



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

**título del TFG
Documentación Técnica**



Presentado por nombre alumno
en Universidad de Burgos — 6 de junio de 2021
Tutor: nombre tutor

Índice general

| | |
|---|------------|
| Índice general | i |
| Índice de figuras | iii |
| Índice de tablas | v |
| Apéndice A Plan de Proyecto Software | 1 |
| A.1. Introducción | 1 |
| A.2. Planificación temporal | 2 |
| A.3. Estudio de viabilidad | 3 |
| Apéndice B Especificación de Requisitos | 7 |
| B.1. Introducción | 7 |
| B.2. Objetivos generales | 7 |
| B.3. Catalogo de requisitos | 7 |
| B.4. Especificación de requisitos | 10 |
| Apéndice C Especificación de diseño | 23 |
| C.1. Introducción | 23 |
| C.2. Diseño de datos | 23 |
| C.3. Diseño procedimental | 23 |
| C.4. Diseño arquitectónico | 23 |
| Apéndice D Documentación técnica de programación | 25 |
| D.1. Introducción | 25 |
| D.2. Estructura de directorios | 25 |
| D.3. Manual del programador | 27 |

| | |
|--|-----------|
| D.4. Compilación, instalación y ejecución del proyecto | 27 |
| D.5. Pruebas del sistema | 27 |
| Apéndice E Documentación de usuario | 29 |
| E.1. Introducción | 29 |
| E.2. Requisitos de usuarios | 29 |
| E.3. Instalación | 29 |
| E.4. Manual del usuario | 29 |
| Apéndice F Documento de diseño del juego | 31 |
| F.1. Introducción | 31 |
| F.2. Mecánicas del juego | 33 |
| F.3. Niveles | 36 |
| F.4. Interfaces | 40 |
| F.5. Entidades | 41 |
| F.6. Controles | 52 |
| Bibliografía | 55 |

Índice de figuras

| | |
|--|----|
| A.1. Captura de pantalla del tablero Kanban utilizado | 3 |
| A.2. Tabla de Excel con el sueldo calculado del trabajador | 4 |
| A.3. Tabla de Excel con los activos del proyecto | 4 |
| A.4. Tabla de Excel con el balance del proyecto | 5 |
| A.5. Tabla de Excel con el balance del proyecto con 3 meses más de desarrollo | 5 |
| A.6. Tabla de Excel con el balance del proyecto con 6 meses más de desarrollo | 6 |
| B.1. Diagrama de casos de uso | 10 |
| F.1. Nivel tutorial de los portales | 37 |
| F.2. Pantalla de prueba de mecánicas básicas | 38 |
| F.3. Pantalla de prueba de los portales | 38 |
| F.4. Pantalla de prueba de los obstáculos móviles | 38 |
| F.5. Pantalla de prueba de los creadores de impulso | 39 |
| F.6. Pantalla de prueba de los modificadores de gravedad | 39 |
| F.7. Pantalla de prueba de los modificadores temporales | 40 |
| F.8. Idea inicial de la pantalla de selección de nivel | 40 |
| F.9. Idea inicial de la pantalla de opciones | 41 |
| F.10.Sprite utilizado para el Player | 42 |
| F.11.Sprite utilizado para el obstáculo estático | 43 |
| F.12.Sprite utilizado para el obstáculo que sigue una rutina | 44 |
| F.13.Obstáculo móvil rápido | 44 |
| F.14.Obstáculo móvil (velocidad intermedia) | 45 |
| F.15.Obstáculo móvil lento | 45 |
| F.16.Sprite utilizado para los portales | 46 |
| F.17.Sprite utilizado para la partícula de impulso | 47 |

| | |
|---|----|
| F.18.Sprite utilizado para el amplificador de impulso | 48 |
| F.19.Sprite utilizado para el amplificador de impulso | 49 |
| F.20.Sprite utilizado para la zona de tiempo escalado | 50 |
| F.21.Sprite utilizado para el inversor de gravedad | 51 |
| F.22.Sprite utilizado para el obstáculo superdenso | 51 |
| F.23.Nomenclatura de los mandos de PlayStation | 52 |

Índice de tablas

Apéndice A

Plan de Proyecto Software

A.1. Introducción

La metodología de trabajo seguida ha sido la metodología Kanban [6]. El método Kanban toma la filosofía de las metodologías evolutivas e incrementales haciendo énfasis en la intención de la entrega del producto justo a tiempo.

Uno de los elementos clave de esta metodología es el tablero Kanban [4]. Este tablero esta dividido en una serie de apartados definidos que representarán el flujo de trabajo de las tareas a realizar. A este tablero se van añadiendo las tareas que se van a realizar, y estas van viajando de una fase de desarrollo de la tarea a otra. Habrá una fase de desarrollo final que represente que esa tarea esta completada que será la última fase a la que accedan todas las tareas.

Las tareas del tablero Kanban pueden ser clasificadas en distintas etiquetas según el ámbito de la tarea.

Un videojuego es un producto software cuyo grado de completitud es difícil de estimar, un bug puede retrasar mucho la ejecución. Es además frecuente que no se cumplan los plazos de desarrollo de los videojuego. Los tableros Kanban puede resultar útil para enfrentar estos problemas ya que son útiles para:

- Ayudar a visualizar el flujo de trabajo, facilitando identificar adelantos y retrasos y conocer el estado del proyecto.
- Identificar y reducir cuellos de botella.

- Es muy compatible con la integración continua.

A.2. Planificación temporal

El tablero Kanban diseñado para este proyecto es uno en el que las tareas pueden estar en cuatro posibles fases de desarrollo:

- **To do:** En esta fase se almacenan las tareas que han sido identificadas y se tiene planeado llevar a cabo pero cuyo desarrollo no ha comenzado todavía.
- **In progress:** La tarea esta siendo llevada a cabo.
- **QA/Testing:** Fase de revisión de la tarea. Se comprueba que se ha realizado correctamente. Una tarea también puede llegar a esta fase debido a que se desee una revisión de esta tarea, pero no con la intención de darla por completada, sino para que de esta revisión surjan soluciones a problemas surgidos durante la realización de la tarea o debatir sobre lo que pide implementar la tarea (como por ejemplo no estar de acuerdo en la necesidad de utilizar una herramienta para llevar a cabo la tarea).
- **Done** Esta es la fase final del tablero. Todas las tareas que llegan a esta fase se dan por terminadas.

El tablero Kanban se puede encontrar en el repositorio de Github del proyecto en el apartado de "Projects"(<https://github.com/Kencho/mri1001-tfg/projects/1>)

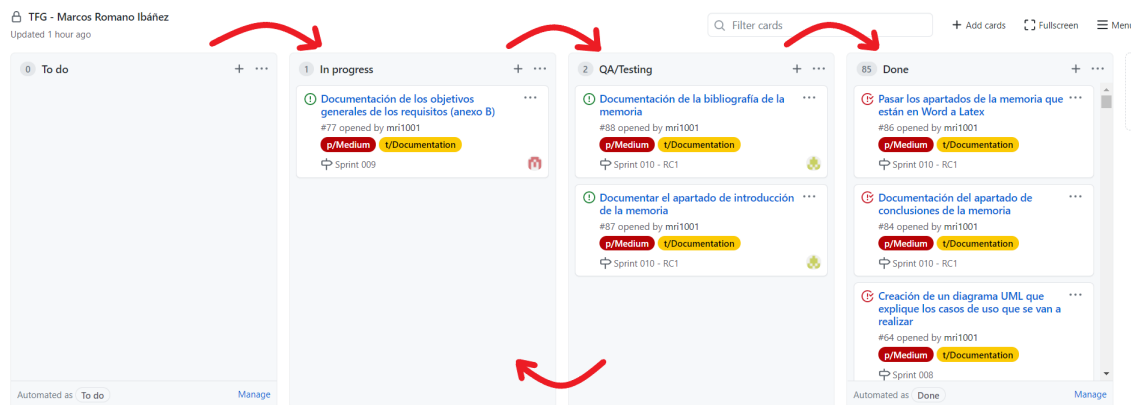


Figura A.1: Captura de pantalla del tablero Kanban utilizado

A.3. Estudio de viabilidad

A continuación se va a proceder a realizar el estudio de viabilidad del proyecto. Dejar claro que este estudio de viabilidad no va a ser representativo de un estudio de viabilidad de un videojuego, ya que este TFG se ha centrado sobre todo en el punto de vista del programador obviando el resto de vertientes del proyecto como el apartado artístico o sonoro en los cuales se ha hecho uso de recursos de la comunidad gratuitos, de manera que los sueldos y otros gastos asociados a estos trabajadores no van a constar en el estudio de viabilidad económica.

