



ugr

Universidad
de **Granada**

TRABAJO FIN DE GRADO
INGENIERÍA EN INFORMÁTICA

Desarrollo de un videojuego

Videojuego basado en historia del arte

Autores

Jesús García Godoy
Samuel Carmona Soria

Director

Carlos Cruz Corona

Colaboradores

Ximena Hidalgo



DECSAI



Escuela Técnica Superior de Ingenierías Informática y de
Telecomunicación

—
Granada, 11 de septiembre de 2016



ugr | Universidad
de **Granada**

DEGREE'S FINAL PROJECT
COMPUTER ENGINEERING

Video game development

Art history based video game

Authors

Jesús García Godoy
Samuel Carmona Soria

Director

Carlos Cruz Corona

Collaborators

Ximena Hidalgo



DECSAI



Escuela Técnica Superior de Ingenierías Informática y de
Telecomunicación

—
Granada, 11th September, 2016

Desarrollo de un videojuego: Videojuego de historia del arte

Jesús García Godoy

Samuel Carmona Soria

Resumen

El presente proyecto final de grado trata básicamente del desarrollo de un videojuego desde cero. La temática para tal juego ha sido elegida por la diseñadora colaboradora, Ximena Hidalgo, y tal elección ha sido la historia del arte.

La plataforma elegida para el desarrollo ha sido Unity3D, el cual tiene bastante potencia como motor gráfico además de tener una curva de aprendizaje moderada, dando pie a poder realizar este proyecto desde cero.

El principal objetivo del proyecto es, pues, desarrollar a modo de demo, un videojuego con temática de historia del arte que contenga distintas mecánicas y fases, así como un breve argumento. El objetivo del juego será superar las distintas pantallas superando obstáculos, solventando ciertos enigmas o derrotando enemigos, además de obtener cierta puntuación y además seguir un simple argumento, el cual propicia que con una pequeña historia el juego sea más interesante y no solamente para jugar a secas.

Palabras clave

videojuego historia arte unity3D 2D plataformas
renacentismo xilografía abstracción

Video game development: Art history based video game

Jesús García Godoy

Samuel Carmona Soria

Abstract

This degree final project consists basically on developing a videogame from scratch. The topic for the game has been chosen by the collaborator designer, Ximena Hidalgo, and it has been art history.

The chosen platform for the development has been Unity3D, which has enough power as graphical engine, as well as having a moderately acceptable learning curve, letting us develop this project from scratch.

The main objective of this project is to develop, as a demonstration, a videogame related to the art history, which will contain different mechanics and stages, as well as a short plot. The objective of the game will be to finish the different stages surpassing obstacles, solving some enigmas or defeating certain enemies, apart from obtaining some score and follow a simple plot, which makes more interesting the gameplay for the player and not just to play without any precise goal.

Keywords

videogame history art unity3D 2D platforms renaissance
xylography abstraction

Nosotros, **Jesús García Godoy** y **Samuel Carmona Soria**, alumnos de la titulación GRADO EN INGENIERÍA INFORMÁTICA de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI's 47320367Y y 50624716Z, respectivamente, autorizamos la ubicación de la siguiente copia de nuestro Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.



Fdo: Samuel Carmona Soria

Fdo: Jesús García Godoy

Granada a 11 de septiembre de 2016.

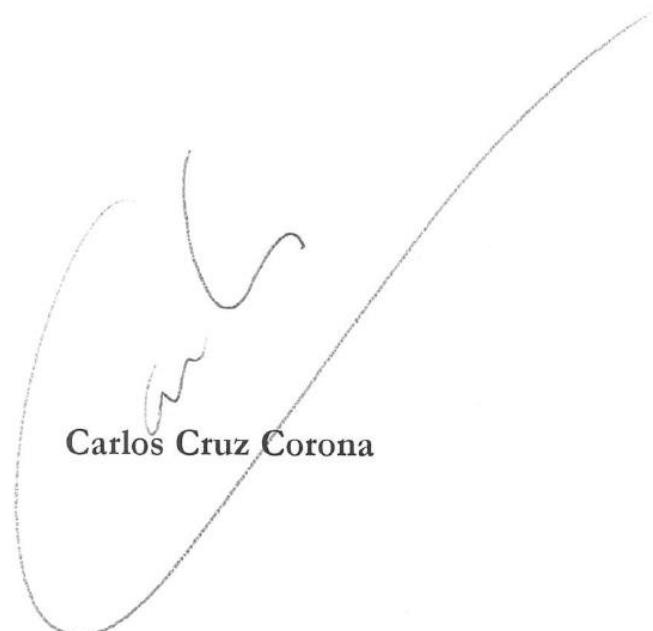
D. Carlos Cruz Corona, Profesor del XXXX del Departamento DECSAI de la Universidad de Granada.

Informa:

Que el presente trabajo, titulado ***Desarrollo de un Videojuego, Videojuego Basado en Historia del Arte***, ha sido realizado bajo su supervisión por **Jesús García Godoy y Samuel Carmona Soria**, y autorizo la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a X de septiembre de 2016 .

El director:



A handwritten signature in black ink, enclosed within a hand-drawn circle. The signature reads "Carlos Cruz Corona".

Agradecimientos

Carlos Cruz Corona - Tutor de proyecto - Idea general

Ximena Hidalgo - Diseñadora gráfica y autora de la idea

Asbjørn Thirslund - CEO, Fundador de Brackeys - Ayuda mediante
preguntas

ÍNDICE

1. Introducción	22
1.1 Objetivo	22
1.2 Argumento.....	22
1.3 Objetivo del juego.....	23
2. Estado del arte.....	24
2.1 Historia de los videojuegos	24
2.1.1 1970-1979: La eclosión de los videojuegos.....	25
2.1.2 1980-1989: La década de los 8 bits	26
2.1.3 1990-1999: La revolución de las 3D	27
2.1.4 Desde el 2000: El comienzo del nuevo siglo	30
2.2 Historia del arte	31
2.2.1 Época clásica: Renacimiento	31
2.2.2 Época vanguardista: Abstracción	34
2.2.3 Época técnica: Grabado Xilográfico	37
2.3 El videojuego como obra de arte	39
2.4 Inspiraciones	43
2.4.1 Apotheon	44
2.4.2 Limbo	45
2.4.3 Nihilumbra.....	46
2.4.4 Rock of Ages.....	47
2.4.5 Ori and the Blind Forest	48
2.4.6 The Witness	49
2.4.7 The Unfinished Swan	50
2.4.8 Psychonauts	51
2.4.9 Super Mario 64	52
2.4.10 Inside Out.....	53
2.4.11 The Legend of Zelda: A Link Between Worlds.....	54
3. Análisis	55
3.1 Análisis de requisitos.....	55
3.1.1 Actores.....	55
3.1.2 Requisitos funcionales	55
3.1.3 Requisitos no funcionales	56
3.1.4 Requisitos de diseño (restricciones).....	57
3.2 Ejemplo de caso de uso	57

4. Herramientas	58
4.1 Requerimientos	58
4.2 Software y Lenguajes de Programación	58
4.2.1 Motor gráfico	58
4.2.2 Lenguajes usados.....	58
4.2.3 IDE.....	58
4.3 Plataformas	58
4.4 Unity	59
4.5 Jerarquía de objetos - Unity.....	65
4.5.1 Jerarquía - Personaje principal.....	65
5. Planificación.....	67
5.1 Fases	67
5.2 Definición y estimación de tiempo para las tareas de cada fase	67
5.3 Diagramas de temporización	69
6. Implementación.....	70
6.1 Flujo de Juego.....	70
Inicio	70
Resultados.....	71
6.2 Mecánicas del juego.....	72
6.2.1 Personaje Principal	72
6.2.2 Enemigo azul	78
6.2.3 Transformación xilográfica - Protagonista	80
6.2.4 Perro de las Meninas	82
6.2.5 Enemigo volador	84
6.2.6 Animaciones - Transiciones	85
6.2.7 Escenarios	88
6.2.8 Fases	90
6.2.9 Sonido.....	117
7. Código	119
7.1 Ejemplos de clases - MainCharacterController	121
7.1.1 Atributos.....	121
7.1.2 Métodos:	123
7.2 Ejemplos de clases - TakeBone	127
7.2.1 Atributos.....	127
7.2.2 Métodos	127
7.3 Ejemplos de clases - DumbEnemy	128

7.3.1 Atributos	128
7.3.2 Métodos	128
8. Diagramas	131
8.1 Diagrama de clases	131
8.2 Diagrama de flujo	132
8.3 Ejemplos de diagramas de secuencia	133
8.3.1 Diagrama de secuencia del seguimiento de la cámara.....	133
8.3.2 Diagrama de secuencia de cómo se asigna la salud al personaje.....	134
8.3.3 Diagrama de secuencia de la función Update (refresca cada frame) del personaje de la fase de abstracción	135
8.3.4 Diagrama de secuencia de la función Update del personaje de la fase del Renacimiento.....	136
9. Conclusiones y trabajo futuro	137
9.1 Conclusiones	137
9.2 Trabajo futuro	137
10. Aspectos legales	137
11. Anexos	138
12. Bibliografía / Referencias	138

1. Introducción

El presente proyecto final de grado trata del desarrollo de un videojuego desde cero. La temática para tal juego ha sido elegida por la diseñadora colaboradora, Ximena Hidalgo, y tal elección ha sido la historia del arte.

La plataforma elegida para el desarrollo ha sido Unity3D, el cual tiene bastante potencia como motor gráfico además de tener una curva de aprendizaje moderada, dando pie a poder realizar este proyecto desde cero.

1.1 Objetivo

El principal objetivo del proyecto, pues, es desarrollar a modo de demo, un videojuego con temática de historia del arte que contenga distintas mecánicas y fases, así como un breve argumento. El objetivo del juego será superar las distintas pantallas superando obstáculos, solventando ciertos enigmas o derrotando enemigos, además de obtener cierta puntuación y además seguir un simple argumento, el cual propicia que con una pequeña historia el juego sea más interesante y no solamente para jugar a secas.

1.2 Argumento

Nuestra historia se sitúa en la actualidad, y cuenta la historia de Dorian, un joven artista con una gran imaginación, y sin embargo, en busca de inspiración para su nueva obra, para lo cual no tiene reparo en hacer alguna que otra travesura.

Nos situamos concretamente en el Museo del Prado, situado en el Palacio Real de Madrid, donde se halla el famoso cuadro de Velázquez: *Las Meninas*.

Dorian, ante la imposibilidad del acceso al museo por cuestiones de alboroto en el pasado, decide colarse una noche, estando el museo cerrado, mediante un conductor de ventilación. Empieza así un recorrido viendo varias de las obras allí expuestas, hasta que, al llegar al cuadro de Velázquez cuando empieza a darse cuenta de que algo extraño está sucediendo: ¡ Los elementos del cuadro parecen estar moviéndose !

Nuestro protagonista no puede creerse lo que está viendo y duda de si su imaginación le está jugando una mala pasada o es un hecho real. No ha salido aún de su asombro cuando se da cuenta de que está solo en la sala y todo ha adquirido un tono algo pintoresco, y de repente...los personajes del cuadro le miran y salen corriendo! Y en un instante no queda nadie en el cuadro, excepto Velázquez y el perro.

Velázquez no está contento con la situación, pues su obra se ha quedado vacía, y se dirige directamente a Dorian, encomendándole la tarea de ir a buscar a los personajes del cuadro, para lo cual contará con la ayuda del perro, quien los rastreará.

Antes de que Dorian pudiese responder, una especie de portal se abrió ante él y éste lo absorbió cual agujero negro. Y así es como empieza la aventura artística e histórica de nuestro joven protagonista, que se verá envuelto en el mundo del arte, saltando de época en época y viviendo aventuras, con el fin de recuperar los personajes perdidos del cuadro de Velázquez.

1.3 Objetivo del juego

El objetivo principal, como se menciona en el argumento, es encontrar, con ayuda del perro, algunos de los personajes del cuadro de Las Meninas, que han desaparecido al notar nuestra presencia en el museo.

A través de distintas fases en las que aparecerán distintos enemigos que nos podrán eliminar, así como pruebas que tendremos que resolver para seguir avanzando.

Será necesario recoger un cierto número de ítems, derrotar a enemigos, y encontrar a los personajes del cuadro perdidos en cada nivel para superar ciertas fases.

Finalmente, si logramos superarlo todo, completaremos el juego y se mostrarán los créditos.

2. Estado del arte

2.1 Historia de los videojuegos

Durante bastante tiempo ha sido complicado señalar cual fue el primer videojuego, principalmente debido a las múltiples definiciones de este que se han ido estableciendo, pero se puede considerar como primer videojuego el Nought and crosses, también llamado OXO, desarrollado por Alexander S. Douglas en 1952. El juego era una versión computerizada del tres en raya que se ejecutaba sobre la EDSAC y permitía enfrentar a un jugador humano contra la máquina.

En 1958 William Higinbotham creó, sirviéndose de un programa para el cálculo de trayectorias y un osciloscopio, Tennis for Two (tenis para dos): un simulador de tenis de mesa para entretenimiento de los visitantes de la exposición Brookhaven National Laboratory.

Este videojuego fue el primero en permitir el juego entre dos jugadores humanos. Cuatro años más tarde Steve Russell, un estudiante del Instituto de Tecnología de Massachusetts, dedicó seis meses a crear un juego para computadora usando gráficos vectoriales: Spacewar.

En este juego, dos jugadores controlaban la dirección y la velocidad de dos naves espaciales que luchaban entre ellas. El videojuego funcionaba sobre un PDP-1 y fue el primero en tener un cierto éxito, aunque apenas fue conocido fuera del ámbito universitario.

En 1966 Ralph Baer empezó a desarrollar junto a Albert Maricon y Ted Dabney, un proyecto de videojuego llamado Fox and Hounds dando inicio al videojuego doméstico. Este proyecto evolucionaría hasta convertirse en la Magnavox Odyssey, el primer sistema doméstico de videojuegos lanzado en 1972 que se conectaba a la televisión y que permitía jugar a varios juegos pregrabados.

[1]

2.1.1 1970-1979: La eclosión de los videojuegos

Un hito importante en el inicio de los videojuegos tuvo lugar en 1971 cuando Nolan Bushnell comenzó a comercializar Computer Space, una versión de Space War, aunque otra versión recreativa de Space War como fue Galaxy War puede que se le adelantara a principios de los 70 en el campus de la universidad de Standford.

La ascensión de los videojuegos llegó con la máquina recreativa Pong que es considerada la versión comercial del juego Tennis for Two de Higinbotham. El sistema fue diseñado por Al Alcom para Nolan Bushnell en la recién fundada Atari.

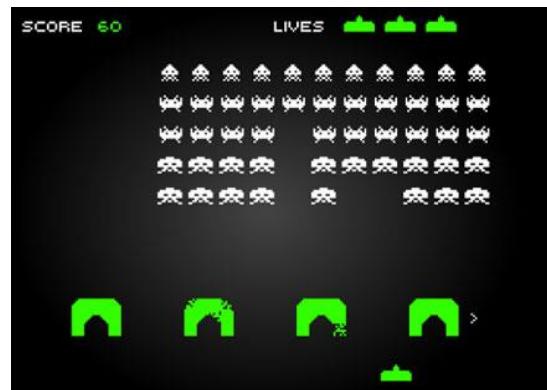


Figura 2.1.1. Space Invaders

El juego se presentó en 1972 y fue la piedra angular del videojuego como industria. Durante los años siguientes se implantaron numerosos avances técnicos en los videojuegos (destacando los microprocesadores y los chips de memoria). Aparecieron en los salones recreativos juegos como Space Invaders (Taito) o Asteroids (Atari).

