valores y contadores de los discos y resetearlos correctamente.
- A la hora de guardar partidas en el ejecutable .exe no se conserva el archivo de partida guardada porque se guarda dentro del proyecto, para arreglarlo habría que guardarla dentro del sistema de archivos del SO.
- Si el jugador se sitúa en el límite entre una pantalla y la contigua los enemigos se detendrán y permite eliminarlos desde la distancia. Esto sucede por la implementación de A\*, que sólo comienza a funcionar si se detecta que el jugador está en la misma pantalla que el enemigo, al estar en el límite permanecen desactivados.
- Con el boss del mundo 1 existen distintos bugs: en ocasiones su movimiento se ve distorsionado, el sprite cambia de dirección múltiples veces. Si el jugador se sitúa pegado a una pared puede darse que el boss se pegue y quede bloqueado sin hacer daño, a veces incluso puede ser que el jugador se quede bloqueado contra el boss y no sea capaz de escapar. La solución a esto sería aumentar un poco la distancia de ataque del jefe.
- Con el algoritmo de búsqueda A\* se plantea otro problema: debido al diseño de los mapas, existen puntos en los cuales los enemigos no pueden seguir calculando el path para encontrar al jugador. Esto se debe a que el mapa binario que se emplea para el cálculo del path recibe las posiciones de los obstáculos de manera “inflada” para evitar la colisión visual de los enemigos contra paredes u otros enemigos. Es por esto que en pantallas donde existan cavidades donde los obstáculos “inflados” cierran completamente el paso hacia el jugador harán que la máquina de estados de los enemigos se detenga hasta que el jugador se vuelva a situar en una casilla alcanzable.