Otro tema que diferirá de otros videojuegos desarrollados con Unity será que si en el último ejercicio fiscal los ingresos fueron superiores a 100.000 dolares americanos, se esta obligado a comprar la edición profesional de Unity [5]. Al no comercializar el juego no es necesario estar pendiente de estos gastos. Presuponiendo también que es el primer producto desarrollado con Unity no habrá precedentes que obliguen a obtener la licencia profesional de Unity.

Viabilidad económica

Se va a valorar la viabilidad económica del proyecto desde el punto de vista económico. Para ello se evaluarán dos aspectos: los costes y los beneficios.

Los cálculos se harán asumiendo que el proyecto lo lleve a cabo una empresa y que se tenga liquidez suficiente como para no requerir financiamiento.

Costes

Costes de recursos humanos: El salario medio de un programador de videojuegos en España es de 32.100 € [3]. Sin embargo el sueldo varía mucho de un programador senior a uno junior. Se va a asumir que en este caso el programador es junior (es un alumno de 4º). El sueldo neto medio mensual de un programador junior es de 1.160 €.

| Categoría profesional | Salario base mensual | SS a cargo de la empresa | SS a cargo del trabajador | SS total | IRPF | Coste total mensual | Cote total 5 meses |
|-----------------------|----------------------|--------------------------|---------------------------|----------|------|---------------------|--------------------|
| Programador Junior | 1160 | 29,90% | 6,35% | 36,25% | 30% | 1856 | 9280 |

Figura A.2: Tabla de Excel con el sueldo calculado del trabajador

Activos: El único activo que ha sido necesario para el desarrollo del proyecto ha sido el ordenador portátil en el que se ha creado el videojuego. Se estima que la vida útil de este activo será de 5 años, con lo que se amortizará en ese tiempo. En los 5 meses que ha durado el proyecto el coste de amortización del portátil es de 100 €

Las ventas no superarán los 100.000 €, así no hará falta pagar la versión profesional de Unity, resultando el software utilizado gratuito.

| ACTIVOS | Año 1 |
|--------------|-------------|
| Ordenador | 1.200 |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| TOTAL | 1200 |

Figura A.3: Tabla de Excel con los activos del proyecto

Beneficios

La media de unidades vendidas de por los videojuegos indies (desarrollados por muy pocas personas) esta en torno a las 1.500 € unidades [2]. Si se

Figura A.6: Tabla de Excel con el balance del proyecto con 6 meses más de desarrollo

En cuanto a la viabilidad legal, afortunadamente para este proyecto es sencillo el marco legal.

Los elementos artísticos utilizados en el videojuego han sido obtenidos de personas que los han cedido para su libre uso. Todo este contenido se han regido por las distintas leyes de CopyRigth, pero todas las estas leyes permiten su libre uso pero requiriendo mencionar al autor. Esas leyes son:

- Attribution 4.0 International (<https://creativecommons.org/licenses/by/4.0/>)
- Attribution 3.0 Unported (<https://creativecommons.org/licenses/by/3.0/>)
- Attribution 2.0 Generic (<https://creativecommons.org/licenses/by/2.0/>)
- Attribution 1.0 Generic (<https://creativecommons.org/licenses/by/1.0/>)

Apéndice B

Especificación de Requisitos

B.1. Introducción

B.2. Objetivos generales

El desarrollo del proyecto busca lograr los siguientes objetivos:

- Desarrollar un fichero de diseño del juego que resuma en qué va a consistir el juego y que mecánicas implementará.
- Desarrollar un videojuego que implemente las mecánicas definidas en el fichero de diseño del juego.
- Ofrecer versiones del producto final (el videojuego) para Windows, Linux y WebGL.
- Ofrecer un juego controlable tanto con teclado y ratón como con mando.

B.3. Catalogo de requisitos

Requisitos funcionales

Los requisitos funcionales especificarán cual es el funcionamiento que se espera del producto. Se concretarán a continuación.

- **RF-1 Gestión de menús:**El jugador deber poder navegar por los menús.

- **RF-1.1 Navegación entre pantallas:** Las pantallas deben de permitir navegar a otras pantallas (de manera directa o indirecta).
- **RF-2 Gestión del menú principal:** Deberá haber un menú principal que sea la primera escena que se muestre en el videojuego.
 - **RF-2.1 Selección de nivel:** Debe de poderse acceder a cualquier nivel desde el menú principal.
 - **RF-2.2 Cierre de la aplicación:** Se debe de poder cerrar la aplicación desde el menú principal.
 - **RF-2.3 Viaje al menú de opciones:** Se puede viajar desde el menú principal al menú de opciones.
- **RF-3 Gestión del menú de opciones:** Deberá haber un menú de opciones que permita modificar aspectos generales del juego. Desde el menú de opciones se debe poder volver al menú principal.
- **RF-4 Gestión de niveles:** Los niveles deben de contener todos los elementos necesarios y obligatorios de un nivel.
 - **RF-4.1 Controlar un avatar:** Debe de haber un avatar controlable por el jugador que realice las operaciones que le indique el jugador.
 - **RF-4.2 Zonas de muerte:** Debe haber zonas que maten al avatar del jugador cuando entre en contacto con ellas y devuelvan el nivel a un estado inicial.
 - **RF-4.3 Zonas de victoria (meta):** Debe haber zonas en las que, al entrar, se considere el nivel terminado y se vuelva al menú principal.
 - **RF-4.4 Gestión del menú de pausa:** Se debe poder acceder al menú de pausa, que parará la ejecución del nivel.
 - **RF-4.4.1 Abrir el menú de pausa:** Se deberá de poder abrir el menú de pausa, lo que pausara la ejecución del nivel.
 - **RF-4.4.2 Operaciones del menú de pausa:** Se deben de poder realizar las mismas operaciones que en el menú de opciones.

- **RF-4.4.3 Cerrar el menú de pausa:** Se deberá poder cerrar el menú de pausa, lo que reanudará la ejecución del nivel.
- **RF-4.5 Manipulación de la gravedad:** Debe de ser posible manipular la gravedad del nivel.
- **RF-4.6 Manipulación del tiempo:** Debe de ser posible modificar la escala de tiempo que afecta al nivel.
- **RF-4.7 Aplicación de impulsos:** Se debe poder aplicar impulsos que varíen la trayectoria que lleva un objeto.
- **RF-4.8 Obstáculos:** Puede haber obstáculos que maten al avatar jugable cuando entre en contacto con ellos.
- **RF-4.9 Portales:** Puede haber portales que teletransporten al avatar jugable desde el punto en el que se encuentra a otro que corresponderá con otro portal.
- **RF-5 Gestión del avatar del jugador:** El jugador debe contar con un avatar controlable por el este. El avatar debe ser capaz de: saltar, moverse, realizar un acelerón y activar el tiempo bala.

Requisitos no funcionales

Al ser el producto a entregar un videojuego (un software de ocio), resulta clave especificar que requisitos no funcionales se van a tener en cuenta.

- **Facilidad de uso:** Un videojuego cuyo uso no sea sencillo y cómodo puede perder muchos jugadores por esa única razón. Es clave que los jugadores tengan una experiencia agradable cuando interactúen con el juego.
- **Soporte:** Los cambios y actualizaciones del videojuego tienen que ser transparentes al usuario, pues no tiene necesidad de conocer como funciona internamente el juego, sino solo abrir el ejecutable y disfrutar del juego.
- **Apariencia o interfaz externa (*look and feel*):** No se ha especificado un diseño de interfaz, sin embargo en el mundo de los videojuegos hay un modelo general muy establecido. Es lógico adoptarlo para ofrecerle al jugador una experiencia que le resulte familiar.

- **Escalabilidad:** En un videojuego se van a ir añadiendo continuamente funcionalidades sobre el código para poder implementar todas las mecánicas, tanto definidas como que puedan surgir en el futuro. Por ello el código debe ser fácil de mantener y extender.