[2]

2.1.2 1980-1989: La década de los 8 bits

Los años 80 comenzaron con un fuerte crecimiento en el sector del videojuego alentado por la popularidad de los salones de máquinas recreativas y de las primeras videoconsolas aparecidas durante la década de los 70.

Durante estos años destacan sistemas como Odyssey 2 (Phillips), Intellivision (Mattel), Colecovision (Coleco), Atari 5200, Commodore 64, Turbografx (NEC). Por otro lado en las máquinas recreativas triunfaron juegos como el famoso Pacman (Namco), Battle Zone (Atari), Pole Position (Namco), Tron (Midway) o Zaxxon (Sega).



Figura 2.1.2. Pacman

El negocio asociado a esta nueva industria alcanzó grandes cosas en estos primeros años de los 80, pero sin embargo, en 1983 comenzó la llamada crisis del videojuego, afectando principalmente a Estados Unidos y Canadá, y que no llegaría a su fin hasta 1985.

Japón apostó por el mundo de las consolas con el éxito de la Famicom (llamada en occidente como Nintendo Entertainment System), lanzada por Nintendo en 1983 mientras en Europa se decantaba por los microordenadores como el Commodore 64 o el Spectrum.

A la salida de su particular crisis los norteamericanos continuaron la senda abierta por los japoneses y adoptaron la NES como principal sistema de videojuegos. A lo largo de la década fueron apareciendo nuevos sistemas domésticos como la Master System (Sega), el Amiga (Commodore) y el 7800 (Atari) con juegos hoy en día considerados clásicos como el Tetris.

A finales de los 80 comenzaron a aparecer las consolas de 16 bits como la Mega Drive de Sega y los microordenadores fueron lentamente sustituidos por las computadoras personales basadas en arquitecturas de IBM.

[3]

En 1985 apareció Super Mario Bros, que supuso un punto de inflexión en el desarrollo de los juegos electrónicos, ya que la mayoría de los juegos anteriores sólo contenían unas

pocas pantallas que se repetían en un bucle y el objetivo simplemente era hacer una alta puntuación. El juego desarrollado por Nintendo supuso un estallido de creatividad. Por primera vez teníamos un objetivo y un final en un videojuego. En los años posteriores otras compañías emularon su estilo de juego.

En el campo de las recreativas, destacaron videojuegos como Defender, Rally-X, Dig Dug, Bubble Bobble, Gauntlet, Out Run o Shinobi además de producirse un cambio en cuanto a la nacionalidad de los juegos pasando a ser Japón la mayor productora.

Otra rama de los videojuegos que creció con fuerza fue la de los videojuegos portátiles. Estos comenzaron a principios de los 70 con los primeros juegos completamente electrónicos lanzados por Mattel, los cuales difícilmente podían considerarse como videojuegos, y fueron creciendo en popularidad gracias a conversiones de recreativas como las realizadas por Coleco o adictivos microjuegos como las Game & Watch de Nintendo. La evolución definitiva de las portátiles como plataformas de videojuego llegó en 1989 con el lanzamiento de la Game Boy (Nintendo).

[4]

2.1.3 1990-1999: La revolución de las 3D

A principios de los años 90 las videoconsolas dieron un importante salto técnico gracias a la competición de la llamada "generación de 16 bits" compuesta por la Mega Drive, la Super Nintendo Entertainment de Nintendo, la PC Engine de NEC, conocida como TurboGrafx en occidente y la CPS Changer de (Capcom).

[1]



Figura 2.1.3A. Sega Genesis



Figura 2.1.3B. Super NES

Junto a ellas también apareció la Neo Geo (SNK) una consola que igualaba las prestaciones técnicas de un arcade pero demasiado cara para llegar de forma masiva a los hogares.

Esta generación supuso un importante aumento en la cantidad de jugadores y la introducción de tecnologías como el CD-ROM, una importante evolución dentro de los diferentes géneros de videojuegos, principalmente gracias a las nuevas capacidades técnicas.

Mientras tanto diversas compañías habían comenzado a trabajar en videojuegos con entornos tridimensionales, principalmente en el campo de los PC, obteniendo diferentes resultados desde las “2D y media” de Doom, 3D completas de 4D Boxing a las 3D sobre entornos pre-renderizados de Alone in Dark. Referente a las ya antiguas consolas de 16 bits, su mayor y último logro se produciría por el SNES mediante la tecnología 3D de pre-renderizados de SGI, siendo su máxima expresión juegos como Donkey Kong Country y Killer Instinct. También surgió el primero juego poligonal en consola, la competencia de la SNES, Mega-Drive, lanzó el Virtual Racing, que tuvo un gran éxito ya que marcó un antes y un después en los juegos 3D en consola.



Figura 2.1.3C. Sony PlayStation



Figura 2.1.3D. Sega Megadrive

Rápidamente los videojuegos en 3D fueron ocupando un importante lugar en el mercado, principalmente gracias a la llamada "generación de 32 bits" en las videoconsolas: Sony PlayStation y Sega Saturn (principalmente en Japón); y la "generación de 64 bits" en las videoconsolas: Nintendo 64 y Atari jaguar. En cuanto a los ordenadores, se crearon las aceleradoras 3D.

[6]



Figura 2.1.3E. Nintendo 64

La consola de Sony apareció tras un proyecto iniciado con Nintendo (denominado SNES PlayStation), que consistía en un periférico para SNES con lector de CD. Al final Nintendo rechazó la propuesta de Sony, puesto que Sega había desarrollado algo parecido sin tener éxito, y Sony lanzó independientemente PlayStation.

Por su parte los arcades comenzaron un lento pero imparable declive según aumentaba el acceso a consolas y ordenadores más potentes.

[7]

Por su parte los videojuegos portátiles, producto de las nuevas tecnologías más poderosas, comenzaron su verdadero auge, uniéndose a la Game Boy máquinas como la Game Gear (Sega), Linx (Atari) o la Neo Geo Pocket (SNK), aunque ninguna pudo hacerle frente a la popularidad de la Game Boy, siendo esta y sus descendientes (Game Boy Pocket, Game Boy Color, Game Boy Advance, Game Boy Advance SP) las dominadoras del mercado.

Hacia finales de la década la consola más popular era la PlayStation con juegos como Final Fantasy VII (Square), Resident Evil (Capcom), Winning Eleven 4 (Konami), Gran Turismo (Polyphony Digital) y Metal Gear Solid (konami).

[7]

En PC eran muy populares los FPS (juegos de acción en primera persona) como Quake (id Software), Unreal (Epic Megagames) o Half-Life (Valve), y los RTS (juegos de estrategia en tiempo real) como Command & Conquer (Westwood) o Starcraft (Blizzard). Además, conexiones entre ordenadores mediante internet facilitaron el juego multijugador, convirtiéndolo en la opción predilecta de muchos jugadores, y fueron las responsables del nacimiento de los MMORPG (juegos de rol multijugador online) como Ultima Online (Origin). Finalmente en 1998 apareció en Japón la Dreamcast (Sega) y daría comienzo a la “generación de los 128 bits”.



Figura 2.1.3F. Dreamcast

2.1.4 Desde el 2000: El comienzo del nuevo siglo

En el 2000 Sony lanzó la anticipada PlayStation 2 y Sega lanzó otra consola con las mismas características técnicas de la Dreamcast, nada más que venía con un monitor de 14 pulgadas, un teclado, altavoces y los mismos mandos llamados Dreamcast Drivers 2000 Series CX-1.

Microsoft entra en la industria de las consolas creando la Xbox en 2001.

Nintendo lanzó el sucesor de la Nintendo 64, la Gamecube, y la primera Game Boy completamente nueva desde la creación de la compañía, la Game Boy Advance. Sega viendo que no podría competir, especialmente con una nueva máquina como la de Sony, anunció que ya no produciría hardware, convirtiéndose sólo en desarrolladora de software en 2002.

El ordenador personal PC es la plataforma más cara de juegos pero también la que permite mayor flexibilidad. Esta flexibilidad proviene del hecho de poder añadir al ordenador componentes que se pueden mejorar constantemente, como son tarjetas gráficas o de sonido y accesorios como volantes, pedales y mandos, etc. Además es posible actualizar los juegos con parches oficiales o con nuevos añadidos realizados por la compañía que creó el juego o por otros usuarios.

[1]

2.2 Historia del arte

- **ARTE:** *Acto mediante el cual, valiéndose de la materia o de lo visible, imita o expresa el hombre lo material o lo invisible, y crea copiando o fantaseando. En sentido amplio, podemos denominar como Arte a toda creación u obra que exprese lo que el hombre desea exteriorizar, obedeciendo a sus propios patrones de belleza y estética. El artista para crear, requiere ante todo estar dotado de imaginación, a través de la cual responde al vasto y multiforme mundo externo expresando sus sentimientos por medio de palabras, formas, colores y sonidos.*

[8]

Tras esta definición introductoria podríamos repasar toda la historia del arte, pero dada su extensión y dado que este proyecto pretende centrarse más en el desarrollo de un videojuego relacionado con la historia del arte y en la historia del arte en sí, nos centraremos solamente en las épocas artísticas/históricas que aparecen en el juego, que son tres.

2.2.1 Época clásica: Renacimiento

El Renacimiento comenzó como un movimiento orientado por artistas e intelectuales en Italia, bajo el signo del Humanismo; es un renacer de las artes donde los asuntos representados, desde el punto de vista ético y estético se liberaron de los vínculos del concepto de vida cristiano. Para ellos el arte ya no era un servicio anónimo, ofrecido a Dios y a la iglesia, sino un himno personal en alabanza a la belleza; así, se perfecciona el dibujo y se utiliza como base de la pintura. La cuna del Renacimiento fue Florencia. Naturalmente la pintura no se liberó en seguida de la influencia gótica, pero gradualmente evolucionó hacia un nuevo concepto de la belleza. En el dibujo, los cuerpos adoptan formas naturales y se vuelven plásticos; se procura destacar la expresión facial, que algunas veces revela los grandes conflictos del alma, un ejemplo de ello es la expresión de desesperación que Masaccio le dio a Eva en su cuadro Expulsión de Adán y Eva del paraíso. En el siglo XV adquiere preponderancia el retrato, a las personas pudientes les gusta retratarse de busto o en medallón, y por tanto surge multitud de personas cuyos rasgos quedan labrados en madera. Raramente se ve un desnudo entre la profusión de vírgenes y santos, sólo con suma discreción se insinúa la mundana sensualidad en ciertas representaciones del arte eclesiástico, por ejemplo, las referentes a mártires y pecadores. Al principio, la mayoría de las imágenes alusivas a la carne pecadora se situaban en la periferia de grandes escenas decorativas, donde el artista tenía más libertad de expresión; ya en el siglo XIV, se prefiere interpretar estos temas por medio del desnudo femenino. Entre los representantes más significativos del Renacimiento, podemos destacar por sus majestuosas obras pictóricas a Sandro Botticeli, Miguel Ángel Buonarroti, Durero, Tintoretto, El Greco, Leonardo de Vinci y Rafael Sanzio; de ellos, algunos como Leonardo de Vinci, por ejemplo, destacó más como dibujante, ya que a través del dibujo realiza sus famosos estudios anatómicos; sus dibujos están plenos de rasgos finos pero firmes, destacando las expresiones humanas y también se puede apreciar que están envueltos en una sutil y fina aureola de luz difusa.

[9]



Figura 2.2.1A. Retrato de La Mona Lisa, obra de Leonardo da Vinci

Se llama Renacimiento al gran movimiento artístico y filosófico que se produce en Europa, en Italia en primer lugar, a fines del siglo XV, y que muestra como principal característica, que se manifiesta particularmente en las artes, su admiración por la antigüedad clásica, que toma como modelo. El nombre de Renacimiento alude a lo que este movimiento quiso ser: un renacer o volver a nacer de la cultura grecolatina. Comenzó como un movimiento orientado por artistas e intelectuales en Italia, bajo el signo del Humanismo; es un renacer de las artes donde los asuntos representados, desde el punto de vista ético y estético se liberaron de los vínculos del concepto de vida cristiano. Para ellos el arte ya no era un servicio anónimo, ofrecido a Dios y a la iglesia, sino un himno personal en alabanza a la belleza.

[10]

La cuna del Renacimiento fue Florencia. Naturalmente la pintura no se liberó en seguida de la influencia gótica, pero gradualmente evolucionó hacia un nuevo concepto de la belleza. La cultura greco-romana había sido desplazada durante la Edad Media. En esta época lo novedoso es el arte gótico y bizantino, pero en Roma, estas nuevas concepciones artísticas, enmarcadas en el acto religioso no tienen mayor auge, debido a los recelos de los sabios humanistas orientales que emigran a esta ciudad después de la caída de Constantinopla; es así como al ser rechazado el estilo gótico y bizantino, y puestas en un primer plano las antiguas formas greco-romanas, surge el arte del renacimiento, que se expande por toda Europa (Francia, Inglaterra, Alemania y la Península Ibérica, especialmente).

[10]

PRINCIPALES CAUSAS DEL RENACIMIENTO

- Conservación en universidades y conventos medievales de valiosos manuscritos de autores griegos y romanos.
- Uso del latín como lengua culta, que hacía posible la lectura de las obras clásicas.
- La presencia en tierra Italiana de ruinas romanas que tenían que despertar en los curiosos el deseo de conocer la civilización que había levantado tales monumentos.
- La invención de la imprenta, que contribuyó a la difusión de los escritos de los poetas, filósofos y sabios de la antigüedad y de los modernos.
- Los descubrimientos geográficos, el avance de las ciencias naturales y el progreso de las técnicas que inspiran una confianza ilimitada en el poder de la inteligencia humana y estimulan a la acción.

[11]



Figura 2.2.1B. Adán de Miguel Ángel, en la Capilla Sixtina, con el Creador

CARACTERÍSTICAS GENERALES DEL ARTE RENACENTISTA

- Imitación de la arquitectura y la escultura de Grecia y Roma.
- Realización de una belleza ideal, ajustada a cánones dictados por la razón.
- Búsqueda de la serenidad y el equilibrio que proceden de la armonía del todo.
- Creación de obras, cuya claridad y perfección, atributos exigidos por la razón universal, les dan una validez permanente.

[11]

ETAPAS DEL RENACIMIENTO

- PRERRENACIMIENTO O TRECENTO: siglos XIII y XIV. Coincide con el período gótico europeo.
- QUATROCENTO: llega hasta finales del siglo XV y su centro cultural es la ciudad.
- CINQUECENTO: llena todo el siglo XVI y su cabeza es Roma.

[11]

2.2.2 Época vanguardista: Abstracción

Hablaremos primero del arte vanguardista en general, para después concretar con el arte abstracto.

