SMASH TV

Jose René Huacon Bravo
Marc Sevilla Hortelano
MP03 - Programación
01/06/2025 - Granollers
ESCOLA PIA GRANOLLERS - DAMVIOD

Índice

DOCUMENTO TÉCNICO

Controles básicos

El movimiento básico es con las flechas de direcciones y dispararemos con la tecla espacio. Los movimientos posibles son norte, sur, este y oeste,

Lista de objetivos cumplidos

En el Game Design Document describimos varios objetivos del juego, como puede ser el hecho de que haya varias salas, haya un boss al final de la sala y hayan power ups que ayuden al jugador. Por falta de tiempo no hemos podido completar esos puntos, pero por el contrario hemos conseguido los siguientes:

Enemigos

Hemos hecho un total de 3 enemigos: Blob, Grunt y Mine.

Su funcionamiento es básico, todos persiguen al jugador y tratan de hacer contacto con este para hacerle perder una vida. Las animaciones también son básicas debido a que el sprite sheet que usamos está hecho para que se repita en bucle.

El único que ha resultado más complejo ha sido blob. Por una parte por tema animación y por otra parte por tema funcionamiento. Esto es debido a que usamos su animación de muerte también para animación de movimiento.

Usamos los 3 primeros sprites de blob que simulan como un salto, y cuando acaba este cambia de marcha los sprites para volver al estado inicial. Esto lo hemos conseguido con una variable bool que determina el orden de animaciones según si está vivo o no. Si está vivo hará los 3 primeros frames de animación y luego irá marcha atrás. Si está muerto hará su animación completa que es la de muerte.

Hemos decidido esta manera ya que la tercera posición del sprite encaja perfectamente para simular que se está disparando una bala.

Movimiento del jugador

En el update del jugador tenemos la captura de teclas para detectar en qué movimiento va nuestro jugador. Se habilita este movimiento cuando el jugador está en el estado *is alive*.

Para controlar diversos problemas con la dirección se ha hecho lo siguiente

- Nos guardamos la última posición guardada en un *stack*
- En base a esta posición elegimos el sprite adecuado para que cuando este quieto se quede mirando esa dirección
- Cuando dispare y esté quieto también se quede disparando hacia esa misma dirección.

Movimiento de los enemigos

Los enemigos tienen la posición exacta de nuestro jugador. En base a estas coordenadas se acerca a ellos. Para hacer el movimiento más impredecible tiene una variable que decide si prioriza moverse por la coordenada x o si prefiere priorizar la coordenada y. Este valor cambia cada 60 frames.

Todos los enemigos funcionan de esta manera. Blob tiene la excepción de que en mitad de su animación dispara una bala hacia la dirección del jugador.

Funcionamiento de balas

La bala spawnea en la posición del jugador. Este elige la dirección según la última posición guardada (como hemos visto en jugador, que se guardaba la última posición para saber en qué dirección mirar).

Según en qué dirección está mirando actualiza su posición en cada frame, y al colisionar con un enemigo o al cabo de un tiempo desaparece. Al colisionar con un enemigo, también provoca la desaparición de este.

Manejo de escenas

Tenemos un vector de escenas donde vamos almacenando todas las escenas del juego. En este caso cada escena es una clase. En el update de cada escena está la lógica para controlar cómo cambiar de escena, por ejemplo, en Menu.cpp cambiamos de escena a Map. Esto lo conseguimos gracias al SceneDirector, que nos ayuda a movernos en escenas mediante un enum de escenas.

Manejo de puntuación / High score

Al guardar la puntuación con ficheros binarios tenía que buscar una manera de saber cuántas entradas iba a tener mi documento y saber cuánto iba a medir. Para ello se ha decidido que el límite de caracteres del jugador será de 5.

Esto es debido a que cada carácter vale 1 byte, el carácter que marcará el final de la cadena valdrá 1 byte El int donde se guardará la puntuación y por último, el número donde guardaremos la puntuación será un int, 4 bytes. La suma de estos elementos nos dará un total de 10 bytes.

Este número es ideal para calcular el número de entradas, ya que si dividimos el size entre 10 siempre nos dará el número de entradas que habrá.