B.4. Especificación de requisitos

Diagramas de casos de uso

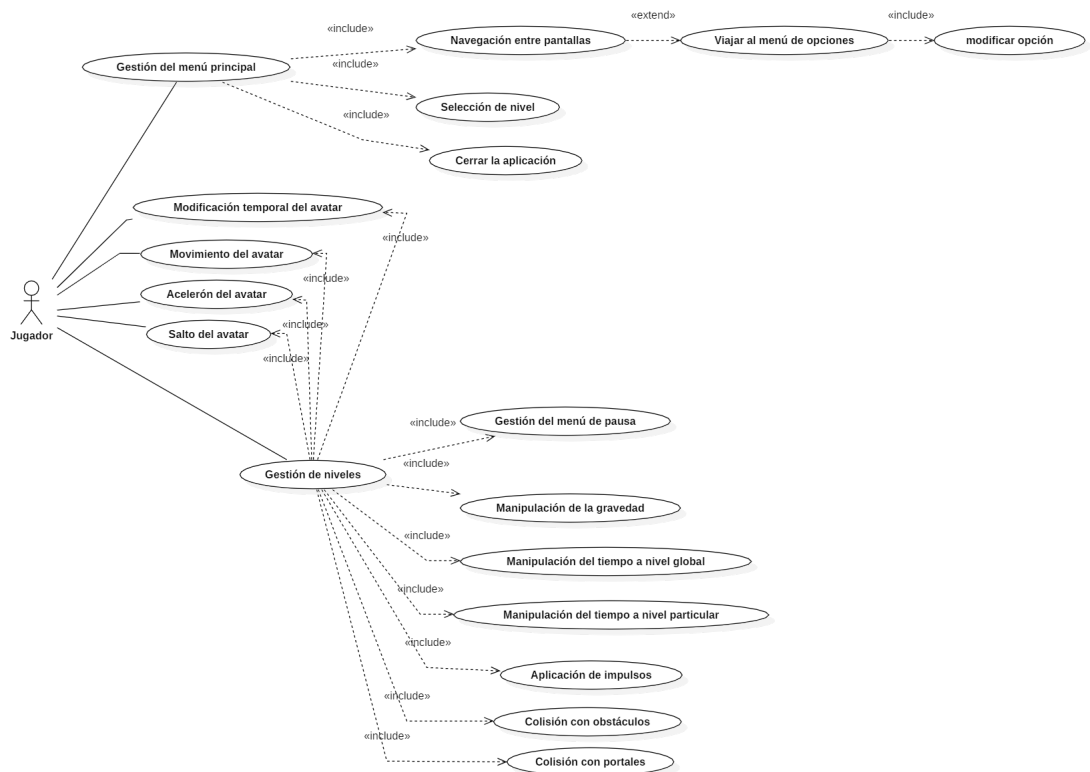


Figura B.1: Diagrama de casos de uso

Actores

Solo habrá un actor: el jugador del videojuego.

Casos de uso

Caso de uso 01

| | |
|-------------------------------|--|
| CU-01 | Gestión del menú principal |
| Descripción | Permite al usuario seleccionar escena, cerrar el juego y acceder al menú de opciones |
| Requisitos funcionales | RF-1, RF-1.1, RF-2, RF-2.1, RF-2.2, RF-2.3 |
| Precondición | Haber ejecutado el juego |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. Mostrar todos los niveles que se pueden jugar 2. Mostrar el botón de transición al menú de opciones 3. Mostrar el botón cierre del programa |
| Postcondiciones | Haber transicionado a otra escena |
| Excepciones | Si se pulsa el botón de cierre del programa parar la ejecución del programa en vez de transicionar a otra escena |
| Frecuencia | Muy Alta |
| Importancia | Alta |

Caso de uso 02

| | |
|-------------------------------|--|
| CU-02 | Navegación entre pantallas |
| Descripción | Permite al usuario cambiar de la escena actual a la escena escogida |
| Requisitos funcionales | RF-1, RF-1.1, RF-2.1, RF-2.3, RF-3, RF-4.3, RF-4.4.3 |
| Precondición | Tener una escena a la que se desea cambiar |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. Cargar la escena a la que se va a transicionar 2. Inicializar la escena 3. Cambiar a la escena a la que se ha transicionado |
| Postcondiciones | Haber transicionado a otra escena |
| Excepciones | No las hay |
| Frecuencia | Muy Alta |
| Importancia | Alta |

Caso de uso 03

| | |
|-------------------------------|--|
| CU-03 | Selección de nivel |
| Descripción | Permite al usuario elegir el nivel que desea jugar |
| Requisitos funcionales | RF-1, RF-2, RF-2.1 |
| Precondición | Estar en el menú principal |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. Se muestra al jugador todos los niveles que se puede jugar 2. El usuario puede desplazarse entre los botones de selección de nivel 3. El jugador se sitúa sobre el botón de selección del nivel que desea jugar 4. El jugador pulsa el botón de acceso al nivel que desea jugar 5. Se inicializa el nivel seleccionado 6. El jugador entra al nivel seleccionado |
| Postcondiciones | Haber inicializado el nivel correctamente |
| Excepciones | Si el nivel no se ha inicializado correctamente. Si no se ha asignado ninguna escena a la que transicionar al pulsar el botón de selección de niveles |
| Frecuencia | Muy Alta |
| Importancia | Alta |

Caso de uso 04

| | |
|-------------------------------|---|
| CU-04 | Cerrar la aplicación |
| Descripción | Permitir al usuario cerrar el juego desde el menú principal |
| Requisitos funcionales | RF-1, RF-2, RF-2.2 |
| Precondición | Estar en el menú principal |

| | |
|---------------------------|---|
| Secuencia de pasos | <ol style="list-style-type: none"> 1. Se muestra al jugador todos los botones con los que puede interactuar 2. El usuario se desplaza al botón de cierre del juego 3. El jugador pulsa el botón de cierre del juego 4. El juego se cierra |
| Postcondiciones | Haber parado la ejecución del programa |
| Excepciones | No las hay |
| Frecuencia | Moderada |
| Importancia | Alta |

Caso de uso 05

| | |
|-------------------------------|--|
| CU-05 | Viajar al menú de opciones |
| Descripción | Permitir al usuario viajar al menú de opciones desde el menú principal |
| Requisitos funcionales | RF-1, RF-2, RF-2.3 |
| Precondición | Estar en el menú principal |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. Se muestra al jugador todos los botones con los que puede interactuar 2. El usuario se desplaza al botón del menú de opciones 3. El jugador pulsa el botón del menú de opciones 4. Inicializar la escena del menú de opciones 5. Cambiar a la escena del menú de opciones |
| Postcondiciones | Estar en la escena del menú de opciones |
| Excepciones | No las hay |
| Frecuencia | Moderada-Baja |
| Importancia | Alta |

Caso de uso 06

| | |
|-------------------------------|---|
| CU-06 | Modificar opción |
| Descripción | Permitir al usuario modificar características generales del juego |
| Requisitos funcionales | RF-3 |
| Precondición | Estar en el menú de opciones o el menú de pausa |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. Se muestra al jugador todas las opciones que modificar 2. Seleccionar la opción que se desea modificar 3. Modificar el valor asociado a esa opción 4. Hacer el cambio persistente |
| Postcondiciones | Haber modificado esa característica general y como afecta al juego |
| Excepciones | A la hora de mostrar las opciones, si el jugador no le ha dado un valor concreto previamente, se le asignará a la característica general un valor por defecto |
| Frecuencia | Moderada-Baja |
| Importancia | Alta |

Caso de uso 07

| | |
|-------------------------------|---|
| CU-07 | Gestión de niveles |
| Descripción | El usuario manejará con un avatar en un nivel seleccionado hasta salir de él interactuando con los elementos que contiene |
| Requisitos funcionales | RF-4, RF-4.1, RF-4.2, RF-4.3, RF-4.4, RF-4.5, RF-4.6, RF-4.7, RF-4.8, RF-4.9, RF-5 |
| Precondición | Haber inicializado correctamente el nivel |

| | |
|---------------------------|--|
| Secuencia de pasos | <ol style="list-style-type: none"> 1. Se inicializan los objetos de la escena 2. Se muestra el avatar jugable y su posición en el nivel 3. Se le da el control del avatar jugable al jugador 4. Se le da la opción al jugador de interactuar con distintos elementos 5. El avatar jugable podrá morir 6. Se puede abrir el menú de pausa 7. Llegar a la zona de victoria 8. Volver al menú principal |
| Postcondiciones | Haber vuelto al menú principal |
| Excepciones | Se puede volver al menú de pausa sin haber pasado por las operaciones 4, 5, 6, 7 y 8 |
| Frecuencia | Muy Alta |
| Importancia | Muy Alta |