2.2.2.1 Arte vanguardista

Diferentes rupturas con los modelos de belleza dominantes en Europa durante el primer tercio del siglo XX. Este es nombrado en plural ya que hace referencia a tendencias o movimientos, no a un sólo estilo artístico.

Las razones o motivos principales para el surgimiento del arte vanguardista fueron 3 acontecimientos políticos, la constitución de la segunda y la tercera República Francesa (1848 y 1871) y la Primera Guerra Mundial (1914). Estas provocaron una reacción intelectual en contra de la sociedad de la época lo que provocó el comienzo del estereotipo de un artista incomprendido, bohemio y comprometido con una serie de valores contrarios a todo el mundo que provocaba situaciones miserables y desafortunadas.

Además hubo un acontecimiento artístico que influyó sobre el surgimiento del arte vanguardista, este fue el comienzo de los Salones de París en el cual se encontraban diferentes muestras artísticas anuales de prestigio que contaban con un jurado y donde fueron rechazados varios pintores impresionistas. Por lo cual inauguraron el Salón de los

Rechazados con el objetivo de que sus trabajos fueran apreciados y valorados por el público.

También para entender el comienzo del Arte Vanguardista se debe tener en cuenta aspectos del Siglo siguiente, en el cual hubo muchos cambios y aportaciones significativas que modificaron ideas y modos de vida. Entre ellos se encuentran la segunda Revolución Industrial, con la aparición del motor de explosión, la publicación de la Teoría de la Relatividad de Albert Einstein y la Interpretación de los sueños de Sigmund Freud, la popularización de la fotografía y además el nacimiento del Cine, entre otros.

[12]



Figura 2.2.2.1. Gernica de Picasso, ejemplo del arte vanguardista, en concreto cubista.

Características generales del Arte Vanguardista:

- La ruptura con lo anterior.
- El deseo de novedad y experimentación.
- Representan una desavenencia total (del color, de las normas, de la composición y del lenguaje estético.)
- El cambio es producido también desde una ideología ya que comporta una pendencia de pensamiento y de valores.

[12]

En el siglo XIX James McNeill Whistler realizó el primer cuadro abstracto, en su pintura Nocturne in Black and Gold: The falling Rocket, dando más énfasis a la sensación visual que a la representación de los objetos.

Los principales acontecimientos que ocurrieron para provocar esta obra fueron tres movimientos artísticos: El Romanticismo, el impresionismo y el expresionismo; consiguiendo la independencia de los artistas. Además el mecenazgo de la iglesia disminuyó y el privado del público fue más capaz de proporcionar una forma de vida para los artistas.

Desde principios del siglo XX las conexiones culturales entre los distintos artistas europeos y norteamericanos se volvieron muy activas, tratando de crear una forma de arte que igualara las aspiraciones del modernismo. Estas ideas fueron capaces de influirse a través de libros de artistas, exposiciones y manifiestos de manera que muchas fuentes estaban abiertas a la experimentación y formaron la base de la diversidad de los modos de abstracción.

[12]

2.2.2.2 Arte abstracto

Arte abstracto: Estilo artístico que enfatiza los aspectos cromáticos, formales y estructurales, acentuándolos, resaltando su valor y fuerza expresiva, sin tratar de imitar modelos o formas naturales. Este deja de considerar necesaria la representación figurativa y tiende a sustituirla por un lenguaje visual autónomo. Usa este lenguaje visual de forma, color y línea para crear una composición que pueda existir con independencia de referencias visuales del mundo real.

La abstracción utilizada en las diferentes obras se puede apartar de la realidad de una forma ligera, parcial o completa. La abstracción pura nació en 1910 como reacción del realismo e influido por la aparición de la fotografía la cual provocó la crisis del arte figurativo. Además el arte abstracto designa una serie de tendencias en pintura, escultura y artes gráficas, que rechazan la copia o la imitación de cualquier modelo exterior a la conciencia del artista; prescindiendo de toda figuración (espacio real, objetos, paisajes, figuras, seres animados e incluso formas geométricas si se representan como objetos reales, con iluminación y perspectiva), proponiendo una realidad distinta a la realidad.

[13]



Figura 2.2.2.2. Cuadro moderno abstracto de Kandinsky (Rusia)

Una de las razones principales para el surgimiento del arte abstracto fue la accesibilidad del arte de las culturas distintas a Europa, las cuales mostraron formas alternativas de describir experiencias visuales a los artistas. A diferencia del arte occidental el cual estaba sometido a la lógica de la perspectiva y a un intento de reproducir una ilusión de realidad visible.

Así, a fines del siglo XIX muchos artistas buscaron crear un nuevo tipo de arte que demuestre los cambios que se estaban produciendo en la tecnología, ciencia y filosofía. Además reflejaban las preocupaciones intelectuales y sociales en todas las áreas de la cultura occidental de esa época.

La abstracción pura nació en 1910 como reacción del realismo e influido por la aparición de la fotografía la cual provocó la crisis del arte figurativo. Esto se elaboró a partir de las experiencias de vanguardia precedentes: el fauvismo y el expresionismo que liberaron el color, lo que derivó hacia la abstracción lírica o informalismo; el cubismo que hizo hincapié en la conceptualización de la forma y de la composición, lo que llevó a otro tipo de abstracción (geométricas y constructivas). Tanto la abstracción geométrica como la abstracción lírica son a menudo totalmente abstractas. Debe tenerse en cuenta que el arte abstracto desde sus comienzos, ha tendido hacia dos polos: uno, cuyos orígenes se remontan al fauvismo, libre y lírico; el otro, inspirándose más en el cubismo, rigurosamente geométrico.

[13]

2.2.3 Época técnica: Grabado Xilográfico

Xilografía. La palabra xilografía proviene del término griego “xylon” - que significa madera - y del término “grafos” - grabado -. Por tanto, cuando hablamos de una xilografía o grabado xilográfico nos referimos a un grabado realizado sobre una plancha de madera. El proceso xilográfico consiste en dejar en relieve aquellas partes del bloque de madera que corresponden al dibujo, mientras que el resto se vacía. En el momento de imprimir se entinta la superficie que sobresale.

La xilografía es el sistema de grabado más antiguo que se conoce. En Oriente ya se practicaba des del siglo VII, especialmente para la estampación de tejidos. En Europa su introducción fue más tardía y, aunque los primeros testimonios que se conservan son del siglo XIV - juegos de naipes, estampas religiosas y calendarios -, habrá que esperar hasta el siglo XV, con la difusión del papel, para poder hablar de la auténtica expansión de la xilografía.



Figura 2.2.3. Ejemplo de arte xilográfico

A pesar de que es una técnica básicamente aplicada a la reproducción de imágenes, también fue utilizada para la reproducción de textos. La gran expansión de la xilografía se produjo en el siglo XV, sobre todo en Alemania, donde destaca la figura de Albrecht Dürer. También hemos de tener en cuenta la obra de Lucas Cranach y ya en el siglo XVI la de Hans Holbein. En el siglo XVII el grabado en madera entró en una fase de declive al verse desplazado poco a poco por el grabado en metal. A pesar de ello, en la siguiente centuria se vivió un verdadero resurgimiento, fruto de la difusión de la xilografía a testa llevada a cabo por Thomas Bewick.

[14]

Esta nueva técnica de talla de la madera permitía alcanzar un nivel de precisión muy elevado, lo que motivó que durante el siglo XIX se aplicase sobre todo a la ilustración de libros y de publicaciones periódicas. Una vez liberada de su función utilitaria, gracias a la aparición de la fotografía y de los sistemas fotomecánicos de reproducción, artistas como Paul Gauguin, Eduard Munich y los miembros del grupo expresionista alemán “Die Brücke” encontraron en la xilografía un extraordinario medio de expresión y creatividad.

[14]

2.3 El videojuego como obra de arte

Los videojuegos siempre han sido objeto de debate en cuanto a si deben considerarse arte como tal o no. Multitud de personas a lo largo de la historia reciente, y ya no tan reciente, se han cuestionado este hecho, y aún hoy en día a pleno siglo XXI no está clara la respuesta. Mientras que unos defienden que por supuesto deberían ser considerados arte, entre los cuales nos incluimos los creadores de este proyecto, otros no lo ven así, pero en cualquier caso hay que reconocer que tiene potencial de sobra para serlo, no hay más que ver algunos ejemplos de juegos que superan en calidad artística a cualquier otra cosa considerada “obra de arte” según ciertos colectivos. Todo es relativo, como suele pasar en muchas ocasiones.



Figura 2.3A. Ejemplo del arte que involucra un videojuego.

No es tarea fácil, considerando la tendencia de la sociedad a establecer prejuicios sobre los videojuegos, relacionándolos con la idea de pasatiempo infantil, “comecocos”, o de adolescentes ociosos, transmisores de valores erróneos, violencia o, simplemente, entretenimiento banal. Cabe decir que gran parte de estos prejuicios son a causa de la ignorancia respecto al sector y la cultura de la persona.

[15]

A pesar de ello, quienes reconocemos la belleza artística de algunos videojuegos y damos nuestro voto en apoyo a su reconocimiento dentro del mundo del arte, lo cual es cada día más palpable, podemos echar algo de luz sobre el asunto.

Para corroborar nuestra opinión se pueden hacer comparaciones con otras disciplinas artísticas para denotar los factores que se comparten con las artes y así fortalecer los argumentos que defendemos. No todos los videojuegos pueden considerarse “artísticos”,

por eso hacemos hincapié en la idea del potencial artístico implícito en el videojuego como disciplina y la existencia de obras dignas de reconocimiento.

[15]

Para comenzar esta exposición de hechos, vamos a pensar en que ofrece un buen videojuego dentro de los parámetros artísticos, no de jugabilidad. Considerando el amplio rango de posibilidades, podemos destacar lo siguiente:

- Diseño de personajes y escenarios: especialmente hoy en día, donde los videojuegos son superproducciones, muchos dibujos y diseños son encargados a artistas importantes y de calidad, consiguiendo resultados impactantes.
- Generación de ambientes: hay juegos que parecen simples pero que transmiten algo, y el mero hecho de crear una atmósfera en la que uno se sienta inmerso es todo un logro.
- Texturas e iluminación: tanto juegos que pretenden parecerse cada dia más a la realidad, como otros en los que el componente visual es el punto impactante y una delicia visual.
- Banda sonora: como en el elemento de diseño, este elemento no es menos importante, y hay juegos con bandas sonoras hechas, por ejemplo, por orquestas sinfónicas, dignas del más alto reconocimiento y que transmiten calidad, en consecuencia, llegan a emocionar.
- Argumento y narrativa: para algunos jugadores esto lleva a ser lo más relevante, ya que hay argumentos de videojuegos que nada tienen que envidiar a ningún libro o película, y los cuales acumulan una cantidad de seguidores inimaginable, seguidores a la espera de la continuación de esa trama.
- Creatividad: participación de profesionales de distintas disciplinas artísticas pueden hacer de un videojuego una verdadera obra de arte de alta creatividad.

Observando capturas de algunos videojuegos, es fácil apreciar el potencial artístico de las imágenes, algunas son dignas de exponer en las mejores galerías y museos de arte. Añadiéndole todas las mecánicas y el movimiento es cuando se ve el potencial que puede llegar a alcanzar.

[15]



Figura 2.3B. *Okami*, otro ejemplo del arte que involucra un videojuego.

Si hay algo especial en el tipo de arte del videojuego es la interactividad. En cualquier otra de las artes, cuando oímos, vemos, escuchamos, la elección queda (casi) siempre excluida. Somos espectadores pasivos de lo narrado o sentido. Alguien predefine cada paso, sonoro o plástico, del camino. Sin embargo, el videojuego permite que el receptor de la obra asuma un papel sino creador, sí activo en la obra que se le presenta. El videojuego nos da un recurso de una potencia que aún desconoce sus infinitos horizontes. Nos permite elegir.

Hay, para quien esto escribe, un punto de inflexión en la relación del videojuego y el público masivo. Un punto propiciado, en realidad, por el cine. En 1995, con Toy Story, Pixar no solo demuestra que las imágenes generadas por ordenador pueden narrar una historia con la eficacia de la imagen real (o los dibujos animados); también demuestra que esa historia puede contener un sensibilidad honda, una profundidad liberada de muchas de las limitaciones del cine real. A la larga, de hecho, Pixar acabará convirtiéndose en reducto artístico dentro del cine, manantial de creatividad al que público y crítica acabará acudiendo en busca de verdadero arte. Para los más escépticos, aducir que muchos videojuegos (La saga Metal Gear, o las mejores entregas de Final Fantasy, por poner un par de ejemplos mayores) son en realidad una enorme película animada, compacta, coherente y emotiva (a la altura de sus hermanas prestigiosas), cosida por las partes interactivas. Y solo es un ejemplo dentro de un mundo sin límites para la creatividad.

Luego está la parte emocional. Parte nada desdeñable en esta materia. Aquella parte relacionada con la infancia, o con el descubrimiento; el poder de aquello con lo que te has formado, capaz de convertir en clásico algo personal. En ese campo sí que no hay dudas: solo cabe reivindicar que jóvenes generaciones enteras han recorrido su camino de formación personal, su educación sentimental, acompañados de Link, Snake, Cloud o Ryo Hazuki, como generaciones anteriores iban de la mano de la Señora Bovary, Ismael, Dartagnan o el joven Caulfield. Quizá yo esté viviendo dentro de un terrible error y sea sacrilegio mezclar todos esos nombres citados. Pero releo la frase anterior y no me parece estar exagerando.

[15]



Figura 3.3C. Personajes clásicos de la historia de los videojuegos

Las reticencias que supone el reconocimiento oficial del videojuego como forma de arte son achacables a la relativa modernidad del tema, siendo tan, y cada vez más, innovador y diferente a las demás artes.

Basta recordar otras formas de arte, hoy muy posicionadas en su categoría artística, como la fotografía o el cine, las que sufrieron un proceso similar cuyo rechazo inicial fue absoluto para finalmente ser legitimadas como arte.

De cualquier forma, el videojuego se configura a sí mismo como arte o medio artístico con sobrados argumentos. Hoy en día hay numerosas ferias o premios internacionales destacados en el mundo del arte están incluyendo dentro de sus categorías artísticas el reconocimiento al trabajo creativo de los videojuegos, así como ferias y premios propios exclusivamente de éstos.

Cabe mencionar ferias como el E3. La Electronic Entertainment Expo, también conocida como E3, es la convención de videojuegos más importante de la industria.

La E3 era celebrada la tercera semana de mayo de cada año en el Centro de Convenciones de Los Ángeles en la ciudad de Los Ángeles, California. Como excepciones, en 2007 se celebró del 11 al 13 de julio en Santa Mónica, California y en 2009 se celebró del 1 de junio al 4 de junio en Los Ángeles. Según la Entertainment Software Association (ESA) la E3 de 2005 superó los 70 000 visitantes. En la actualidad, se celebra en el mes de junio o julio.

Muchos desarrolladores de videojuegos exponen sus creaciones y novedades en software y hardware. Además desde 1998 se entrega un premio al mejor videojuego de la E3.



Figura 2.3D. Logotipo de la feria E3.

Como vemos, los videojuegos están comenzando poco a poco a ser reconocidos como medio audiovisual de interés artístico. Y nos complace decir que el camino se está abriendo a paso firme.