Estas entradas las almacenaremos en un struct que tendrá los elementos de puntuación y nombre. La razón de este struct es porque haremos una instancia como vector y allí podremos aplicar el algoritmo de bubble sort para poder hacer un ranking de puntuación.

Manejo de texto en la pantalla

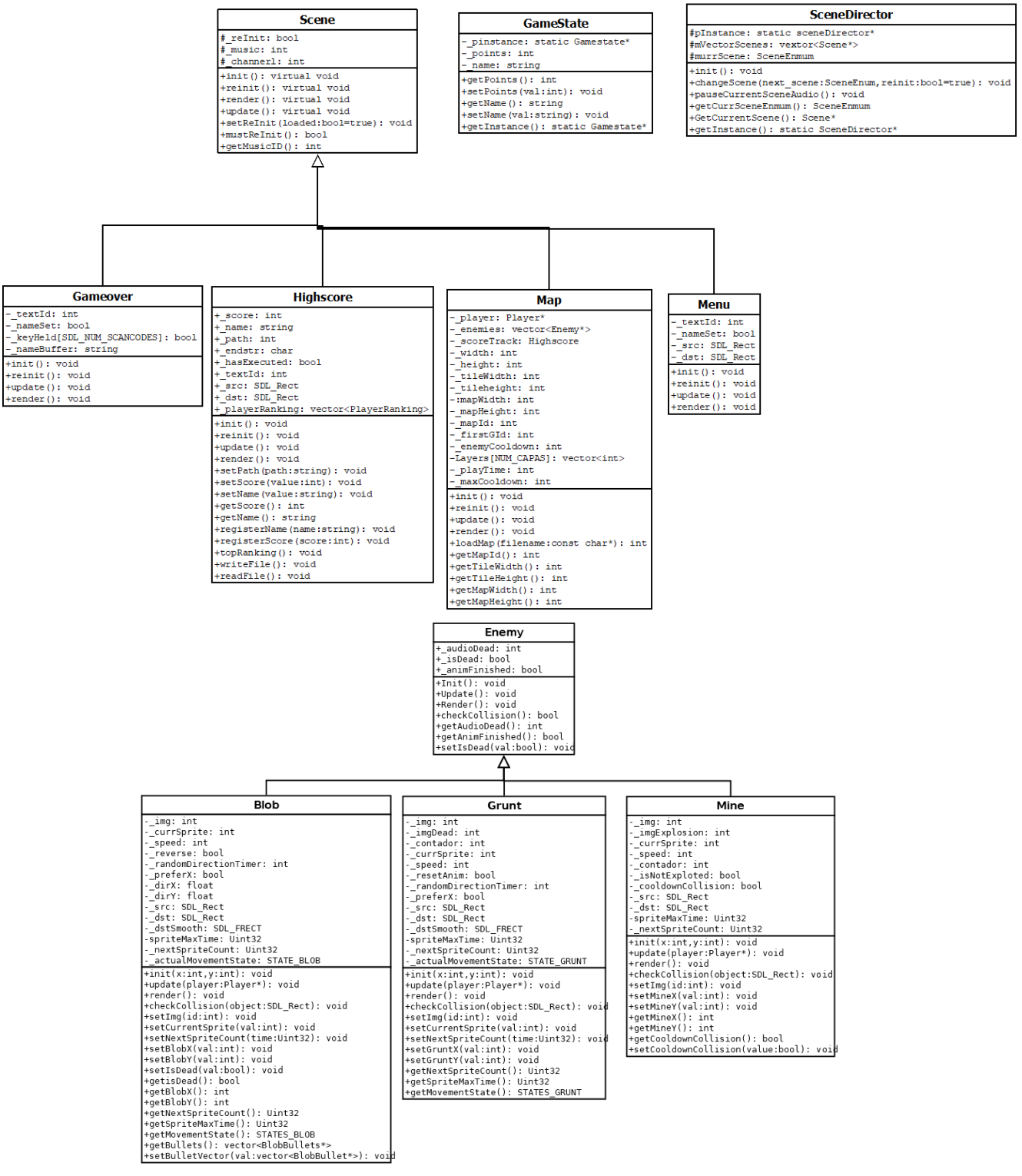
Para el manejo de teclas de pantalla hemos hecho uso de una librería llamada ttf. Esta es una librería que nos ayuda a mostrar cadenas de texto por la pantalla y aplicarles una fuente personalizada.