Caso de uso 08

| | |
|-------------------------------|---|
| CU-08 | Gestión del menú de pausa |
| Descripción | El usuario podrá abrir un menú de pausa similar al de opciones desde el nivel |
| Requisitos funcionales | RF-4.4, RF-4.4.1, RF-4.4.2, RF-4.4.3 |
| Precondición | Estar en un nivel |

| | |
|---------------------------|--|
| Secuencia de pasos | <ol style="list-style-type: none"> 1. Pulsar el botón de apertura del menú de pausa 2. Pausar la ejecución del juego 3. Abrir el menú del nivel 4. El usuario podrá interactuar con menú de opciones 5. El usuario podrá volver al menú principal desde el menú de opciones 6. Pulsar el botón de cierre del menú de pausa 7. Cerrar el menú de pausa 8. Reanudar la ejecución del nivel |
| Postcondiciones | Haber vuelto al nivel, que se estará ejecutando con normalidad |
| Excepciones | En caso de pulsar el botón de volver al menú principal se volverá a este y no al nivel |
| Frecuencia | Moderada-Alta |
| Importancia | Alta |

Caso de uso 09

| | |
|-------------------------------|---|
| CU-09 | Manipulación de la gravedad |
| Descripción | Los objetos manipuladores de gravedad interactuarán con el avatar jugable |
| Requisitos funcionales | RF-4.5 |
| Precondición | Estar en un nivel con objetos manipuladores de gravedad |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. El avatar jugable entra en contacto con un objeto manipulador de gravedad 2. El objeto manipulador de gravedad modifica como afecta la gravedad al avatar jugable 3. El avatar jugable deja de estar en contacto con el objeto manipulador de gravedad |
| Postcondiciones | No las hay |

| | |
|--------------------|---|
| Excepciones | Es posible que el objeto manipulador de gravedad deje de modificar la gravedad cuando el avatar |
| Frecuencia | Moderada-Baja |
| Importancia | Alta |

Caso de uso 10

| | |
|-------------------------------|--|
| CU-10 | Modificación del tiempo a nivel global |
| Descripción | Los objetos modificadores del tiempo podrán modificar la escala de tiempo que afecta a los objetos a nivel global jugable |
| Requisitos funcionales | RF-4.6 |
| Precondición | Estar en un nivel con un avatar jugable |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. El usuario pulsará el botón de escalado de tiempo 2. La escala de tiempo global será modificada mientras dure el efecto 3. El efecto de dejará de aplicarse 4. La escala de tiempo global volverá al estado en el que se encontraba antes de aplicar el efecto 5. El usuario deberá esperar un tiempo antes de volver a activar este efecto |
| Postcondiciones | El escala de tiempo tiene que ser la misma que antes de aplicar el efecto |
| Excepciones | <p>Si el usuario intenta activar este efecto durante el tiempo que esta deshabilitado el efecto no se aplicará</p> <p>Si el usuario intenta activar este efecto mientras ya se esta llevando a cabo el efecto no se aplicará</p> |
| Frecuencia | Moderada-Baja |
| Importancia | Alta |

Caso de uso 11

| | |
|--------------|--|
| CU-11 | Modificación del tiempo a nivel particular |
|--------------|--|

| | |
|-------------------------------|---|
| Descripción | Los objetos modificadores del tiempo podrán modificar la escala de tiempo que afecta a un objeto en particular |
| Requisitos funcionales | RF-4.6 |
| Precondición | Estar en un nivel con modificadores de tiempo a nivel particular Estar en un nivel con objetos a los que les afecte el tiempo a nivel particular |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. Un objeto al que le afecte el tiempo entra en contacto con el objeto modificador de tiempo a nivel particular 2. Se modifica la escala de tiempo del objeto al que le afecta el tiempo 3. Se cancela la modificación sobre escala de tiempo que efectúa el modificador de tiempo a nivel particular 4. La escala de tiempo provocada por el modificador de tiempo a nivel particular se tiene que haber cancelado |
| Postcondiciones | La escala de tiempo provocada por el modificador de tiempo a nivel particular se tiene que haber cancelado |
| Excepciones | Si un objeto al que le afecte el tiempo esta en contacto con varios modificadores de tiempo particulares se aplicarán todas las modificaciones superponiéndose entre ellas |
| Frecuencia | Moderada-Baja |
| Importancia | Alta |

Caso de uso 12

| | |
|-------------------------------|---|
| CU-12 | Aplicación de impulsos |
| Descripción | Los objetos aplicadores de impulsos aplicarán impulsos sobre el avatar jugable |
| Requisitos funcionales | RF-4.7 |
| Precondición | Estar en un nivel con objetos aplicadores de impulso Estar en un nivel con un avatar jugable |

| | |
|---------------------------|--|
| Secuencia de pasos | <ol style="list-style-type: none"> 1. El avatar jugable entra en contacto con el objeto aplicador de impulso 2. El objeto aplicador de impulso aplica un impulso sobre el avatar jugable |
| Postcondiciones | La velocidad que lleva el avatar debe de haber variado |
| Excepciones | Un modificador de impulso que aplique un impulso que sea un multiplicador de impulso que lleva un avatar jugable quieto no variará la velocidad de este |
| Frecuencia | Moderada-Baja |
| Importancia | Alta |

Caso de uso 13

| | |
|-------------------------------|---|
| CU-13 | Colisión con obstáculos |
| Descripción | El avatar podrá colisionar con obstáculos |
| Requisitos funcionales | RF-4.8 |
| Precondición | Estar en un nivel con obstáculos Estar en un nivel con un avatar jugable |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. El avatar jugable entra en contacto con el obstáculo 2. El avatar jugable muere 3. El avatar jugable reaparece |
| Postcondiciones | El avatar se encuentra en la posición de inicio del nivel |
| Excepciones | No las hay |
| Frecuencia | Moderada-Baja |
| Importancia | Moderada-Alta |

Caso de uso 14

| | |
|-------------------------------|--|
| CU-14 | Colisión con portales |
| Descripción | El avatar jugable se teletransportará al colisionar con un portal |
| Requisitos funcionales | RF-4.9 |
| Precondición | Estar en un nivel con al menos 2 portales Estar en un nivel con un avatar jugable |

| | |
|---------------------------|---|
| Secuencia de pasos | <ol style="list-style-type: none"> 1. El avatar jugable entra en contacto con un portal 2. El avatar jugable cambiará su posición con la del portal pareja de aquel con el que ha colisionado |
| Postcondiciones | La posición del avatar será la misma que la del portal pareja de aquel con el que se ha colisionado |
| Excepciones | Si el portal no tiene un portal parejo el avatar jugable no cambiará su posición cuando colisione con él |
| Frecuencia | Moderada-Baja |
| Importancia | Moderada-Alta |

Caso de uso 15

| | |
|-------------------------------|--|
| CU-15 | Movimiento del avatar |
| Descripción | Permite usuario ordenarle al avatar jugable realizar un movimiento horizontal |
| Requisitos funcionales | RF-5 |
| Precondición | Estar en un nivel con un avatar jugable |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. El usuario pulsa el botón de desplazamiento del avatar jugable hacia la derecha 2. El avatar jugable se desplaza hacia la derecha 3. El usuario pulsa el botón de desplazamiento del avatar jugable hacia la izquierda 4. El avatar jugable se desplaza hacia la izquierda |
| Postcondiciones | La velocidad del avatar jugable ha variado |
| Excepciones | Si el avatar jugable esta en contacto con una pared no podrá moverse en esa dirección |
| Frecuencia | Muy Alta |
| Importancia | Muy Alta |

Caso de uso 16

| | |
|-------------------------------|---|
| CU-16 | Salto del avatar |
| Descripción | Permite usuario ordenarle al avatar jugable realizar un salto |
| Requisitos funcionales | RF-5 |
| Precondición | Estar en un nivel con un avatar jugable |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. El usuario pulsa el botón de salto del avatar jugable 2. El avatar jugable realiza el salto |
| Postcondiciones | El avatar se encontrará en el aire |
| Excepciones | Si el avatar jugable se encuentra en el aire no realizará el salto |
| Frecuencia | Muy Alta |
| Importancia | Muy Alta |

Caso de uso 17

| | |
|-------------------------------|--|
| CU-17 | Acelerón del avatar |
| Descripción | Permite usuario ordenarle al avatar jugable realizar un acelerón |
| Requisitos funcionales | RF-5 |
| Precondición | Estar en un nivel con un avatar jugable |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. El usuario pulsa el botón de acelerón del avatar jugable 2. El avatar jugable empieza a realizar el acelerón 3. El avatar jugable se desplaza en una dirección durante un tiempo 4. Al acabar el tiempo el jugador deja de realizar el acelerón 5. Desactivar temporalmente el acelerón |
| Postcondiciones | El avatar ha variado su posición |
| Excepciones | Si el usuario intenta utilizar el acelerón mientras esta desactivado no podrá utilizarlo |
| Frecuencia | Moderada-Alta |
| Importancia | Alta |