El videojuego se confirmó finalmente como categoría de arte en 2012 por la National Endowment de EEUU, entidad que subvenciona proyectos de carácter artístico.

[15]

2.4 Inspiraciones

Cuando los primeros videojuegos empezaron a aparecer no tenían fuentes de referencia , en cuanto a otros videojuegos respecta, dejando volar la creatividad de los creadores para crear contenido totalmente nuevo y original. Fue así como se fueron y se han ido consagrando los clásicos, algunos considerados ya objetos de culto por algunos, entre los que nos incluimos.

Dicho esto pues, es fácil deducir que gran parte de los videojuegos de hoy en día tienen una serie de inspiraciones y/o referencias de clásicos y otros no tan clásicos, dado el gran bagaje del que disponemos, gracias al ritmo frenético de creación de videojuegos que la industria ha alcanzado hoy en día.

Exponemos aquí una serie de ejemplos de videojuegos de los cuales hemos sacado alguna idea con el hecho de reinventarla o modificarla, o simplemente adaptar alguna mecánica que nos haya parecido atractiva, acorde al proyecto y viable de implementar.

2.4.1 Apotheon

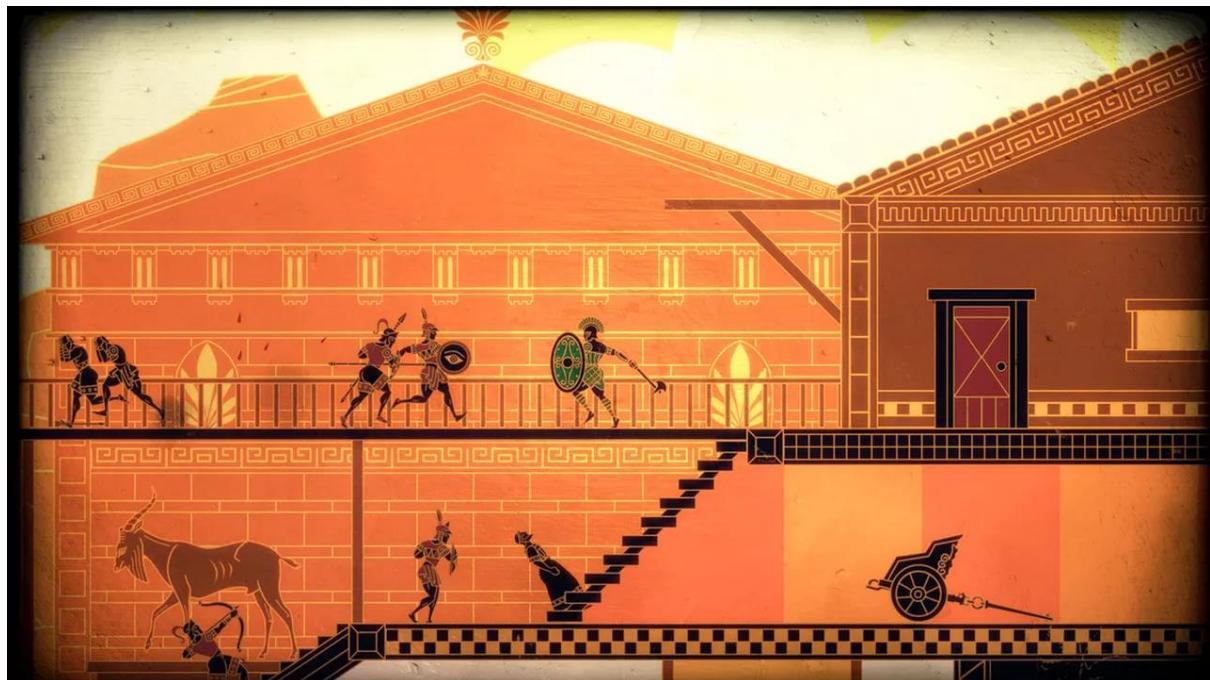


Figura 2.4.1. Apotheon

El peculiar diseño artístico de Apotheon es lo que realmente nos ha cautivado, el cual parece sacado de una vasija de la antigua Grecia. Esto no significa que sus demás apartados sean menos llamativos, de hecho posee una genial banda sonora, argumento basado en la mitología griega y un sistema de juego clásico de avanzar y pegar, pero a la vez fresco y que insta a jugarlo hasta el final.

2.4.2 Limbo



Figura 2.4.2. Limbo

Este oscuro juego indie, es decir, realizado por un estudio pequeño y con pocos recursos, fue toda una sorpresa tan su lanzamiento.

De mecánicas simples (avanzar y resolver ciertos puzzles) pero de gran ambientación y tensión, y es esto precisamente en lo que nos hemos fijado , en el hecho en que la aparente simpleza de la jugabilidad se ve amplificada por la ambientación, la cual te envuelve en todo momento y causa sensación continua de soledad y tensión combinadas.

Además es necesario solventar algunos puzzles que nos harán pensar y estar un rato para resolverlos.

2.4.3 Nihilumbra



Figura 2.4.3. Nihilumbra

Nihilumbra es un juego realizado por BeutiFun Games, un estudio español, y se juega como un videojuego de plataformas estándar, ya que controlas a Born mientras camina y salta a través de los niveles del juego. Hay varios enemigos que el jugador tiene que evitar, ya que, al principio, no hay manera de derrotarlos. El juego se divide en los cinco mundos que Born explora. En cada uno de ellos, al jugador se le concede un nuevo color, con el que el jugador puede pintar sobre el terreno (tocando la pantalla) para modificar el comportamiento del medio ambiente.

Estas mecánicas nos han sido de inspiración para el tema de las habilidades de nuestro personaje, así como el uso del ratón para ejecutar ciertas acciones a la vez que el personaje está en movimiento.

2.4.4 Rock of Ages

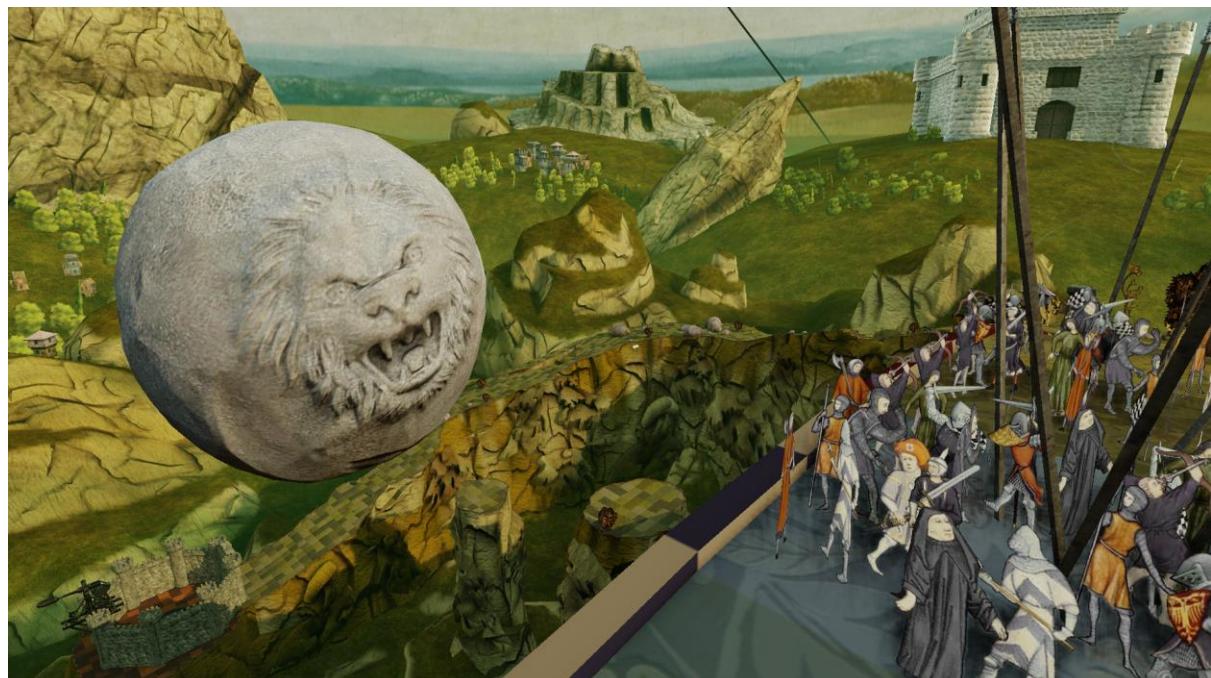


Figura 2.4.4. Rock of Ages

Siguiendo con el mismo tono que Apotheon, en este juego llamado Rock of Ages se puede ver como se hace uso del humor para representar algunas de las escenas. El estilo de juego es totalmente diferente y no nos concierne, era solamente para recalcar que se podría hacer uso del humor en ciertas partes del juego.

2.4.5 Ori and the Blind Forest



Figura 2.4.5. Ori and the Blind Forest

Un ejemplo de juego que cuenta una historia sencilla a la vez que emotiva es *Ori and the Blind Forest*. En él nos hemos basado para adoptar esta idea, la idea de crear un pequeño guión sencillo acorde a la temática del juego, donde se cuenta una historia con principio y final, con el fin de provocar cierto sentimiento de satisfacción o completitud al acabarlo.

2.4.6 The Witness



Figura 2.4.6. The Witness

Éste es un juego que hemos jugado recientemente, aunque es en 3D, está basado en la resolución de puzzles bidimensionales con patrones que se han de deducir observando el entorno e intentando encontrar un sentido lógico.

Hemos pensado en adaptar la idea de este tipo de puzzles para algunas fases del juego, donde prime la observación y haya que usar la cabeza un poco.

El desarrollo de este juego ha sido de 8 años en total, con una complejidad sin igual, complejidad que nosotros no podemos, ni necesitamos, recrear, dado nuestro tiempo de desarrollo.

2.4.7 The Unfinished Swan



Figura 2.4.7. The Unfinished Swan

Otro ejemplo de juego en primera persona con una peculiar mecánica es The Unfinished Swan, donde sobre un aparente mapeado totalmente blanco vamos lanzando bolas de pintura negra y vamos descubriendo los elementos del mapa.

Nos hemos basado en esta idea para el aspecto de la época de la xilografía, donde todo está en escala de grises y se puede jugar con ello.

2.4.8 Psychonauts



Figura 2.4.8. Psychonauts

A nivel artístico, en esta fase del llamado Psychonauts se puede observar como el diseño del entorno hace referencia al arte, teniendo un acabado peculiar y bastante curioso. Hemos querido plasmar algo parecido, gracias a los diseños de Ximena, con lo cual todo el juego en sí muestra constantemente referencias a la época artística/histórica correspondiente.

2.4.9 Super Mario 64



Figura 2.4.9. Super Mario 64

Hace ya muchos años, en la época de Nintendo 64, el juego Super Mario 64 se basaba en una zona principal o lobby, representado por un castillo, donde había varios cuadros grandes colgados de las paredes, donde cada uno representaba una fase del juego y requería un mínimo de estrellas (el objetivo del juego era reunir cuantas más estrellas posibles) para poder acceder al nivel en cuestión.

Decidimos que se podría adaptar la idea para recrear una sala principal donde cada cuadro representa un estilo o período artístico diferente y a los cuales podemos acceder libremente.

2.4.10 Inside Out



Figura 2.4.10. Escena de la abstracción en Inside Out

En la película de animación de Pixar, Inside Out, hay una escena donde los personajes entran a una sala donde se ven sometidos a un proceso de abstracción, pasando por sus diferentes fases, donde van cambiando su apariencia. Dado que tenemos una fase de abstracción en el juego nos ha parecido interesante adoptar una mecánica similar, viéndose el personaje principal afectado en esta fase del juego y siendo posible cambiar su forma física.

Link a la escena en concreto: [\[16\]](#)

2.4.11 The Legend of Zelda: A Link Between Worlds

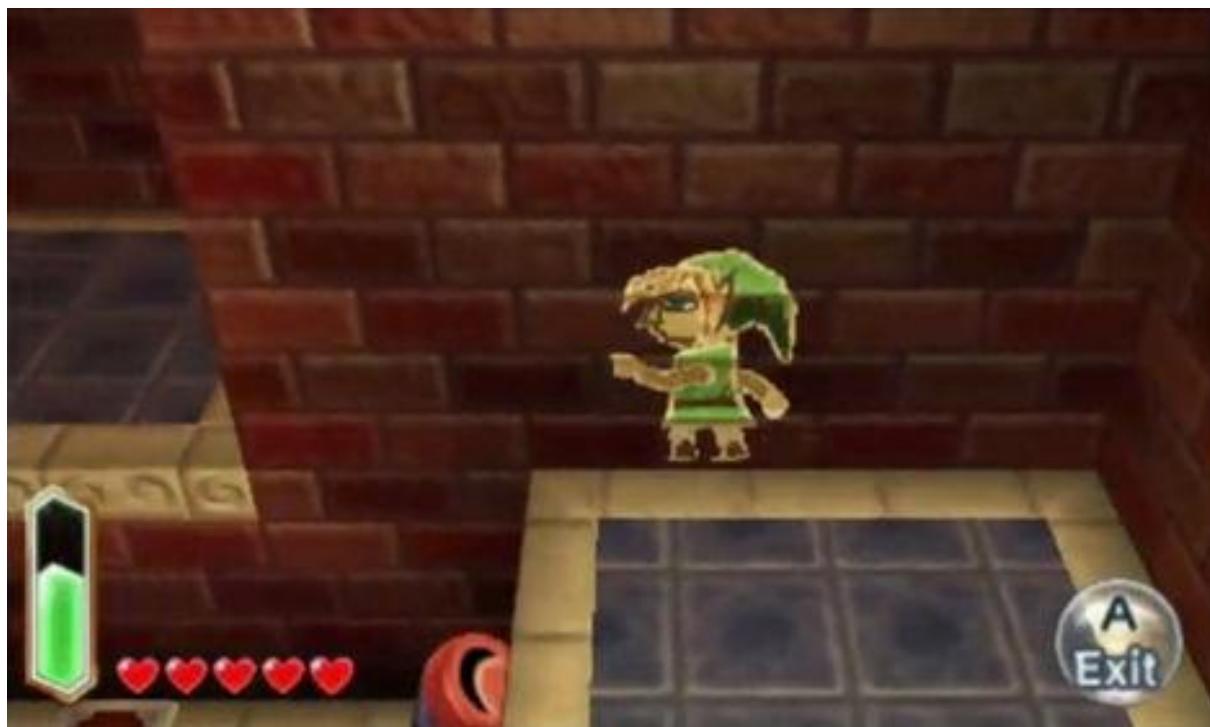


Figura 2.4.11. Ejemplo de mecánica de Zelda: A Link Between Worlds

De este juego de esta tan famosa saga nos hemos inspirado en la habilidad especial del personaje Link para transformarse en dos dimensiones pegándose a la pared como si de un dibujo se tratara, pudiendo avanzar hacia zonas donde sin esta habilidad no sería posible.

Hemos empleado una técnica similar en nuestra fase de abstracción.

3. Análisis

A continuación se detallan los actores implicados en el sistema, requisitos funcionales y no funcionales, ejemplo de caso de uso y por último los diagramas generados para el desarrollo.

