PROYECTO FINAL - PARADIGMAS DE LA PROGRAMACIÓN

Matteo Ferrari Marín, Mario Alonso Alonso, Santiago Urtiaga – 3ºA

INTRODUCCIÓN:

El trabajo consistía en crear un videojuego en Unity empleando el lenguaje de programación C# orientado a objetos. Se ha optado por el famoso videojuego Mario Bros, un juego plataforma en 2D, incluyéndose 3 niveles siguiendo su propio estilo, y, además, un modo versus uno contra uno.

Al ser un trabajo colaborativo, se ha hecho uso de Github como entorno para compartir las implementaciones que ha realizado cada miembro del grupo con el resto. Esta herramienta también permite que varios miembros trabajen al mismo tiempo, cada uno desde su propio entorno.

OBJETOS DEL JUEGO:

- Bloques rígidos: objetos estáticos con colisión sin otra finalidad que ser apoyo para las entidades del juego.

- Fondo: objetos estáticos sin colisión que decoran el entorno.

- Power ups: entidades especiales que el jugador puede recoger para ganar habilidades que tienen el objetivo de facilitar el nivel. Se han implementado 2: champiñón y flor de fuego, explicados más adelante.

- Bloques misteriosos: objetos estáticos que, al ser colisionados por el jugador, hacen aparecer algún power up.

- Monedas: coleccionables estáticos que se pueden recoger a lo largo del nivel. Recolectar 100 de estas otorga una vida adicional al jugador.

- Tuberías: objetos estáticos emparejados que pueden interactuar con el jugador, teletransportándolo bidireccionalmente.

- Bandera: objeto especial estático que se sitúa al final de cada nivel. Lo último que hace el jugador antes de pasarse el nivel es interactuar con ella.

- Castillo: se encuentra justo detrás de la bandera, y al alcanzarlo el jugador avanza al siguiente nivel, que se encuentra en una escena distinta al nivel actual.

- Interfaz de Usuario: se trata de un panel informativo que comunica al jugador cuántas monedas ha recogido, cuántas vidas le quedan, y en que nivel se encuentra.

- Enemigos: entidades que se desplazan por el mapa. Se han implementado 3:
    - Goomba: comienza a desplazarse hacia la izquierda, y cuando colisiona con cualquier objeto del mapa o con otro enemigo cambia de dirección. Si colisiona horizontalmente con el jugador, el jugador es dañado, pero si colisiona por arriba con él, el Goomba cae eliminado.

    - Koopa: comienza a desplazarse como el Goomba, pero cuando el jugador se encuentra cerca, este le empieza a perseguir. Daña al jugador de la misma forma, pero si este colisiona por arriba con él, en vez de caer eliminado el Koopa se resguarda en su concha. Si en este estado el jugador salta encima de la concha, el Koopa es eliminado definitivamente, pero, si el jugador colisiona horizontalmente con ella, la concha comienza a desplazarse rápidamente de lado a lado, eliminando a toda entidad que se cruce en su paso, incluido el jugador. Por otro lado, si pasa un tiempo en el que la concha no cae eliminada ni se desplaza, el Koopa sale de ella, volviendo así al estado inicial.

    - Patroopa: es un enemigo volador que se desplaza de arriba a abajo constantemente. De nuevo, si colisiona horizontalmente con el jugador, este caerá eliminado, pero si el jugador salta encima, el Patroopa se convertirá en un Koopa, perdiendo su capacidad de volar para siempre.

ESTADOS:

- Mario pequeño: es el inicial, y mide un bloque de alto. Si el jugador recibe daño en este estado, se reinicia el nivel (reiniciando la escena) y pierde una vida, o, en su defecto, vuelve al menú principal.

- Mario grande: se obtiene al recoger el power up de champiñón, y mide el doble que el Mario pequeño. Si el jugador recibe daño en este estado, pasa al estado de Mario pequeño.

- Mario de fuego: se obtiene al recoger el power up de flor, y mide igual que el Mario grande. Si el jugador recibe daño en este estado, pasa al estado de Mario grande.

CONTROLES:

- W: Saltar. El salto puede ser de distintas alturas, dependiendo del tiempo que el botón esté pulsado.

- A, D: Moverse a la izquierda y a la derecha respectivamente.

- S: Utilizar tubería. Solo tiene funcionalidad cuando el jugador está situado en el centro de alguna tubería.

- Espacio: Lanzar bola de fuego. Solo disponible cuando el jugador se encuentra en el estado de Mario de fuego.

- Flechas: Movimiento del segundo jugador en el modo versus.

DESARROLLO:

- Descarga de sprites: Los sprites son diseños gráficos que constituyen la sección visible del videojuego, es decir, las texturas que ve el usuario cuando está jugando.

- Implementación del movimiento básico: se crea el Mario, es decir, la entidad que será controlada por el usuario. Se le aplican físicas, colisiones, y la capacidad de desplazarse por el mapa para poder probar las mecánicas e interacciones con el resto de los objetos.

- Creación de prefabs: Los prefabs son objetos que se construyen una vez, se almacenan, y pueden ser reutilizados tantas veces como se quiera. Esta sección es la más larga, ya que incluye la implementación de la lógica de absolutamente todos los objetos y entidades que se van a usar para construir el mapeado. La interacción de cada uno de estos tanto con otros objetos como con el propio jugador debe estar concebida.

- Animar las entidades: Para una mejor experiencia visual, y también para transmitir información relevante al usuario, se implementan animaciones tanto en el Mario como en otras entidades, como los enemigos.