Caso de uso 18

| | |
|-------------------------------|---|
| CU-18 | Modificación temporal del avatar |
| Descripción | Permite al usuario ordenarle al avatar jugable realizar modificar el tiempo global del nivel |
| Requisitos funcionales | RF-5 |
| Precondición | Estar en un nivel con un avatar jugable |
| Secuencia de pasos | <ol style="list-style-type: none"> 1. El usuario pulsa el botón de modificación temporal del avatar jugable 2. La escala de tiempo global cambia 3. Pasado un determinado tiempo la escala de tiempo global vuelve a su valor inicial 4. Se desactiva la modificación temporal del avatar jugable |
| Postcondiciones | La escala temporal global vuelve a ser la misma que antes de aplicar la modificación temporal global |
| Excepciones | Si el usuario intenta utilizar la modificación temporal mientras esta desactivado no podrá utilizarlo |
| Frecuencia | Moderada |
| Importancia | Alta |

Apéndice C

Especificación de diseño

- C.1. Introducción
- C.2. Diseño de datos
- C.3. Diseño procedimental
- C.4. Diseño arquitectónico

Apéndice D

Documentación técnica de programación

D.1. Introducción

D.2. Estructura de directorios

Se va a comentar la estructura de ficheros del repositorio de GitHub que contiene todos los materiales relativos al proyecto.

- `/`: Directorio raíz del repositorio.
- `/docs`: Directorio que contiene todos los ficheros relativos a la documentación.
- `/docs/Latex`: Carpeta con todos los ficheros necesarios para la generación de la memoria y los anexos con Latex.
- `/docs/Latex/img`: Contiene todas las imágenes que aparecerán en los ficheros generados en Latex.
- `/docs/Latex/tex`: Contiene los ficheros de Latex secundarios que utilizarán los principales para generar la memoria y los anexos.
- `/game`: Contiene la solución de Visual Studio con el código relativo al videojuego desarrollado.

- /game/Assets: Contiene todos los ficheros de código en C# y en formatos especializados de Unity necesarios para que Unity pueda exportar el juego correctamente.
- /game/Assets/Audio: Ficheros de audio del proyecto.
- /game/Assets/Character: Elementos estéticos exclusivos del Player.
- /game/Assets/Character/Animations: Animaciones que provee Platformer Microgame utilizadas en el Player.
- /game/Assets/Character/Sprites: Sprites del Player.
- /game/Assets/Environment: Sprites utilizados para crear los mapas de los niveles jugables.
- /game/Assets/ModAssets: Elementos estéticos extra que provee Platformer Microgame para ofrecer más variedad estética.
- /game/Assets/Prefabs: Contiene todos los prefabs del proyecto.
- /game/Assets/Scenes: Contiene todas las escenas que conforman el proyecto.
- /game/Assets/Scripts: Ficheros de C# utilizados en el proyecto.
- /game/Assets/Scripts/Animation: Ficheros de C# utilizados en el proyecto relativos a la animación de sprites.
- /game/Assets/Scripts/Core: Ficheros de C# utilizados en el proyecto. Contiene las clases encargadas del funcionamiento básico de los niveles.
- /game/Assets/Scripts/Gameplay: Ficheros de C# utilizados en el proyecto. Contiene las clases que son eventos de Simulation.
- /game/Assets/Scripts/GizmosUI: Ficheros de C# utilizados en el proyecto. Contiene las clases utilizadas para mostrar los Gizmos en el editor.
- /game/Assets/Scripts/Mechanics: Ficheros de C# utilizados en el proyecto. Contiene las clases que implementa las mecánicas del juego.
- /game/Assets/Scripts/Model: Ficheros de C# utilizados en el proyecto. Contiene una clase con las variables que se consultarán durante los niveles.

- /game/Assets/Scripts/Sound: Ficheros de C# utilizados en el proyecto. Contiene las clases encargadas de gestionar el audio.
- /game/Assets/Scripts/UI: Ficheros de C# utilizados en el proyecto. Contiene todas las clases relativas a la interfaz de usuario.
- /game/Assets/Scripts/View: Ficheros de C# utilizados en el proyecto. Contiene clases que ofrecen comportamientos que afectan de forma especial a como debería verse un objeto.
- /game/Assets/Sprites: Sprites del proyecto que no pertenecen a Platformer Microgame.
- /game/Assets/Scripts: Elementos usados para generar mapas mediante la herramienta Tilemap de Unity.

D.3. Manual del programador

D.4. Compilación, instalación y ejecución del proyecto

D.5. Pruebas del sistema

Apéndice E

Documentación de usuario

- E.1. Introducción
- E.2. Requisitos de usuarios
- E.3. Instalación
- E.4. Manual del usuario

Apéndice F

Documento de diseño del juego

F.1. Introducción

Plataforma:

Ordenador

Versión:

1.0

Jugabilidad y contenido:

El juego consistirá en una serie de niveles independientes que el jugador tendrá que atravesar hasta llegar a la zona de victoria del nivel. Los niveles tendrán una serie de elementos y mecánicas con los que el jugador habrá de interactuar para llegar a la zona de victoria, evitando a distintos obstáculos que pueden truncar la labor del jugador de llegar a la meta.

En el juego habrá un menú principal que permitirá acceder a todos los niveles jugables. Al alcanzar la meta del nivel se volverá al menú principal.

Debido a que la historia no es competencia del TFG, el juego no tendrá historia.

Categoría:

Plataformas 2D

Mecánica:

El jugador manejará un avatar virtual en una serie de niveles de plataformas en los que tratará de llegar a la zona de victoria del nivel utilizando distintas mecánicas de “viajes en el tiempo” y manipulación gravitatoria.

El jugador podrá realizar las siguientes acciones:

- Moverse.
- Saltar.
- Realizar un acelerón en una dirección.

Las distintas mecánicas con las que podrá interactuar el jugador son:

- Un portal que teletransportará al jugador de un punto a otro del nivel.
- Un creador de impulso que empujará al jugador en una dirección determinada. Esta mecánica se puede manifestar de distintas formas, como una plataforma de salto o un elemento del nivel que escala la velocidad que lleva el jugador.
- Tiempo bala, que ralentizará o acelerará el tiempo de uno o más elementos del nivel, haciendo que se muevan más rápido o más lento.
- Modificadores de la gravedad que influirán en cómo la gravedad afectará al elemento del nivel al que afectará.

Tecnología:

El juego se desarrollará en Unity, utilizando el lenguaje de programación C#.

Visión general del juego:

El juego consistirá en un plataformas 2D de viajes en el tiempo en el que el jugador atravesará una serie de niveles en los que el jugador hará uso de varias mecánicas que influirán en el espacio, el tiempo y la gravedad para esquivar los obstáculos que se interpongan en el camino del jugador hasta llegar la zona de victoria.

F.2. Mecánicas del juego

Jugador

El jugador controlará un avatar que es capaz de moverse hacia la derecha e izquierda. También puede saltar. El salto es uniforme, se saltará cada vez que se pulsa el botón de saltar (espacio en teclado y botón X en el mando) y saltará la misma distancia siempre (la fuerza del salto no variará en función de cuánto tiempo se mantenga pulsado el botón de salto).

Si el jugador colisionase con un obstáculo que le haga daño, el jugador morirá y reaparecerá en la zona inicial del nivel.

El jugador además tendrá la habilidad de dar un acelerón hacia la izquierda o la derecha. Este acelerón se realizará en una dirección horizontal, sin variar la posición vertical. Si el acelerón te desplazase X unidades hacia la derecha y se estuviese en el punto $(1,1)$, el jugador acabará el acelerón en la posición $(1, 1+X)$. El acelerón lo podrás hacer una sola vez mientras estés en el aire, y en cuanto toques el suelo podrás volver a utilizarlo. En el suelo podrás hacerlo ilimitadamente. Parte de la gracia de este acelerón, es que después de terminarlo se mantendrá la velocidad que se llevaba durante el acelerón, ahora sí, variando esta según las leyes que rigen el videojuego.

La última herramienta con la que contará el jugador es el tiempo bala. Esta mecánica permitirá reducir la velocidad a la que pasa el tiempo en el nivel permitiendo al jugador actuar con la precisión que ofrece que todo el nivel se reproduzca menor velocidad.