3.1 Análisis de requisitos

3.1.1 Actores

Los actores que interactúan en este proyecto se reducen solamente a uno, el **jugador** propiamente dicho.

Ya que el juego es una experiencia de un solo jugador, no habrá otro que interactúe aparte de éste, aunque como trabajo futuro se pretende implementar un sistema de multijugador.

El flujo del juego depende pues en su totalidad del jugador y de cómo vaya avanzando en el juego, si va realizando unas tareas u otras.

3.1.2 Requisitos funcionales

En esta sección se describen las funcionalidades básicas que el juego debe cubrir.

- **RF-1:** Moverse por un escenario 2D libre.
- **RF-2:** Moverse por un escenario 2,5D libre.
 - RF-2.1: Alternar entre modelo 2D y modelo 3D a voluntad.
- **RF-3:** Interactuar con el escenario.
- **RF-4:** Interactuar con los enemigos
 - **RF-4.1:** Interactuar con enemigos terrestres
 - **RF-4.2 :** Interactuar con enemigos voladores
- **RF-5:** Interactuar con los objetos.
 - **RF-5.1:** Recoger objetos colecciónables
 - **RF-5.2:** Permitir el cambio de fase si tenemos los objetos necesarios.
- **RF-6:** Interactuar con las físicas
 - **RF-6.1:** Interactuar con las físicas 2D.
 - **RF-6.2:** Interactuar con las físicas 3D.
- **RF-7:** Mantener un flujo lineal del juego
 - **RF-7.1:** Cambiar de escena
- **RF-8:** Crear un hilo argumental acorde a la temática.
- **RF-9:** Reflejar fielmente las épocas artísticas involucradas.
 - **RF-9.1:** Crear un mapa acorde al Renacimiento.
 - **RF-9.2:** Adaptar una temática abstracta en unas de las fases.
 - **RF-9.3:** Reflejar el arte xilográfico como escenario.
- **RF-10:** Mostrar secuencias cinematográficas

- **RF-10.1:** Mostrar secuencia de introducción
 - **RF-10.2:** Mostrar secuencia de transición entre escenas
- **RF-11:** Narrar el argumento con voces.
 - **RF-11.1:** Usar distintas voces dependiendo del personaje.
- **RF-12:** Apoyar el flujo del juego con música adecuada.
- **RF-13:** Mostrar los créditos al superar el juego.

3.1.3 Requisitos no funcionales

- **RNF-1:** Obtener los recursos de fuentes open-source.
- **RNF-2:** El control del personaje ha de ser intuitivo.
 - RNF-2.1: El control del personaje 2D ha de responder de manera fluida.
 - RNF-2.2: El control del personaje 3D ha de responder fluidamente y hacer buen uso de las físicas 3D.
- **RNF-3:** El juego ha de poder correr en un ordenador corriente, sin grandes requerimientos técnicos.
- **RNF-4:** La dificultad ha de ser accesible para todo el mundo, no siendo ni muy fácil ni muy complicado.
- **RNF-5:** El jugador ha de saber en todo momento qué ha de hacer y de qué controles dispone para hacerlo.
- **RNF-6:** El diseño ha de ser atractivo.
- **RNF-7:** Se ha de hacer un uso eficiente de los recursos, destruyendo objetos innecesarios.
- **RNF-8:** Representar fielmente las distintas épocas artísticas.

3.1.4 Requisitos de diseño (restricciones)

- **RD-1:** El juego al ser un prototipo, solamente será jugable inicialmente en plataformas Windows.
 - **RD-1.1:** Los controles están ajustados solamente para esta plataforma.
 - **RD-1.2:** Solamente podemos apuntar con el ratón.
- **RD-2:** No se dispone función de guardado de partida dada la brevedad del prototipo del juego.
- **RD-3:** No se informa explícitamente al jugador de los controles. (Pero existe un manual de usuario donde se especifican los controles).

3.2 Ejemplo de caso de uso

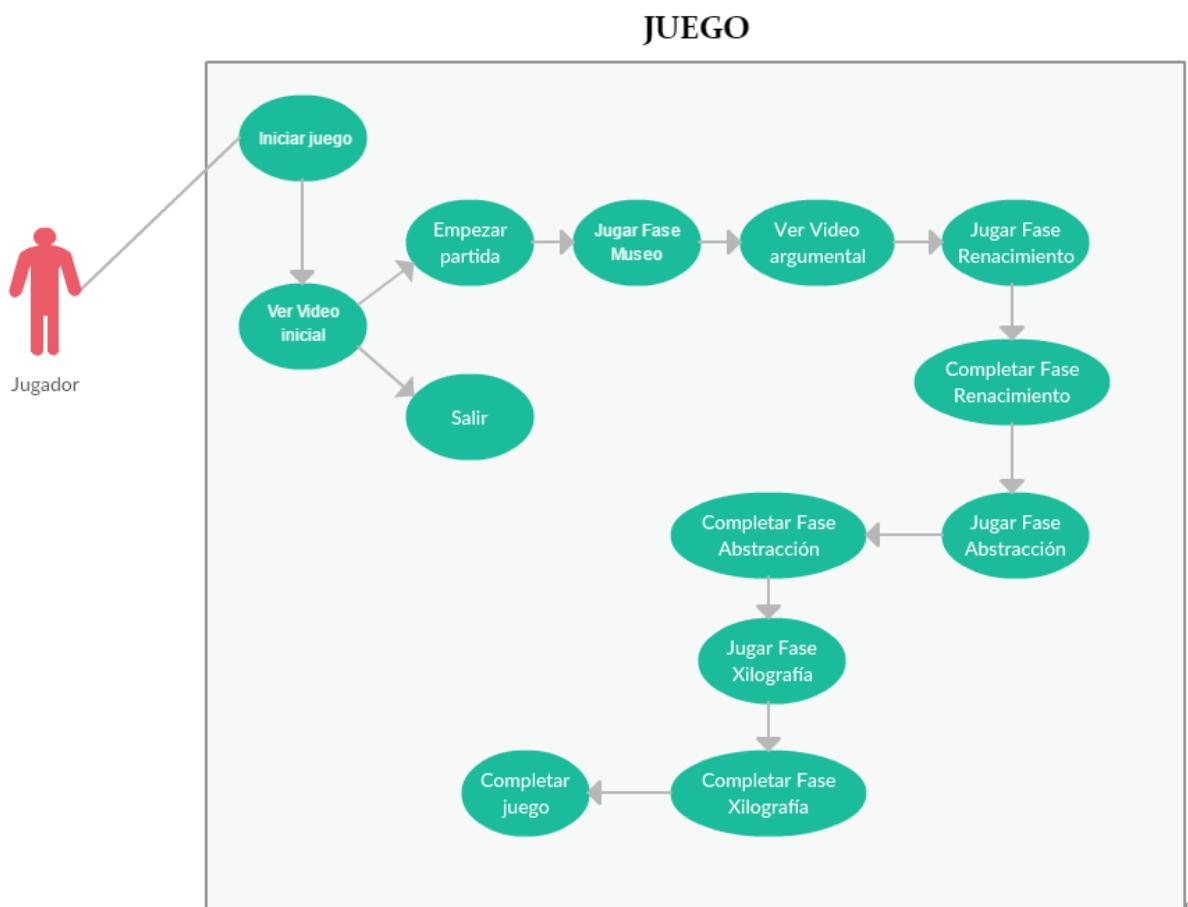


Figura 2.2. Ejemplo de caso de uso

4. Herramientas

4.1 Requerimientos

Para poder ejecutar el videojuego se necesita recurrir a un hardware de gama baja.

- Dispositivo con el sistema operativo Windows (cualquiera de las últimas versiones)
- Memoria RAM de 1 GB RAM
- Memoria de video de 256 MB VRAM
- Procesador de 1 GHz CPU o más
- Teclado y ratón

4.2 Software y Lenguajes de Programación

4.2.1 Motor gráfico

Unity3D dispone de su propio motor gráfico integrado.

4.2.2 Lenguajes usados

- C#

4.2.3 IDE

- Visual Studio 2015
- MonoDeveloper (incluído en Unity)

4.3 Plataformas

Unity permite exportar a muchas plataformas (PC; Mac, Android, IOS, etc...) pero en nuestro caso, al tratarse de una demo y no de un juego de desarrollo totalmente completado, se ha generado exclusivamente para PC, en concreto para el sistema operativo Windows.

4.4 Unity

Unity es el motor de videojuegos elegido para realizar éste proyecto. Es una herramienta multiplataforma creado por Unity Technologies, lanzada inicialmente en 30 de mayo de 2005.

Unity está disponible como plataforma de desarrollo en los sistemas operativos Microsoft Windows, OS X y Linux.

Éste motor de desarrollo permite la compilación de sus proyectos con diferentes tipos de plataformas, móviles, de sobremesa, consola, etc.

También permite el desarrollo de contenido para navegador a través de su plugin web en su lugar se utiliza WebGL (a partir de su versión 5.4.0) .

Unity tiene dos versiones: Unity Professional (pro) y Unity Personal. Para éste proyecto hemos usado la versión Personal.

[26]



Figura 4.4A. Logotipo de Unity

Historia

La aventura de Unity Technologies comenzó en 2004 cuando sus creadores, David Helgason, Nicholas Francis y Joachim Ante, decidieron dar un paso importante en su compañía de videojuegos tras el fracaso de su 'GooBall'.

Dicho juego no obtuvo las ventas que se esperaban, pero en el desarrollo del mismo habían creado unas herramientas muy potentes para su creación.

Tanto así que tuvieron la idea de crear un motor de videojuegos que pequeñas y grandes empresas pudieran utilizar por igual.

[27]

Un entorno agradable para programadores, artistas y diseñadores por igual, que llegase a diferentes plataformas sin obligar a programar el juego específicamente para cada una de ellas. Todo ello a un precio accesible para todas esas pequeñas empresas que empezaban en el mundo de los videojuegos.

Una herramienta que 10 años después se ha convertido en una de las mas populares en todo el mundo.

El éxito de Unity ha crecido sobretodo debido al enfoque en las necesidades de esos desarrolladores independientes que no pueden crear su propio motor del juego, no tienen las herramientas necesarias o no tienen el capital para adquirir licencias para utilizar herramientas de los grandes de la industria.

[27]

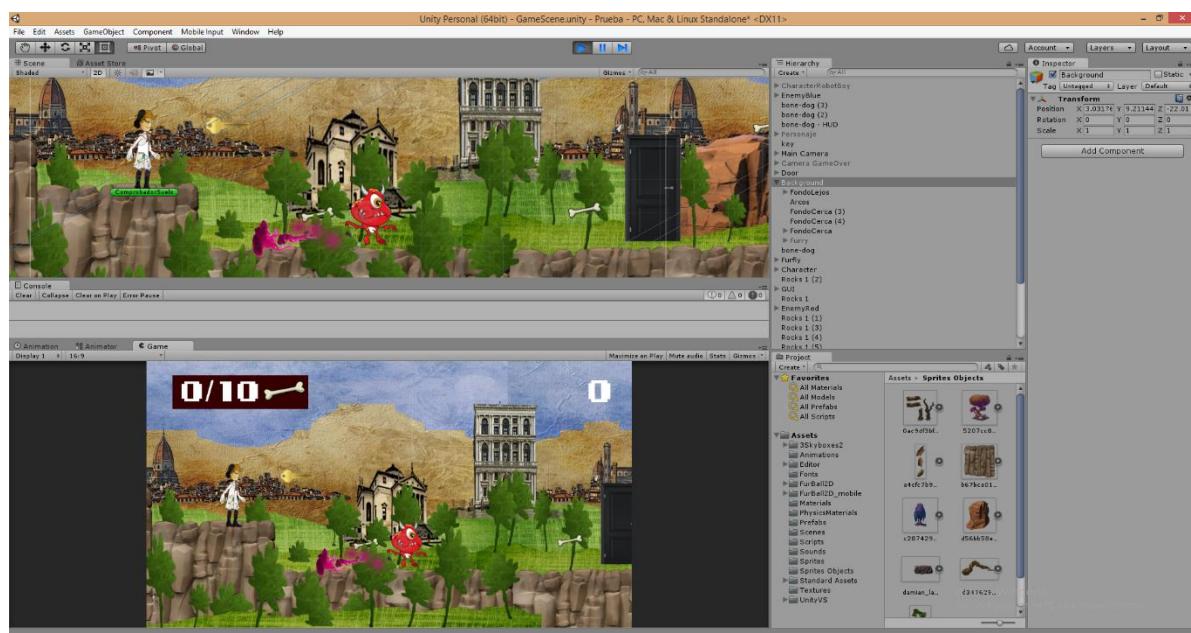


Figura 4.4B. Interfaz de Unity

El enfoque de la compañía es "democratizar el desarrollo de juegos", y hacer el desarrollo de contenidos interactivos en 2D y 3D lo más accesible posible a tantas personas en todo el mundo como sea posible.

[28]

La versión actual del motor es compatible con muchísimas plataformas (PC, Mac, Linux, iOS, Android, BlackBerry, PlayStation, Xbox, Wii, Wii U, Web...) y el sueño de desarrollar una sola vez, darle a publicar y olvidarte de problemas está más cerca que nunca. Eso sí, al final nunca es tan fácil como lo pintan y supone dedicarle el tiempo adecuado para que un juego esté pulido. Aunque de todos modos ha sido un avance respecto al modelo de desarrollo de hace unos años.

[28]

La importancia de un buen sistema de trabajo

El trabajo en equipo para desarrollar un videojuego es crucial. En el desarrollo de un videojuego deben convivir el equipo de arte, el equipo de diseño y el equipo de programación.

Tres secciones de la empresa importantísimas y repletas de profesionales que deberán dejar su ego de lado, trabajar juntos y compartir un mismo sistema de trabajo. Es ahí cuando un buen motor demuestra su solidez.

[27]

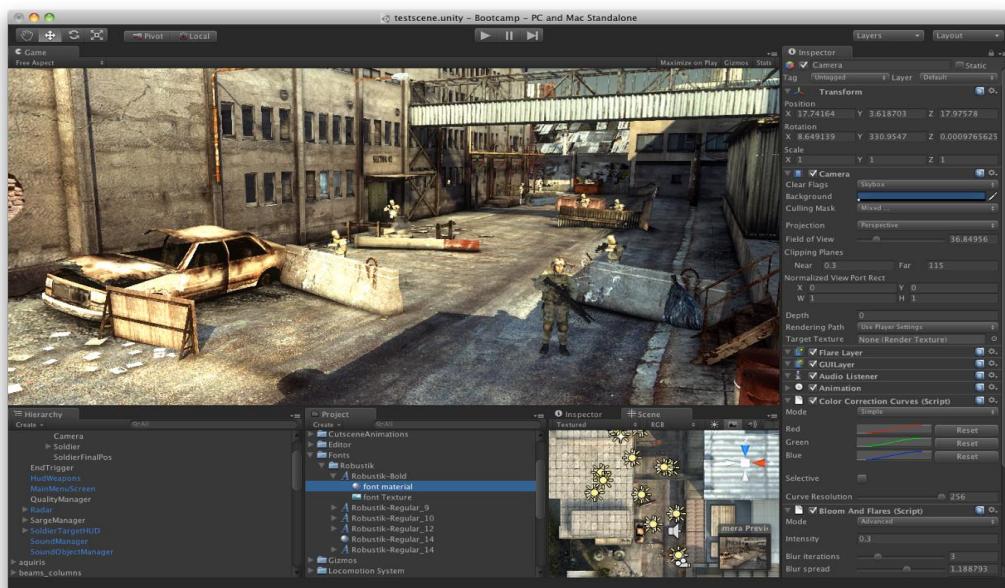


Figura 33. Creación de un juego en Unity

Es ahí en donde una solución como Unity (además de otros motores) puede ser de gran ayuda. Olvidarse de exportadores y funcionalidades básicas para centrarse en programar herramientas sólidas y diseñar niveles divertidos.