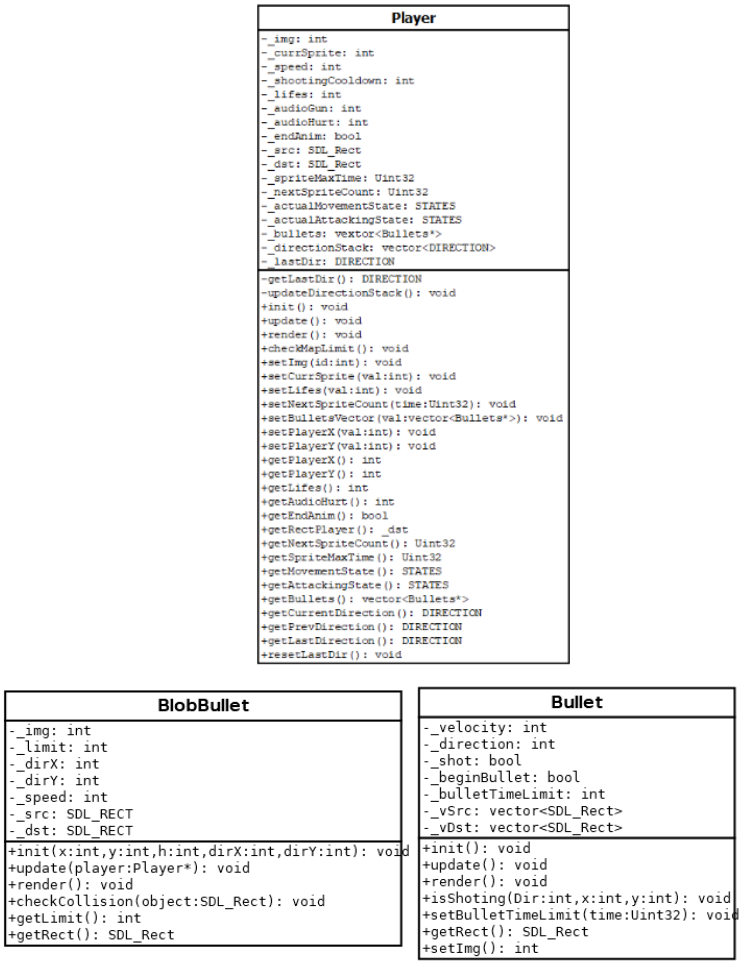
Todo el tema de inicialización está implementado dentro de nuestro engine, concretamente en la clase video.

Para la captura de teclas en tiempo real hicimos que en el bucle de la clase jugador hubiese captura de teclas. En función del ENUM de las teclas iremos sumando carácter a la letra A. La letra A al ser el primer índice de caracteres facilita mucho el proceso, debido a que con el enum solo tenemos que sumarle cuantas posiciones de diferencia hay de A y como resultado tendremos la tecla que hemos pulsado.

Luego todo este bucle lo que hará será sumarlo a un buffer de caracteres y cuando le demos a la tecla enter lo que hará será escribir los valores en el gamestate, que se encargará de escribir en los ficheros binarios.