Enemigos

En principio no habrá enemigos vivos como tales. Sin embargo, sí que habrá obstáculos en el juego que, al tocarlos, mataran al jugador. Estos obstáculos podrán ser estáticos o móviles. Los enemigos estáticos no se mueven y la única dificultad radica en evitar colisionar con él. Los obstáculos móviles, sin embargo, podrán ser de dos tipos.

El primer tipo será un obstáculo que aparecerá en un extremo de la pantalla y se dirigirá al otro en línea recta. Son obstáculos fáciles de esquivar y predecibles. La dificultad de estos obstáculos radica en lo complejo que puede resultar enfrentarse a varios de ellos y la complejidad improvisada que puede surgir al añadir el obstáculo móvil a un escenario complejo ya de por sí.

El segundo tipo de obstáculo móvil será un obstáculo que se limite a una zona del nivel, pero que siga una rutina de movimiento en esa zona. El riesgo de este obstáculo se limita a una zona reducida del nivel, pero dentro de esta zona el jugador correrá serio riesgo de ser asesinado por el obstáculo.

Portal

Los portales son dos elementos enlazados que parten del hecho de que si se entra por un portal la posición del jugador (en ese momento la del portal que se ha atravesado) se convertirá en la posición del portal pareja del portal por el que se ha entrado. Esta mecánica es sencilla y para nada innovadora, pero que funciona muy bien. Un portal teletransportador es una idea sencilla de utilizar y entender por el jugador.

La gracia de los portales está en salir del segundo portal. Cuando se salga del segundo portal, el jugador mantendrá la velocidad con la que entró por el primer portal. Esto convierte al hecho de salir por un portal en una mecánica muy variada (y con posibilidades potencialmente infinitas) y que puede dar lugar a puzles interesantes.

Creador de impulso

Esta mecánica consistirá en influir sobre la dirección de la velocidad que lleva un elemento afectado por físicas. Esta mecánica se puede manifestar de distintas formas:

- Partícula de impulso: Consistirá en un elemento estático en el mapa. Cuando el elemento afectado por las físicas entra en contacto con esta partícula de impulso, este saldrá disparado en una dirección predefinida. La dirección en la que saldrá disparado el elemento afectado por las físicas será siempre la misma. Eso sí, la dirección es individual para cada partícula, siendo que la partícula siempre disparará el elemento afectado por físicas en una dirección concreta, pero que no tendrá por qué ser la misma para dos partículas distintas.
- Amplificador de impulso: Consistirá en un elemento estático, al igual que la partícula, pero con un funcionamiento ligeramente distinto. Cuando el elemento afectado por físicas entra en contacto con el amplificador, la velocidad que lleva el elemento afectado por físicas se verá escalada. Dos ejemplos de amplificador serían: amplificador positivo de impulso ($\text{velocidad} = \text{velocidad} * X$) y amplificador negativo de impulso ($\text{velocidad} = \text{velocidad}/X$). Los amplificadores tendrán

cada uno un valor particular de escalado del impulso que no tiene por qué coincidir con el resto de amplificadores de impulso.

- **Plataforma de salto:** La típica plataforma de salto que propulsará al jugador cuando entra en contacto con esta. Puede no parecer un elemento creador de impulso, y puede que no lo sea, pero como su funcionamiento será igual al de un creador de impulso, se va a considerar un creador de impulso.

Modificadores de gravedad

Elementos que van a afectar en cómo influye la gravedad sobre uno o varios elementos afectados por físicas. Hay dos tipos de modificadores de gravedad en el juego:

Inversor de gravedad: Elemento estático que provocará que la gravedad se invierta. Esta inversión es “absoluta” en el sentido de que invertir una gravedad de $-9,81$ hace que esta se convierta en $9,81$. Pero una gravedad de -8 invertida la convierte en 8 y no en $9,81$.

Obstáculos superdensos: Estos son un tipo especial de enemigo, que, a su alrededor, generarán un capo gravitatorio que empujará a los elementos afectados por físicas hacia ellos. Estos obstáculos tendrán que luchar por la gravedad que afecta a los elementos afectados por físicas. Como ejemplo, si un elemento afectado por físicas se viese afectado por una gravedad de $(-9,81, 0)$ y un obstáculo con una gravedad $(5, 0)$ el elemento afectado por físicas, en caso de estar debajo del obstáculo verá su gravedad será convertida a $(-4,81, 0)$, pero si el elemento afectado por físicas esta encima del obstáculo su gravedad será convertida a $(-13,81, 0)$.

Esta es una explicación rápida para que se entienda, pero también afectará al elemento afectado por físicas la distancia a la que se encuentre del obstáculo.

En caso de alcanzar el centro de este obstáculo el elemento afectado por físicas morirá si tiene vida, si no se quedará atrapado en el centro.

Tiempo bala

Esta mecánica consistirá en manipular como el tiempo afecta a uno o varios elementos afectados por físicas. Esta mecánica es la más costosa de implementar tanto en tiempo como en esfuerzo. Es por ello que no se va a ser demasiado preciso al respecto, pero en principio, esta mecánica se manifestará de dos formas distintas.

Escalar el tiempo: El jugador podrá pulsar el botón de “Tiempo bala” y reducir el tiempo y como este afectará al entorno. Esta reducción de tiempo será una escala de $1/X$ veces la influencia que tiene el tiempo sobre los elementos afectados por físicas. Esta mecánica afectará a todos los elementos por igual y la escala será siempre la misma para todo el juego y todos los niveles.

Zonas de tiempo escalado: Son zonas en el nivel que escalarán el tiempo de todos los elementos que entren dentro de su área de influencia. La forma en la que escalarán el tiempo puede ser positiva (tiempo = tiempo * X) o negativa (tiempo = tiempo/X).

Las zonas de tiempo escalado serán estáticas en el mapa y la magnitud en la que escalan el tiempo es particular para cada zona, pudiendo ser distinta del resto de zonas de tiempo escalado.

F.3. Niveles

Habrán tres tipos distintos de niveles:

Niveles de prueba

Estos niveles serán inaccesibles para el jugador. Son niveles utilizados por el desarrollador para comprobar el correcto funcionamiento del juego e incluso forzar algunos errores intencionalmente. El uso de estos niveles será exclusivo para el desarrollo del proyecto.

Niveles tutorial

Niveles básicos utilizados para introducir mecánicas al jugador y ayudarles a comprender los conceptos que se le explicarán en un entorno controlado. Estos niveles podrán coincidir con algún nivel de prueba que resulte conveniente tanto para comprobar el funcionamiento básico de una mecánica como para explicar el funcionamiento de la mecánica.

Como ejemplo se va a poner el nivel de prueba de la mecánica de los portales:

PruebaPlayerScene

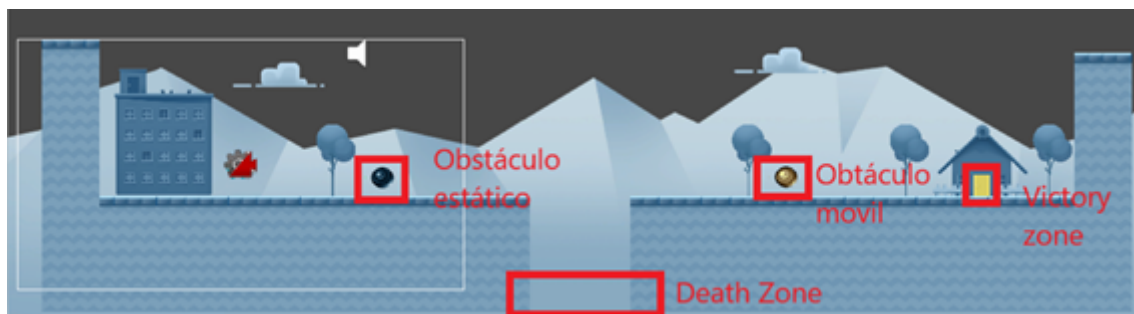


Figura F.2: Pantalla de prueba de mecánicas básicas

PruebaPortalScene

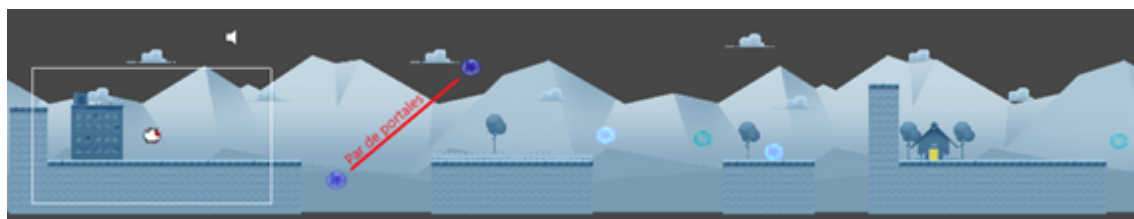


Figura F.3: Pantalla de prueba de los portales

PruebaMovingObstacleScene



Figura F.4: Pantalla de prueba de los obstáculos móviles

PruebaImpulseCreatorsScene



Figura F.5: Pantalla de prueba de los creadores de impulso

PruebaGravityModifierScene

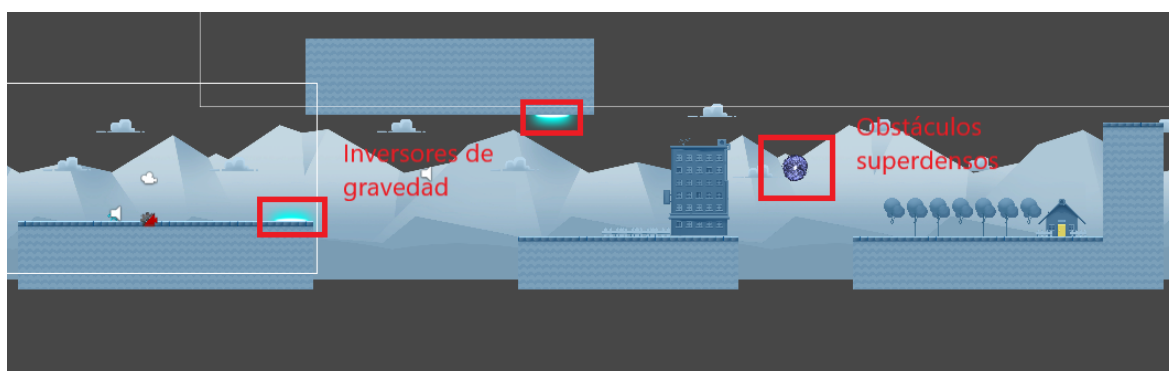


Figura F.6: Pantalla de prueba de los modificadores de gravedad

PruebaTimeModifierScene



Figura F.7: Pantalla de prueba de los modificadores temporales

F.4. Interfaces

El patrón de pantalla que más se va a repetir será el nivel sin ningún elemento extradiegético que afecte a la pantalla del nivel. No se incluirán elementos HUD (Head-Up Display) que monitoricen la vida ni otros elementos, prácticamente por su ausencia (siendo el estado de los elementos variables, si se podrá saltar o si se estará el acelerón disponible para su uso, muy sencillos de mantener en mente y notificar con sonidos o animaciones).

Habrà, aun así dos interfaces extra disponibles para el jugador. Al no tener las interfaces todavía desarrolladas se va a añadir a continuación un esquema de cómo serán estas interfaces.

Pantalla de elección de nivel



Figura F.8: Idea inicial de la pantalla de selección de nivel

El sonido de selección de los botones ha sido obtenido del enlace: <https://opengameart.org/content/botton-sound-pack>
Puedes encontrar al creador en twitter como @listener4me. https://twitter.com/search?q=%40listener4me&src=typed_query

El sonido de desplazamiento entre botones ha sido obtenido en el siguiente enlace: <https://opengameart.org/content/menu-selection-click>

Pantalla de opciones



Figura F.9: Idea inicial de la pantalla de opciones

Los sonidos de cierre del menú de pausa y de la barra de los menús de opciones pertenecen a Jesús Lastra han sido obtenidos en el siguiente enlace: <https://opengameart.org/content/button-clicks-beeps-99-sounds>

La música que suena mientras se esta en estas escenas es 01 B-arb del album B-arb, pertenece a Axel Bermudez y se puede encontrar en el siguiente enlace (junto con el resto de canciones del album del que se ha sacado la canción): <https://axelbermudez.bandcamp.com/releases>

Estas pantallas son orientativas y susceptibles de cambios. De la misma forma no se tiene que seguir ciegamente el patrón de colores pero se ha de entender la existencia de un patrón de colores y como este se va a aplicar.

F.5. Entidades

A continuación se incluirá información más concreta al respecto del diseño de juego como sprites e imágenes que identificarán a una mecánica o

elemento del juego en los niveles, configuración de los controles y descripción una por una de cada nivel del juego.

Elementos

Player

Objeto que controlará el jugador. El jugador tendrá la capacidad de hacer que el Player se mueva, salte y pegue acelerones.

El Player se podrá morir en caso de que colisione con un objeto que reduzca su vida a 0. El Player tendrá solo 1 punto de vida y cuando colisione con un objeto que lo “dañe”, su vida se reducirá a 0 y morirá. Cuando el Player muera reaparecerá en un punto de reaparición establecido en la escena.



Figura F.10: Sprite utilizado para el Player

Los sprites, las animaciones y los sonidos del Player los ofrecía Platformer Microgame para uso libre.

La única excepción a lo anteriormente mentado es el acelerón, cuyo audio ha sido obtenido en el siguiente enlace:

<https://opengameart.org/content/sci-fi-shwop-1>

Enemigos

- Obstáculos: Enemigos que no se moverán. Se encontrarán estáticos en un lugar y si el Player los tocara este se muere.

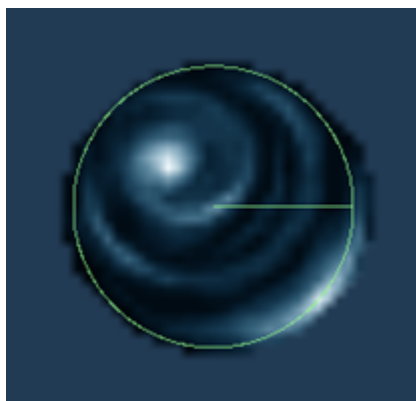


Figura F.11: Sprite utilizado para el obstáculo estático

El sprite utilizado para los obstáculos pertenece a Daniel Cook y ha sido obtenida en el siguiente enlace:

<https://opengameart.org/content/iron-plague-teleportbmp>.

- Obstáculos que seguirán una rutina: Enemigos que se encontrarán en una zona y recorrerán un camino cíclico continuamente.

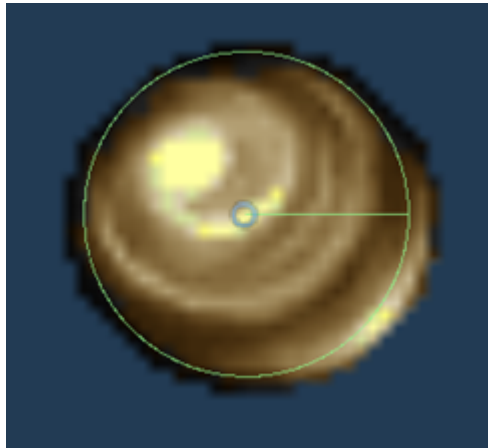


Figura F.12: Sprite utilizado para el obstáculo que sigue una rutina

El sprite utilizado para los obstáculos pertenece a Daniel Cook y ha sido obtenida en el siguiente enlace:

<https://opengameart.org/content/iron-plague-teleportbmp>.

- Obstáculos móviles: Enemigos que aparecerán en un extremo de la pantalla y la atravesará hasta el otro extremo. Si en algún momento el obstáculo móvil tocara al Player, este morirá.



Figura F.13: Obstáculo móvil rápido



Figura F.14: Obstáculo móvil (velocidad intermedia)



Figura F.15: Obstáculo móvil lento

El sprite utilizado para los obstáculos móviles pertenece a Rawdanitsu y ha sido obtenida en el siguiente enlace:

<https://opengameart.org/content/lasers-and-beams>.

Escenarios

- Suelo
- Pared
- Zonas de muerte
- Zonas de victoria

Mecánicas

Portal

Portal que teletransportará a los objetos que entren manteniendo la dirección del movimiento del objeto.

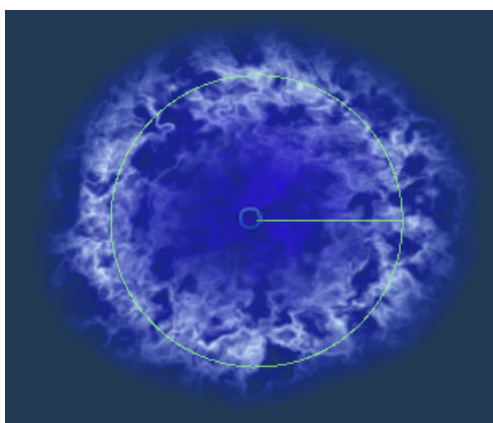


Figura F.16: Sprite utilizado para los portales

El sprite utilizado para los portales pertenece a Hansjörg Malthaner y ha sido obtenida en el siguiente enlace:

<https://opengameart.org/content/animated-portal-or-wormhole-several-variants>.