Unity - Características Principales

Unity puede usarse junto con múltiples software de desarrollo y modelado, tales como 3ds Max, Maya, Softimage, Blender, Modo, ZBrush, Cinema 4D, Cheetah3D, Adobe Photoshop, Adobe Fireworks y Allegorithmic Substance.

Un cambio realizado a un objeto creado con éstos productos, se actualizan automáticamente en todas las instancias de ese objeto durante todo el proyecto sin necesidad de volver a crear o importar manualmente.

El motor gráfico utiliza Direct3D (en Windows), OpenGL (en Mac y Linux), OpenGL ES (en Android y iOS), e interfaces propietarias (Wii). Tiene soporte para mapeado de relieve, reflexión de mapeado, mapeado por paralaje, pantalla de espacio oclusión ambiental (SSAO), sombras dinámicas utilizando mapas de sombras, render a textura y efectos de post-procesamiento de pantalla completa.

[28]

El soporte integrado para Nvidia, el motor de física PhysX, (a partir de Unity 3.0) con soporte en tiempo real para mallas arbitrarias y sin piel, ray casts gruesos, y las capas de colisión.

El scripting viene a través de Mono. El script se basa en Mono, la implementación de código abierto de .NET Framework.

Para realizar los scripts en nuestro proyecto hemos elegido utilizar C#, pero también se puede usar UnityScript o Boo.

Unity también incluye Unity Asset Server - una solución de control de versiones para todos los assets de juego y scripts, un sistema de audio construido con la biblioteca FMOD, con capacidad para reproducir audio comprimido Ogg Vorbis, reproducción de vídeo con códec Theora, un motor de terreno y vegetación , con árboles con soporte de billboarding, determinación de cara oculta con Umbra, una función de iluminación lightmapping y global con Beast, redes multijugador RakNet y una función de búsqueda de caminos en mallas de navegación.

[28]

Unity - Sistema de animaciones

Mecanim es la tecnología de animación de Unity.

Ésta tecnología está diseñada para llevar el movimiento fluido y natural de los personajes con una interfaz eficiente. Mecanim incluye herramientas para la creación de máquinas de estados, árboles de mezcla, manipulación de los conocimientos nativos y retargeting automático de animaciones, desde el editor de Unity. Herramienta que se explicará con más detalle más adelante.

[29]

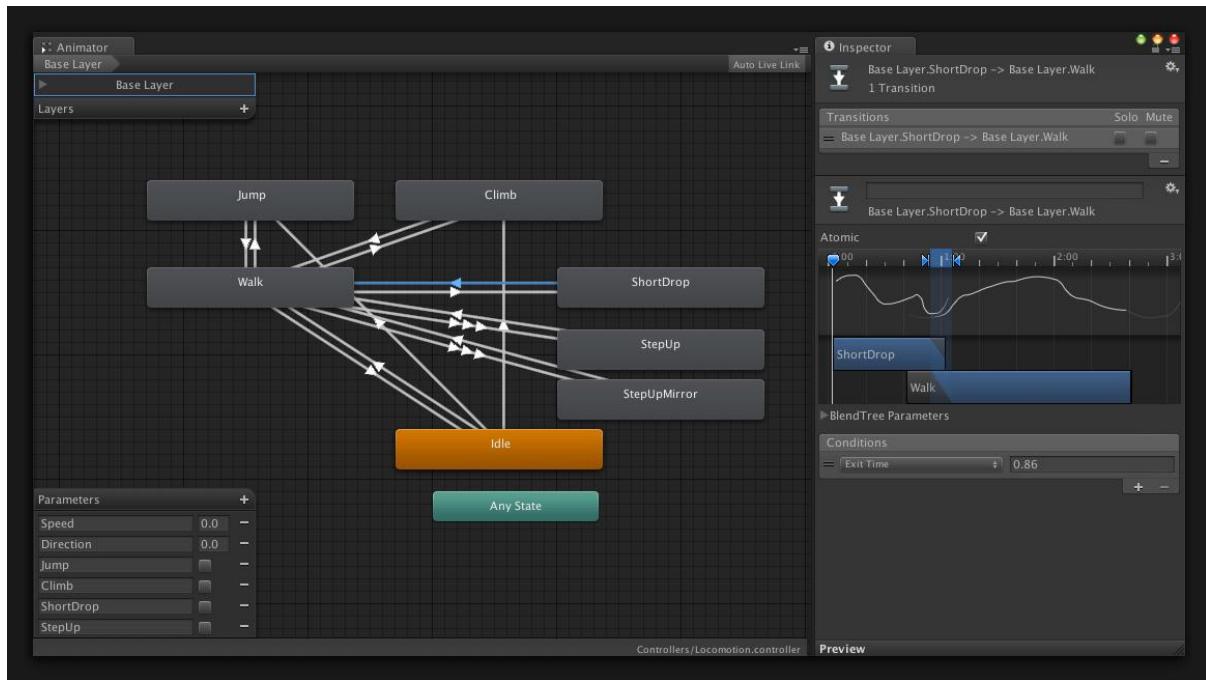


Figura 4.4C. Mechanism - Sistema de animaciones de Unity

Cada elemento en el videojuego que contenga animaciones contendrá un árbol de animaciones, donde se indican las variables y situaciones en las que puede realizar la transición de una animación a otra y viceversa.

Unity - Licencias

Hay dos licencias principales para desarrolladores: Unity personal y Unity Professional . La versión Pro tiene características adicionales, tales como render a textura, determinación de cara oculta, iluminación global y efectos de postprocesamiento. La versión gratuita, por otro lado, muestra una pantalla de bienvenida (en juegos independientes) y una marca de agua (en los juegos web) que no se puede personalizar o desactivar.

Las licencias para el desarrollo en las plataformas, PlayStation 3, PlayStation 4, PlayStation Vita, Xbox 360, Xbox One, Wii, se negocian contactando con un gerente de cuentas regional.

[30]

	PERSONAL EDITION	PROFESSIONAL EDITION
UNITY 5		
What's included		
Engine with all features	ⓘ ✓	ⓘ ✓
Royalty-free	ⓘ ✓	ⓘ ✓
All platforms (limitations apply)	ⓘ ✓	ⓘ ✓
Customizable Splash Screen	✗	ⓘ ✓
Unity Cloud Build Pro - 12 Months	ⓘ ✗	ⓘ ✓
Unity Analytics Pro	ⓘ ✗	ⓘ ✓
Team License	ⓘ ✗	ⓘ ✓
Prioritized bug handling	ⓘ ✗	ⓘ ✓
Game Performance Reporting	ⓘ ✗	ⓘ ✓
Beta access	ⓘ ✗	ⓘ ✓
Unlimited Revenue and Funding	ⓘ ✗	ⓘ ✓
Future platforms included	ⓘ ✗	ⓘ ✓
Professional editor skin	✗	ⓘ ✓
Asset Store Level 11	ⓘ ✗	ⓘ ✓
Professional Community Features	ⓘ ✗	ⓘ ✓
Source code access	ⓘ ✗	ⓘ \$
Premium Support	ⓘ \$	ⓘ \$

Figura 4.4D. Comparativa entre versión Persona y Pro de Unity

Tienen además dos planes más, que podemos consultar en su portal de adquisición:
<https://store.unity.com/es>

Unity - Asset Store

En noviembre de 2010 se lanzó el Unity Asset Store que es un recurso disponible en el editor de Unity. Más de 150.000 usuarios de Unity pueden acceder a la colección de más de 4.400 paquetes de Assets en una amplia gama de categorías, incluyendo modelos 3D, texturas y materiales, sistemas de partículas, música y efectos de sonido, tutoriales y proyectos, paquetes de scripts, extensiones para el editor y servicios en línea.

[28]

4.5 Jerarquía de objetos - Unity

En Unity los juegos se dividen en escenas. La forma más sencilla de entender qué es una escena es pensar en ellas como “niveles” de un mismo juego. Los objetos que hay en cada aparecen ordenados en una jerarquía, que es un conjunto de elementos que tienen relaciones tipo padre-hijo en la que el elemento padre es el contenedor de elementos hijo. Generalmente la jerarquía se inicia con un único objeto “Main Camera” ya que es lo único que se incluye por defecto en cualquier escena. En la jerarquía de escena se nos muestra una lista de todos los elementos que hay actualmente cargados en el editor. Esta lista se refresca dinámicamente durante el juego por lo que si creamos o destruimos objetos durante la ejecución del juego se verán aquí reflejados de inmediato.

[31]

Cada escena se compone de GameObjects: La cámara, los enemigos, el jugador, las plataformas, etc. El gameObject es la unidad básica dentro del motor. Es meramente un contenedor no son capaces de hacer nada por sí mismos. Se organizan en jerarquías.

Para que un GameObject tenga funcionalidad es necesario añadirle componentes. Cada componente le proporciona una funcionalidad distinta: física, movimiento, gráficos, daños...

Todos los GameObject tienen un componente Transform que define su posición, tamaño y rotación del elemento en 3D. Este componente no puede ser eliminado. Un GameObject tiene además la siguiente información:

- Nombre: Lo identifica de otros GameObject.
- Tag: una etiqueta que lo diferencia de otros GameObject
- Layer: una capa a la que pertenece. Utilizado por el motor de físicas.

[31]

4.5.1 Jerarquía - Personaje principal

Podemos ver en la siguiente imagen la jerarquía de objetos que hemos seguido para crear al personaje principal de la aventura.

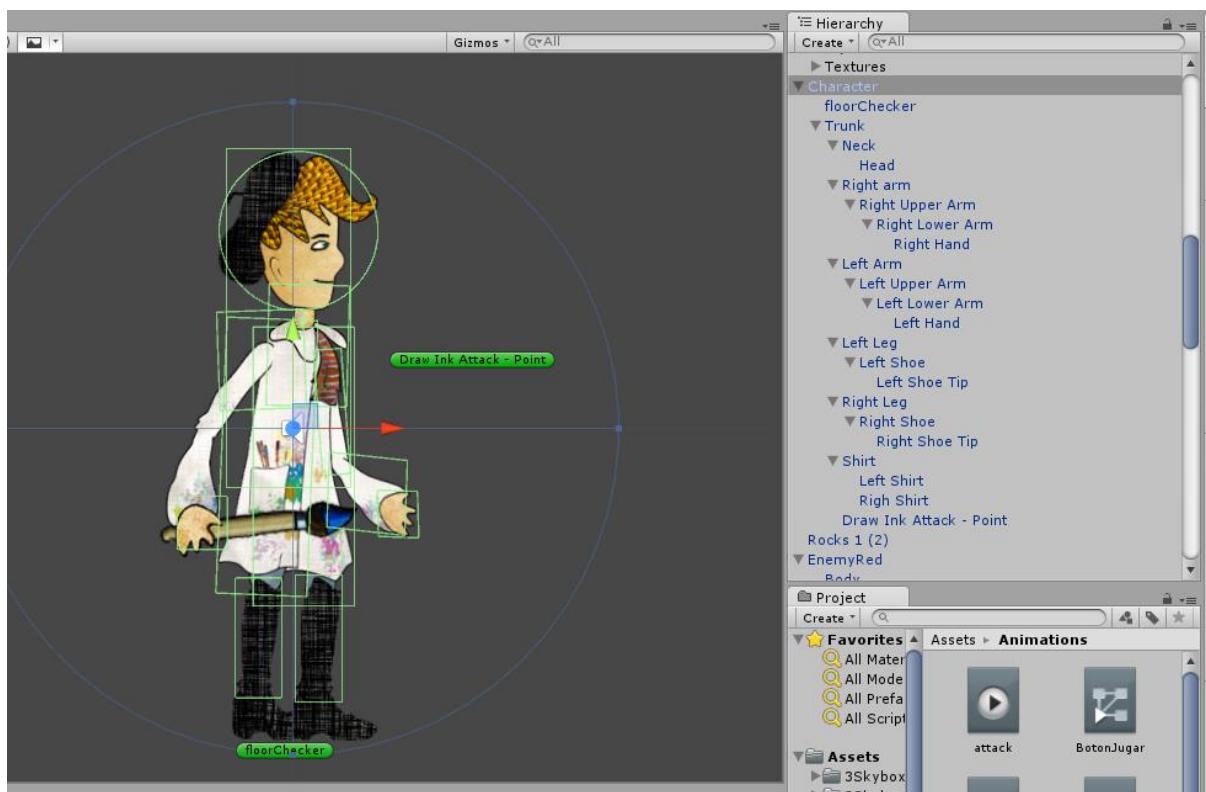


Figura 4.5. Jerarquía de objetos para el personaje principal

5. Planificación

5.1 Fases

Teniendo en cuenta que ambos integrantes del grupo no tenemos experiencia previa en el manejo de Unity, la curva de aprendizaje ha sido un elemento importante a tener en cuenta. Una vez cogido el manejo, la experimentación de mecánicas junto a la realización de diversos tutoriales también ha sido un elemento clave para desarrollar soltura.

- Fase 1: Lectura superficial de documentación de Unity.
- Fase 2: Realización de diversos tutoriales.
- Fase 3: Especificación de las tareas y planificación del tiempo a invertir.
- Fase 4: Análisis específico de cada tarea y bocetos iniciales.
- Fase 5: Recopilación de assets a usar.
- Fase 6: Implementación.
- Fase 7: Depuración de mecánicas.
- Fase 8: Documentación

5.2 Definición y estimación de tiempo para las tareas de cada fase

Las anteriores fases incluyen una lista de sub-tareas las cuales se especifican a continuación.