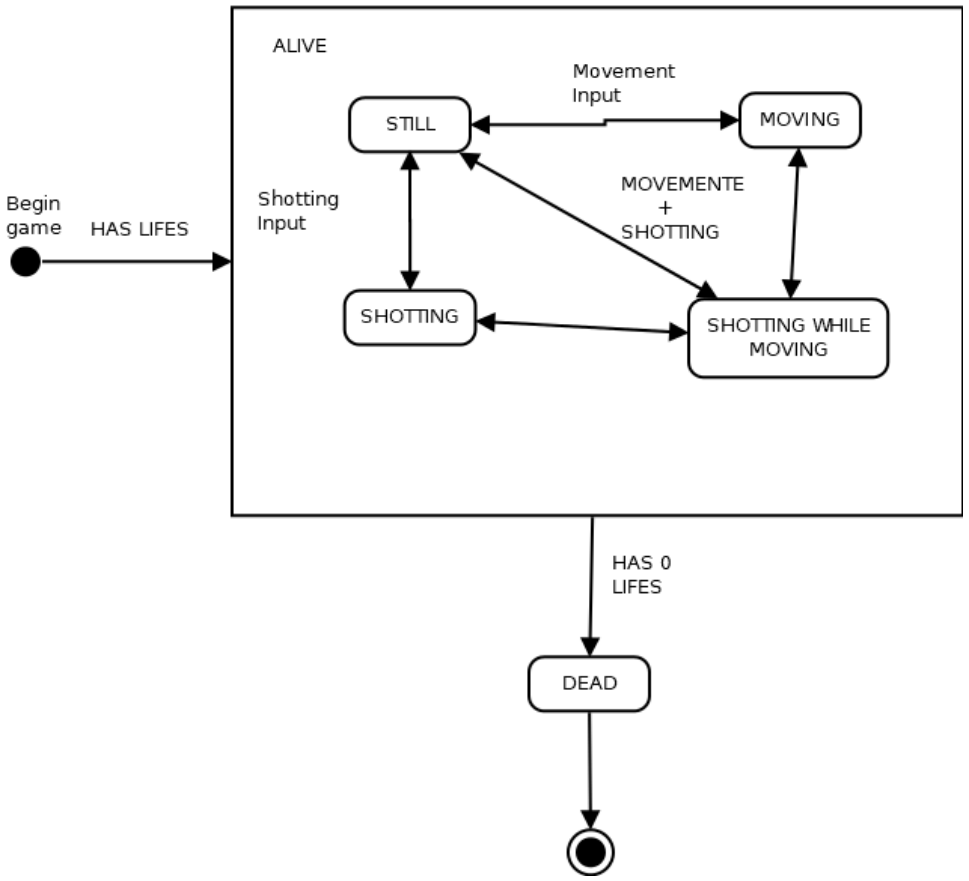
Diagrama de clases



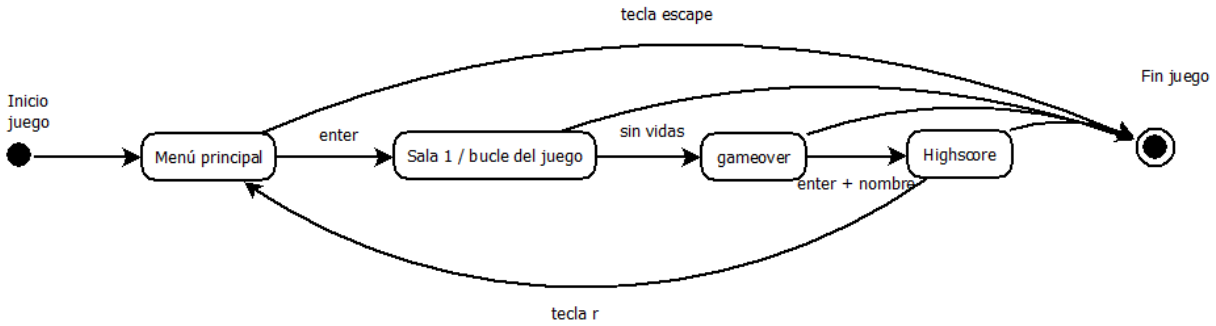


- Las pantallas principales son Gameover, highscore, map y menú
- Estas pantallas heredan de Scene, que se encarga del control de escenas
- Scene lo controla SceneDirector. Tiene un vector donde se guardan las escenas y dependiendo de en qué situación estamos (por ejemplo, de menú a bucle) cambia la escena actual.
- GameState se encarga de almacenar la información de la partida, en este caso, del nombre del jugador y del número de puntos obtenidos.
- Todos los enemigos anteriormente mencionados heredan de la clase Enemy
- Estos usarán principalmente el método de check colisión con jugador para detectar colisiones.

Diagrama de estados personaje



Flujo de escenas



- Al ejecutar el juego con la tecla enter nos llevará al bucle principal del juego
- Iremos a la pantalla de game over una vez nos quedamos sin vidas
- En la pantalla de game over podremos escribir nuestro nombre y nos mandará al high score
- En la pantalla de high score si apretamos la tecla r nos devolverá al menú principal, repitiendo el bucle del juego.