Enlace al trabajo de Hansjörg Malthaner :

<http://opengameart.org/users/varkalandar>

El audio utilizado para los portales pertenece a dklon y ha sido obtenida en el siguiente enlace: <https://opengameart.org/content/quick-zap>.

Enlace al trabajo de dklon : <https://opengameart.org/users/dklon>

Creador de impulso

El objeto afectado recibirá un impulso en una dirección.

- Partícula de impulso: objeto que se encontrará en el escenario. Cuando un objeto toque esa partícula recibirá un impulso en la dirección asignada a la partícula.



Figura F.17: Sprite utilizado para la partícula de impulso

Los sprites utilizados para la partícula de impulso pertenecen a oglsdl y ha sido obtenida en el siguiente enlace:

<https://opengameart.org/content/glow-arrow>.

El audio utilizado para la partícula de impulso de salto pertenece a Bart Kelsey y ha sido obtenida en el siguiente enlace:

<https://opengameart.org/content/spell-2>

- Amplificador de impulso: parecido a la partícula de impulso pero escalando el impulso que lleva el objeto.



Figura F.18: Sprite utilizado para el amplificador de impulso

Los sprites utilizados para el amplificador de impulso son una modificación de un Sprite que pertenece a oglsdl que ha sido obtenida en el siguiente enlace: <https://opengameart.org/content/glow-arrow>.

El audio utilizado para la partícula de impulso de salto pertenece a Bart Kelsey y ha sido obtenida en el siguiente enlace:

<https://opengameart.org/content/spell-2>

- Plataforma de salto: Se define por si sola.



Figura F.19: Sprite utilizado para el amplificador de impulso

Los sprites utilizados para la plataforma de salto pertenecen a diana23570 y ha sido obtenida en el siguiente enlace:

<https://opengameart.org/content/spring>.

El audio utilizado para la plataforma de salto pertenece a Blender Foundation y ha sido obtenida en el siguiente enlace:

<https://opengameart.org/content/sprint-jumpinteraction-sound-yo-frankie>

Tiempo bala

Capacidad de “ralentizar o acelerar el tiempo” influyendo en la velocidad a la que afectará al uno o varios objetos que se vean afectados.

- Escalar el tiempo: Capacidad del Player para escalar el tiempo que afectará a todos los objeto.
- Zonas de tiempo escalado: Zonas que escalarán el tiempo de los objetos que entren en la zona de influencia de la zona.

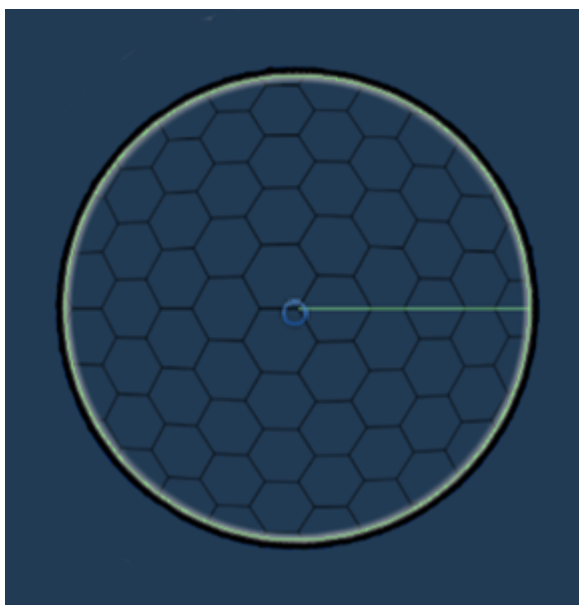


Figura F.20: Sprite utilizado para la zona de tiempo escalado

El sprite utilizado para las zonas de tiempo escalado es una modificación de un srote que pertenece a scenna y ha sido obtenida en el siguiente enlace: <https://opengameart.org/content/circle>.

Acelerón

El jugador realizará un acelerón en la dirección en la que esté mirando.

Modificadores de gravedad

Objetos que afectarán a como la gravedad influye sobre ellos u otros objetos.

- Inversor de gravedad: volverá la gravedad negativa en positiva y viceversa.

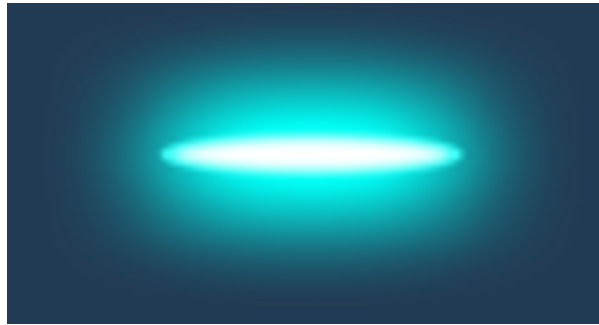


Figura F.21: Sprite utilizado para el inversor de gravedad

El sprite utilizado para los inversores de gravedad pertenece a Rawdanitsu y ha sido obtenida en el siguiente enlace:

<https://opengameart.org/content/lasers-and-beams>.

- Obstáculos superdensos: Obstáculos hacia los que atraerán a los objetos debido a su alta gravedad.

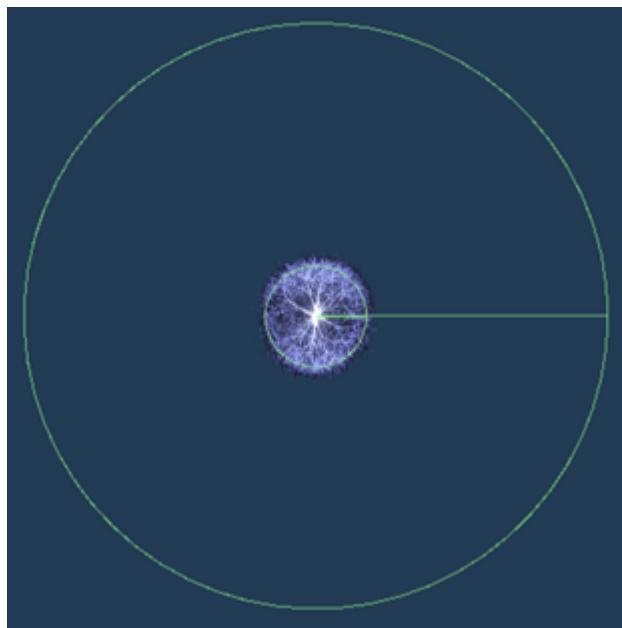


Figura F.22: Sprite utilizado para el obstáculo superdenso

Los sprites utilizados para la plataforma de salto pertenecen a Hansjörg Malthaner y ha sido obtenida en el siguiente enlace:

<https://opengameart.org/content/animated-charged-bolt>.

Enlace al trabajo de Hansjörg Malthaner:

<http://opengameart.org/users/varkalandar>

F.6. Controles

Mando

Los controles del mando se registrarán por la nomenclatura de los controles de PlayStation.



Figura F.23: Nomenclatura de los mandos de PlayStation

- **Movimiento:** Stick izquierdo.
- **Salto:** botón X (Joystick button 1 en Unity)
- **Acelerón:** botón O (Joystick button 2 en Unity)
- **Tiempo bala:** botón R1 (Joystick button 5 en Unity)
- **Abrir el menú de pausa:** botón R2(Joystick button 7 en Unity)

Teclado

- **Movimiento:** flecha izquierda para ir a la izquierda y flecha derecha para ir a la derecha.
- **Salto:** espacio
- **Acelerón:** tecla S

- **Tiempo bala:** tecla D
- **Abrir el menú de pausa:** tecla ESC

Bibliografía

- [1] Ministerio de cultura de España. Ley de propiedad intelectual, 1996. [Internet; Ultimo acceso: 6-junio-2021].
- [2] Sergio C. González. Unidades vendidas por los juegos indies, 2019. [Internet; Ultimo acceso: 6-junio-2021].
- [3] Jobted. Sueldo de un programador de videojuegos, 2021. [Internet; Ultimo acceso: 6-junio-2021].
- [4] Kanban tool. Tablero kanban, 2021. [Internet; Ultimo acceso: 6-junio-2021].
- [5] Unity. Faq unity, 2021. [Internet; Ultimo acceso: 6-junio-2021].
- [6] Wikipedia. Metodología kanban, 2020. [Internet; Ultimo acceso: 6-junio-2021].