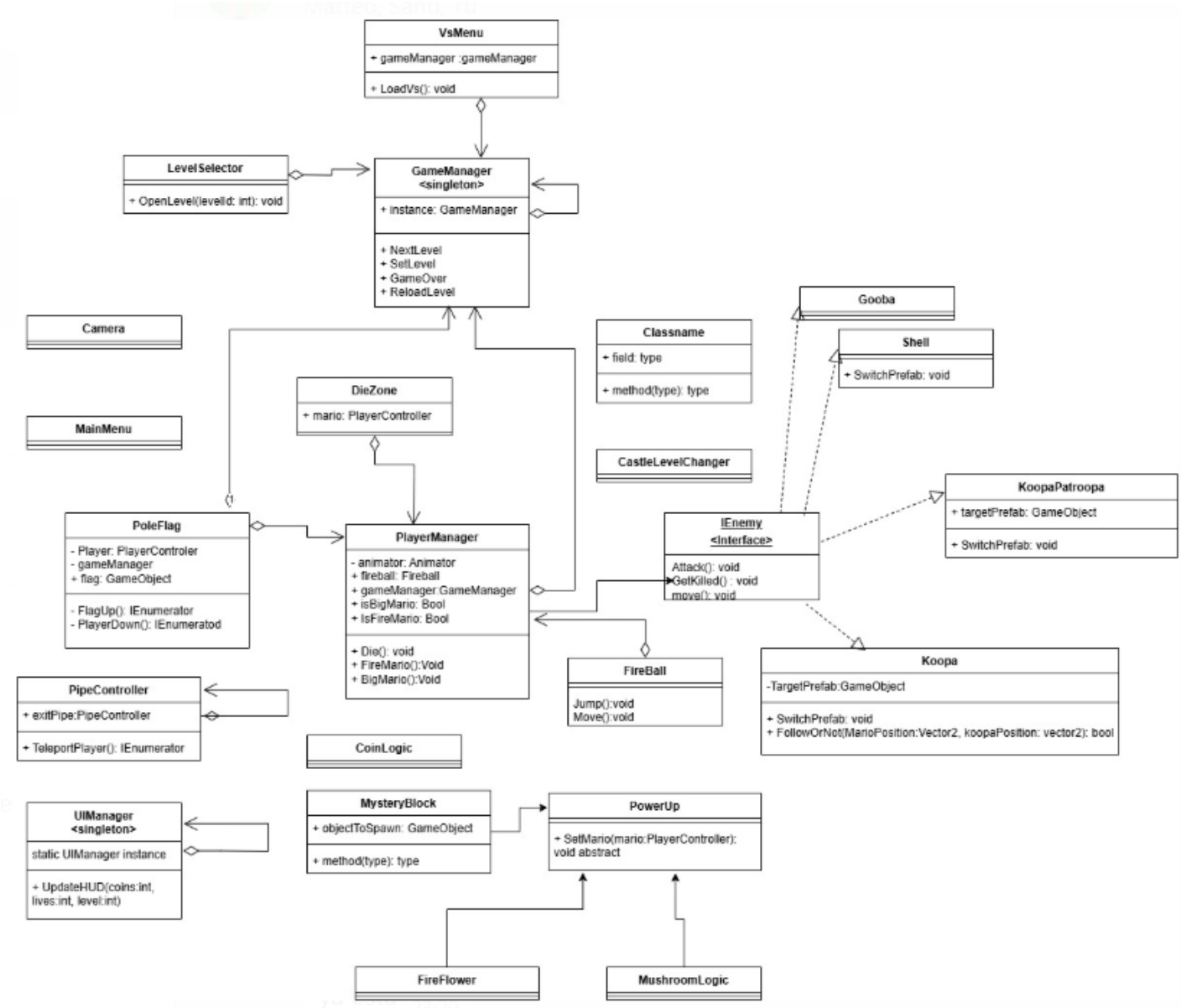
- Diseño de niveles: Una vez se obtienen todos los elementos necesarios, se puede comenzar a diseñar niveles con total libertad y sin límites, cada uno de ellos en su escena particular.

- Desarrollar el modo versus: De la misma forma, podemos emplear el movimiento y los objetos ya disponibles para diseñar nuevos modos de juego con un funcionamiento distinto al principal.

- Adición de menú: Aparecerá al ejecutar el juego, y en él se podrá elegir qué modo de juego jugar, y, en el caso de seleccionar el principal, el usuario seleccionará el nivel que prefiera.

- Implementación de sonido: Finalmente, se añaden efectos de sonido para cada situación significativa que se puede dar en el transcurso del juego.

UML:



PROBLEMAS

- Mergear ramas en Github: en ocasiones, unir las ramas en las que cada miembro estaba trabajando era una tarea demasiado tediosa por culpa de las incompatibilidades que se generaban, e, incluso, imposible, teniendo que volver a una versión antigua o combinar versiones manualmente.

- División del juego en escenas: al principio esta tarea podía llegar a ser algo engorrosa. Sin embargo, el resultado final es mucho más organizado, y, sobre todo, más escalable.

- Solucionar grandes cantidades de casos: al haber muchos tipos de interacciones que deben ser tenidas en cuenta una a una, se ha invertido mucho tiempo en la solución de errores en el juego.

- Implementación de patrones de diseño: cuando se ha creído conveniente, se ha programado siguiendo algunos patrones de diseño, como las factorías o singleton.