Lectura superficial de documentación de Unity

- Tipos de objetos.
- Estilo de programación.
- Dependencias.
- Enfoque en juegos 2D.
- Duración aproximada: 5 días

Realización de diversos tutoriales

- Tutorial de juego plataforma 2D.
- Tutorial de juego de disparos 2D.
- Tutorial de juego hack & slash 2D.
- Tutorial de controlador de personaje avanzado.
- Tutorial de objetos 3D.
- Tutorial de juego en 2.5D (elementos 3D y 2D combinados)
- Tutorial de físicas del motor Unity.
- Duración aproximada: 20 días

Especificación de las tareas y planificación del tiempo a invertir

- Objetivos
- Fases
- Tareas y tiempo a invertir
- Duración aproximada: 4 días

Análisis específico de cada tarea y bocetos iniciales

- Análisis de complejidad de tareas
- Esquemas del flujo que seguirá el juego
- Bocetos de fases y personajes implicados.
- Duración aproximada: 7 días

Recopilación de assets a usar

- Búsqueda de assets libres
- Petición a la diseñadora de assets necesarios
- Creación de assets propios adicionales
- Duración aproximada: 6 días

Implementación

- Creación de escenas
- Montaje de elementos de escenarios
- Aplicación de mecánicas referentes a escenarios (scroll parallax, iluminación, tiling, etc...)
- Creación de personajes principales
- Implementación de mecánicas del personaje principal
- Creación de enemigos
- Implementación de mecánicas de ataque de enemigos
- Creación de animaciones.
- Transiciones entre animaciones.
- Control de cámara.
- Interacción del personaje con los enemigos
- Interacción del personaje con los escenarios.
- Integración de sonidos y música.
- Creación de escenas de video.
- Duración aproximada: 40 días

Depuración de mecánicas

- Detección de errores.
- Mejora de interacciones entre elementos.
- Optimización.
- Pruebas de juego (testeo).
- Duración aproximada: 8 días

Documentación

- Documentación de estado del arte
- Documentación de historia de los videojuegos
- Documentación de épocas artísticas incluidas en el juego.
- Documentación de personajes.
- Documentación de mecánicas usadas.
- Documentación del código.
- Diagramas.
- Duración aproximada: 10 días.

5.3 Diagramas de temporización

La información acerca de la temporización queda expuesta en las siguientes figuras mediante un diagrama de Pert y otro de Gantt.

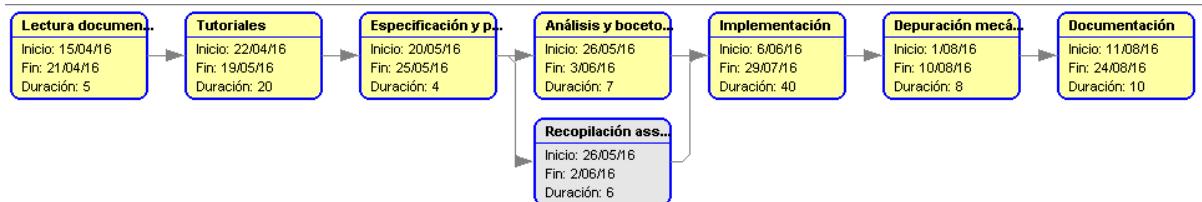


Figura 5.3A. Diagrama de Pert

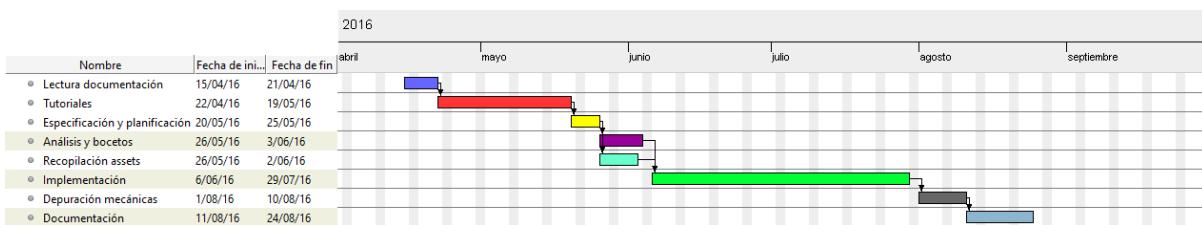


Figura 5.3B. Diagrama de Gantt

6. Implementación

6.1 Flujo de Juego

En esta sección se detalla el transcurso de una partida típica de *The Art Dimension*, desde el inicio en el menú principal hasta los créditos.

Los niveles se organizan de forma secuencial y cada vez que se completa uno, se comienza el siguiente.

El personaje comienza el nivel en el lugar determinado de respawn del nivel.

Inicio

El Jugador inicia el juego y tras la presentación de logos de introducción, se le presenta el menú principal del juego en el que aparecen dos opciones.

Si el jugador desea iniciar una partida seleccionará la opción “Play”, así, el jugador comenzará a jugar en el nivel de introducción, el Museo, donde al llegar al final del recorrido empezará a visualizar lo que supone el comienzo de la historia y el video de introducción con los sucesos en el cuadro de Las Meninas.

Tras el video de introducción empezaremos el nivel localizado en el Renacimiento.

Tras empezar el nivel, veremos como el perro avanza por su parte y se aleja corriendo, dejando detrás algunas huellas.

Éstas huellas son las que debemos seguir para encontrar a los personajes de Las Meninas. Ahora contamos con un pincel con el que podemos realizar ataques y disparar bolas de pintura.

Por el nivel nos encontraremos con diversas plataformas, entre ellas algunas se desplazan automáticamente para alcanzar puntos altos que no llegaríamos saltando. También encontraremos plataformas giratorias.

Nos encontraremos con enemigos, que nos atacaran. Por lo que debemos atacarlos, cuerpo a cuerpo o disparando con nuestro pincel para acabar con ellos.

Veremos que por el mapa podemos encontrar huesos que podemos recoger.

En cierto punto del nivel nos encontraremos el perro, que si hablamos con él nos dirá que necesitamos encontrar al personaje del cuadro perdido en el nivel y que busquemos los huesos para que él pueda seguir guiándonos hacia el final del nivel.



Figura 6.1. Final del nivel de Renacimiento

Cuando hayamos encontrado las condiciones descritas, el perro nos descifrará el camino hacia el final del nivel y la puerta a la siguiente fase.

El siguiente nivel será la fase Xilográfica. Aquí el personaje principal podrá convertirse en una criatura acorde con la ambientación, con la que podrá saltar mucho más alto que con la forma normal, consiguiendo más altura en el salto con cada rebote que realice en el suelo.

Nos encontraremos plataformas parecidas a las del nivel anterior, fusionadas con los fondos xilográficos.

Resultados

El juego finalizará una vez encontremos a todos los personajes desaparecidos de Las Meninas, ya que disponemos de vidas infinitas, y si morimos volveremos a aparecer al inicio de la fase.

6.2 Mecánicas del juego

En esta sección entraremos más en detalle sobre las mecánicas del juego. Se comentarán todos los pilares que fundamentan su jugabilidad y se detallarán las acciones que se podrán llevar a cabo por el jugador dentro de una partida normal.

Además, se ofrecerá una lista con los personajes del juego (tanto protagonista, sus transformaciones, como los enemigos), habilidades, etc.

Por último, se mostrará el mundo en el plano de movimientos, físicas y detección de colisiones.

6.2.1 Personaje Principal

Descripción: Dorian, es el personaje principal en la historia de éste proyecto, es un hombre adulto, pintor, especializado en el mundo de las artes.

Es un hombre normal, apasionado por el arte y su historia desde sus inicios, que se ve envuelto en una serie de sucesos inesperados al principio de su aventura en el videojuego.

Dorian va siempre equipado con sus pinceles, dispuesto a usarlos cuando sea necesario.

Estará equipado con su pincel mágico, el cual puede para disparar bolas de pintura a sus enemigos o al entorno, para poder desenvolverse en su andadura.

Su apariencia es típica de un artista, con su boina y camisa blanca, llena de pintura, donde guarda sus pinceles en un bolsillo.

Movimiento - Personaje principal

Para el movimiento del personaje hemos implementado un script donde se controlan todos los movimientos del personaje en los ejes X e Y, ya que es un juego 2D, aunque ciertas fases tengan elementos 3D y sensación de profundidad.

Está implementado para el uso con teclado y ratón, usándose el primero para moverse en todas las direcciones, saltar, agacharse, etc... y el segundo para apuntar y disparar en las fases en las que esto sea posible.

Par más detalles, consultar el manual de juego.

Animaciones

Se ha procedido a realizar una serie de animaciones (inactivo, corriendo, saltando, tumbado, etc...) manualmente dada una plantilla con las partes separadas del personaje y puestas juntas a posteriori mediante el Sprite Editor de Unity, y procediendo a animarlas con la herramienta de animación de Unity mediante "keyframes", es decir, definiendo en cada punto del tiempo en qué posición se encuentra cada parte del cuerpo del personaje.

He aquí el personaje despiezado:



Figura 6.2.1. Personaje despiezado

Inactivo (Idle)

Animación que se reproduce en bucle cuando estamos parados.

Las partes animadas del personaje pertenecen al tren superior de su cuerpo.

Esta animación se utiliza para dar sensación visual de que el personaje está respirando mientras está parado en estático.

Condiciones:

- Estar tocando el suelo.
- Velocidad igual a cero.

Correr

Animación que se reproduce en bucle mientras estamos corriendo erguidos.

Condiciones:

- Estar tocando el suelo
- Velocidad mayor que 0

Se podrá correr hacia adelante y hacia atrás, en la cual el personaje girará su sentido de orientación.

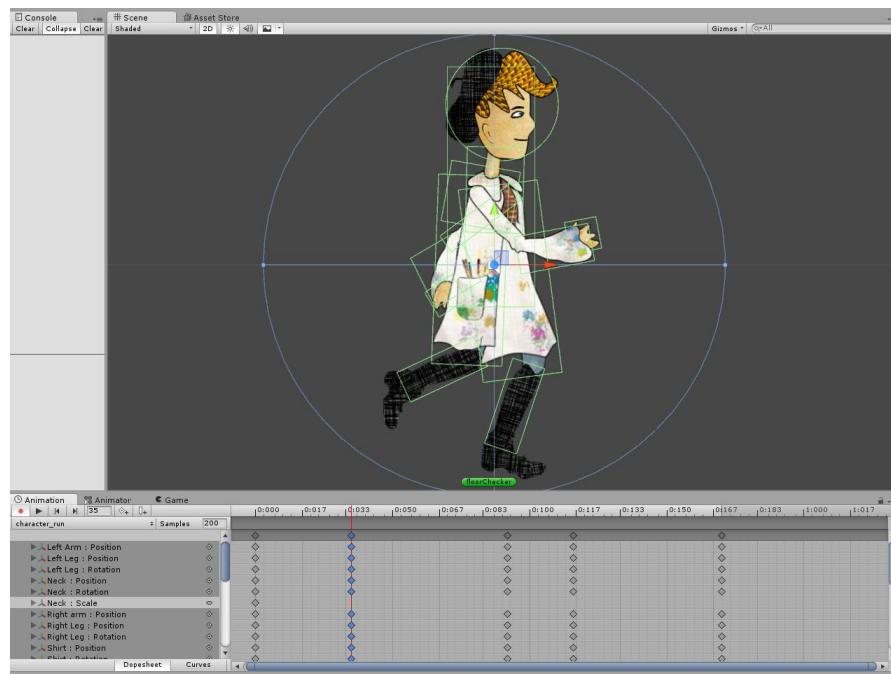


Figura 6.2.1B. Keyframe de la animación correr

Condiciones:

- Estar tocando el suelo
- Velocidad mayor que 0

Correr más rápido

Animación que se reproduce en bucle mientras estamos corriendo y pulsando el botón correspondiente para ir más rápido.

Condiciones:

- Estar tocando el suelo
- Velocidad mayor que 0
- Estar pulsando botón para correr más rápido

La animación es parecida a la anterior, pero se reproducirá más rápido.

Saltar

Animación en estático en la que el personaje eleva su posición.

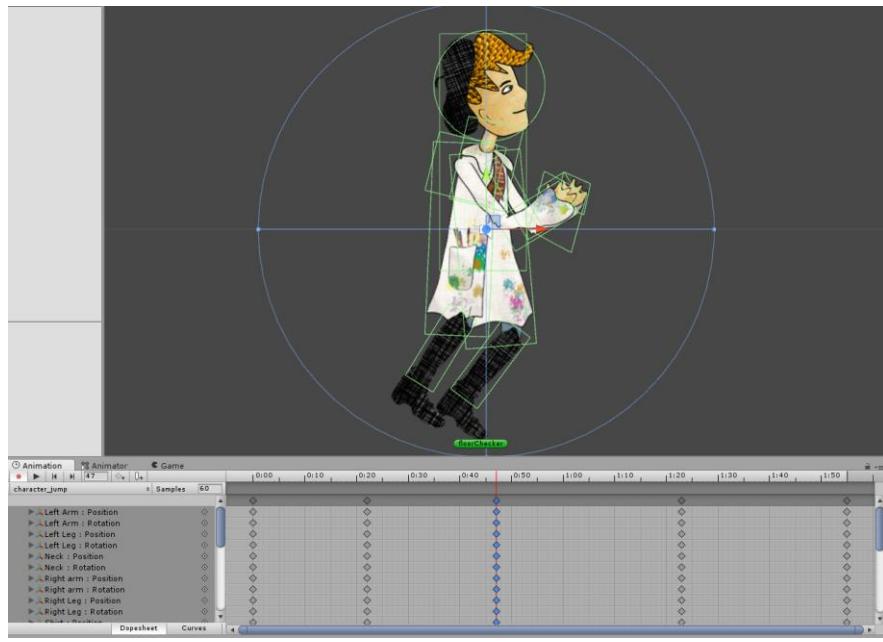


Figura 6.2.1C. Keyframe perteneciente a la animación saltar en estático

Condiciones:

- Estar tocando el suelo
- Velocidad igual a 0

Salto + Correr

Mientras corremos, podemos realizar un salto, realizando una animación de salto diferente.

Condiciones:

- Estar tocando el suelo
- Velocidad mayor que 0

Tumbarse en estático

Animación en la que el personaje está tumbado en el suelo, respirando en estático, similar a la animación idle estando erguido.

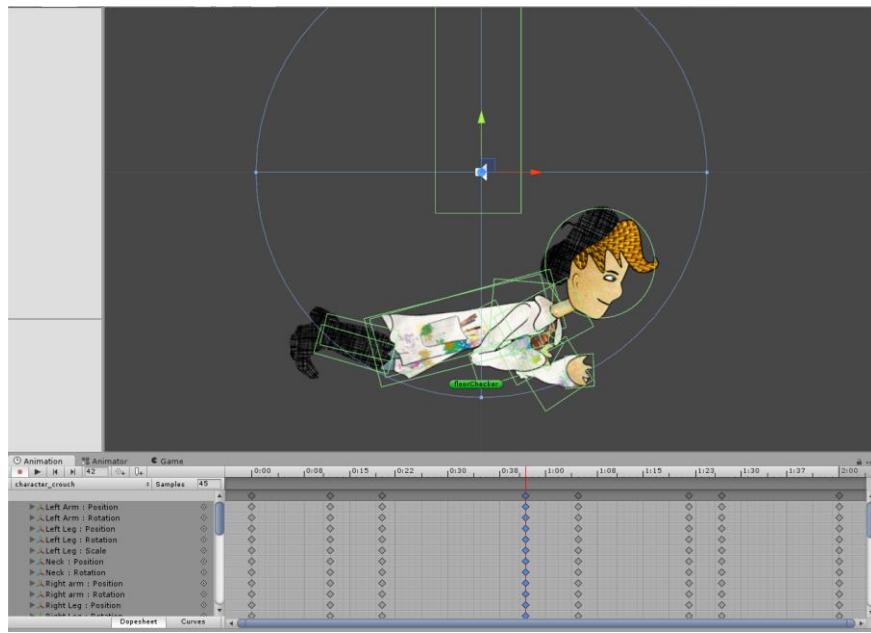


Figura 6.2.1D. KeyFrame de la animación mientras se está agachado

Condiciones:

- Estar tocando el suelo
- Velocidad igual a 0

Si se está agachado debajo de un bloque en el que el personaje no cabe erguido, no podrá levantarse hasta moverse a un lugar con espacio suficiente.

Movimiento tumbado

Animación de movimiento mientras estamos tumbados, hacia adelante o hacia atrás (girando su orientación).

Condiciones:

- Estar tocando el suelo
- Velocidad mayor que 0
- Estar tumbado

Doble Salto

El personaje podrá realizar un segundo salto mientras está en el aire y aún no ha tocado el suelo.

Condiciones:

- Haber realizado un primer salto
- No estar tocando el suelo

Ataques

Golpe

Animación en la que el personaje golpea con su pincel.

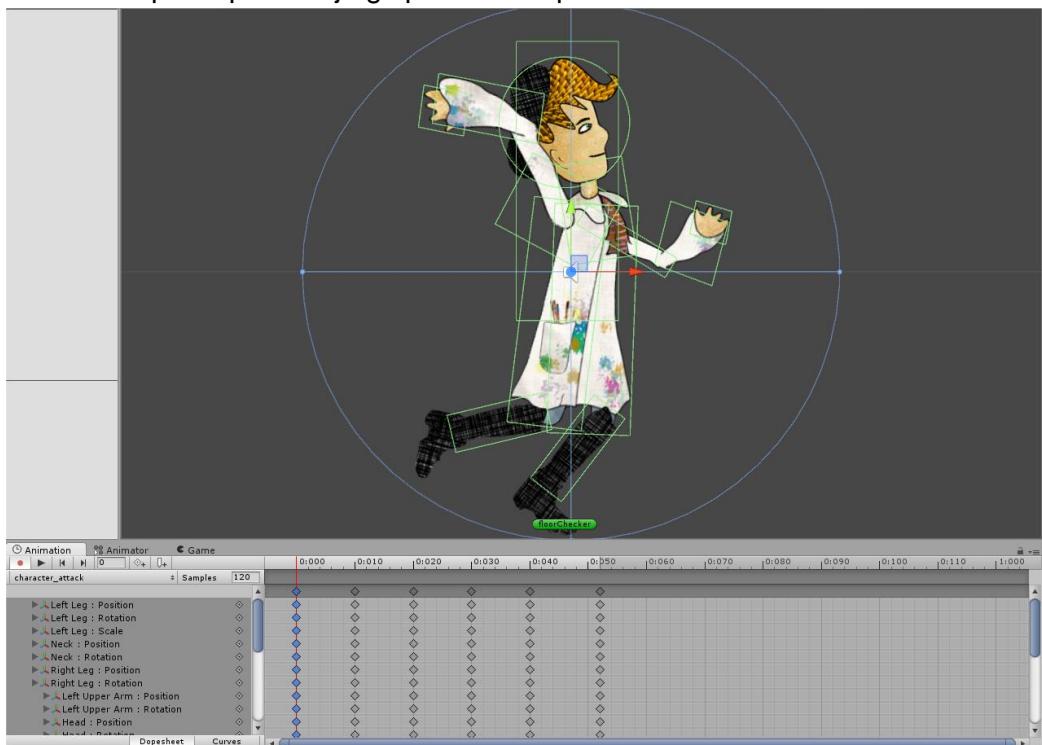


Figura 6.2.1E. Keyframe perteneciente a animación Atacar

Condiciones:

- Se puede atacar en cualquier momento, ya que ésta animación primará ante las demás en caso de peligro con enemigo.

Disparo

Animación en la que el personaje disparará una bola de pintura.

La animación es la misma que la anterior, animación de golpe.

Condiciones:

- Se puede disparar en cualquier momento.

6.2.1.1 Personaje Principal - Abstracto y Xilográfico

Para el mundo abstracto, el personaje principal hará presencia con un aspecto adecuado al nivel en el que se encuentra.

Dorian cambiará su aspecto en las fases de abstracción y xilográfica. Veremos con detalle cada uno en la sección de cada fase.

6.2.2 Enemigo azul



Es un enemigo que podemos encontrarnos en la fase renacentista, la cual veremos más adelante.

Es un enemigo que estará siempre en movimiento, si choca contra un obstáculo u otro enemigo, se dará la vuelta y seguirá su movimiento. Cada pocos segundos nos disparará una bola de pintura que el protagonista deberá esquivar si no quiere perder un corazón de salud.

Figura 6.2.2. Enemigo azul en la Fase Renacentista

Animaciones

Correr

Animación que se reproduce mientras el enemigo está activo.

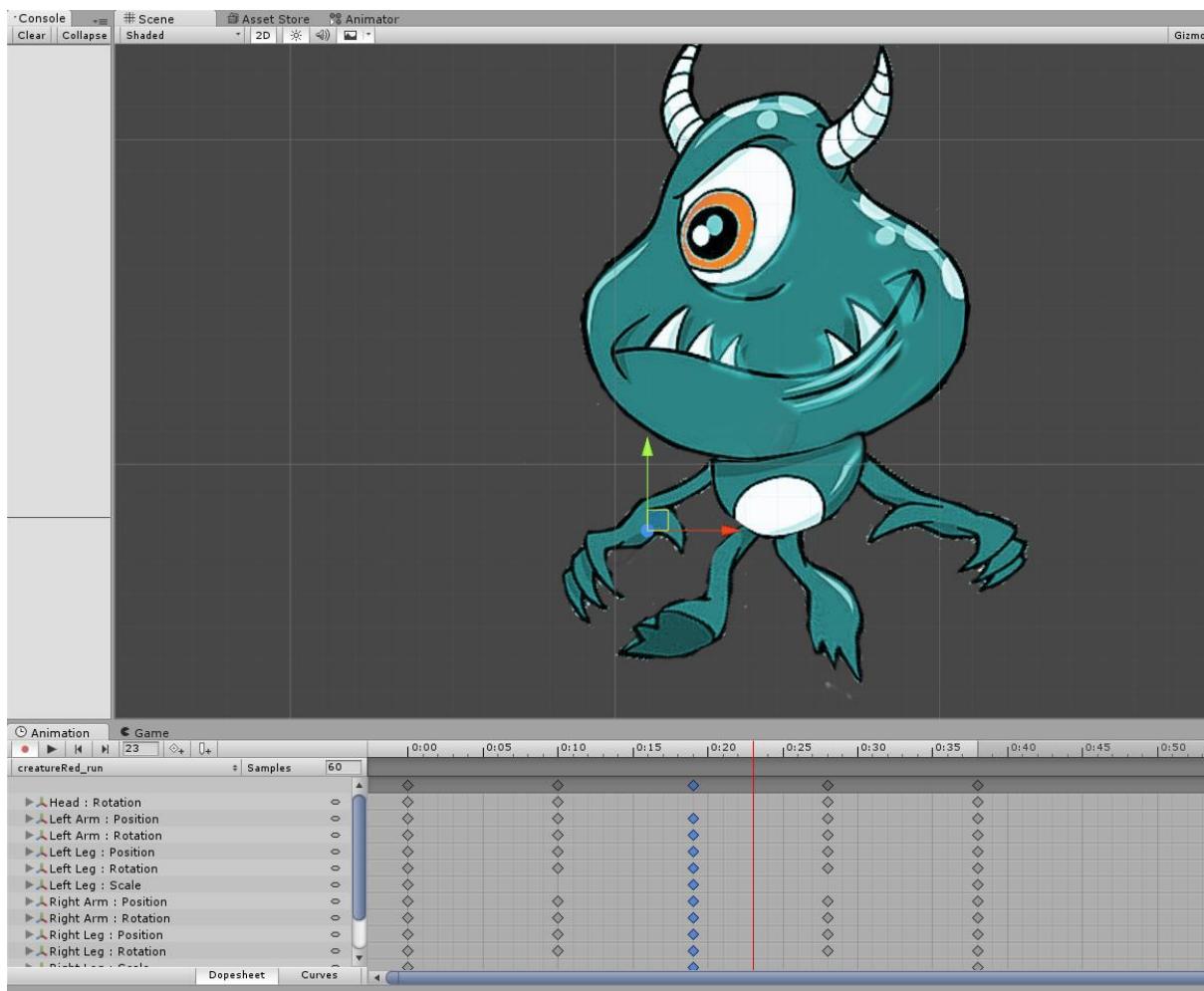


Figura 6.2.2B. Keyframe de la animación correr del enemigo

6.2.3 Transformación xilográfica - Protagonista

Este personaje se trata de la transformación que puede realizar el personaje principal en la época xilográfica.

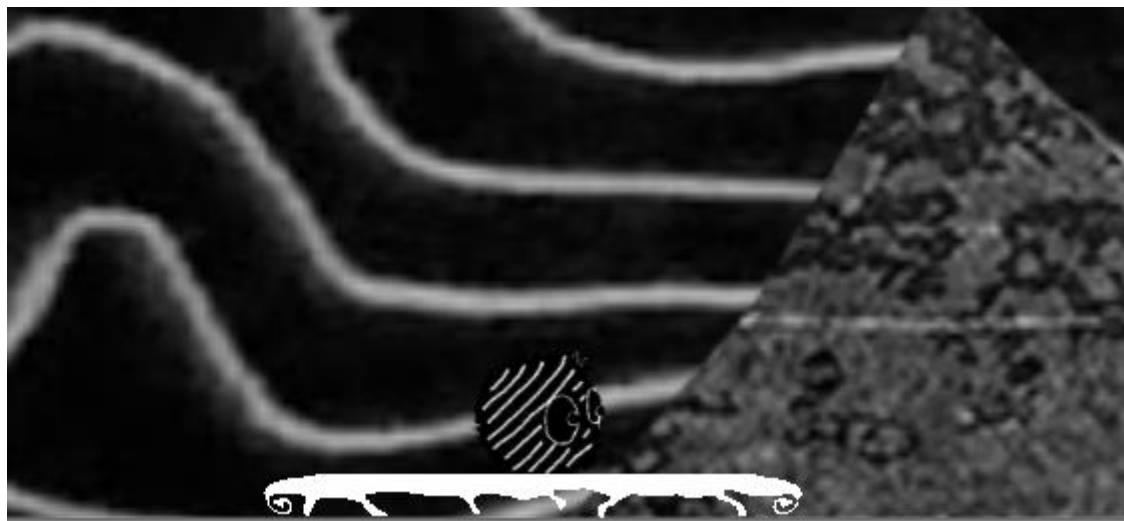


Figura 6.2.3A. Personaje manejable en la época xilográfica

Este personaje tiene un estilo de dibujo acordado a la etapa xilográfica, con sus líneas de relleno en blanco y negro.

Menos la animación en estático y de correr, las diferentes animaciones para los saltos se han realizado a base de varios keyframes, donde en cada uno las líneas blancas del cuerpo del personaje cambian de posición, para dar movimiento a la criatura mientras está en el aire.

Para la animación de salto, cuando la criatura se está elevando, se ha dibujado fotogramas del estilo siguiente:



Figura 6.2.3B. Animación de salto

Para la animación de caída en el aire, se ha realizado con líneas en el sentido opuesto.



Figura 6.2.3C. Fotograma de animación de caída

Podemos ver todas los fotogramas para animar éste personaje en la siguiente imagen:



Figura 6.2.3D. Fotogramas de animación para la criatura xilográfica

6.2.4 Perro de las Meninas

El perro nos guiará por algunas partes del juego con sus huellas para encontrar a los personajes perdidos del cuadro.

Hablar con el perro será una tarea necesaria antes de terminar alguno de los niveles, ya que él será el que nos permita seguir nuestro camino. Cuando hablemos con el perro, Dorian dirá con un diálogo que es lo que necesitamos encontrar antes de continuar el nivel.

Animaciones

Stand

Es la animación del perro en el que permanece quieto en un lugar, y podremos hablar con él.

El perro moverá parte de su cuerpo levemente dando a parecer que está respirando.

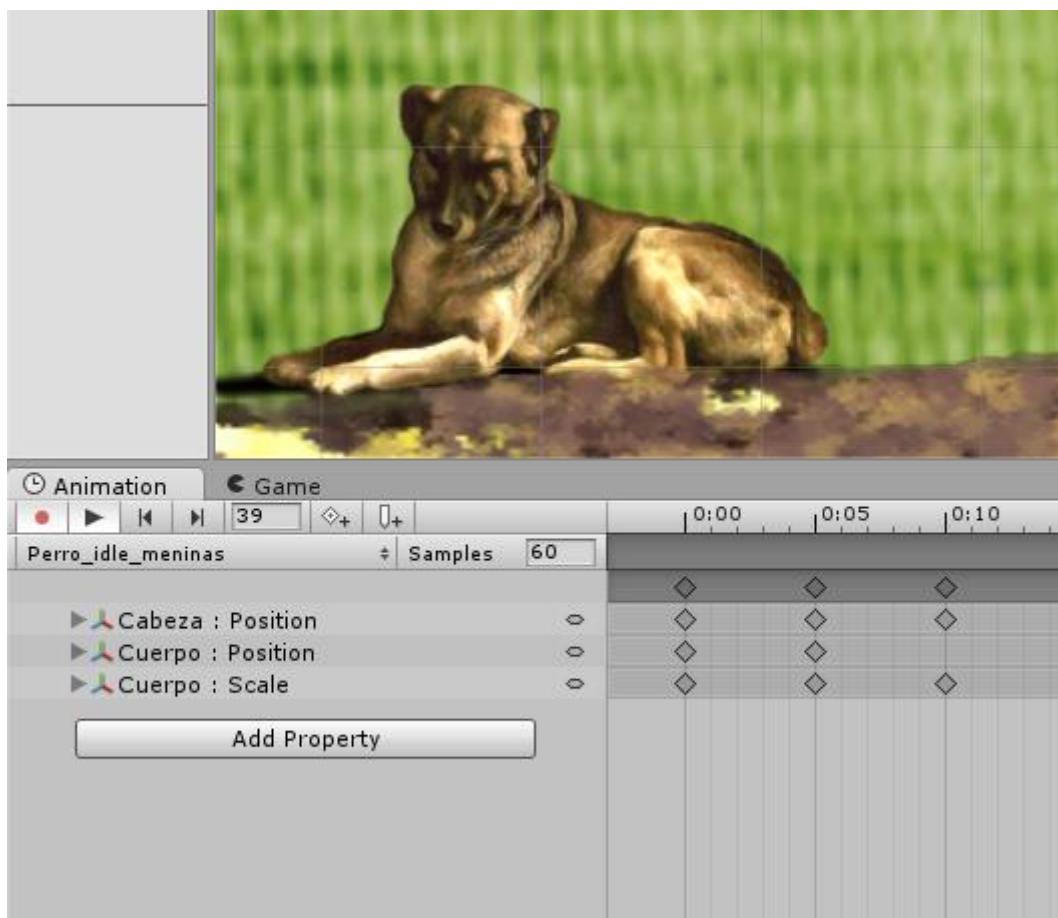


Figura 6.2.4A. Animación estática del perro

Correr

Es la animación que ejecuta el perro cuando nos está guiando el camino.

Para su realización se han recreado 9 fotogramas, extraídos del propio perro de las meninas, recortando manualmente en cada una de ellas la figura del perro.

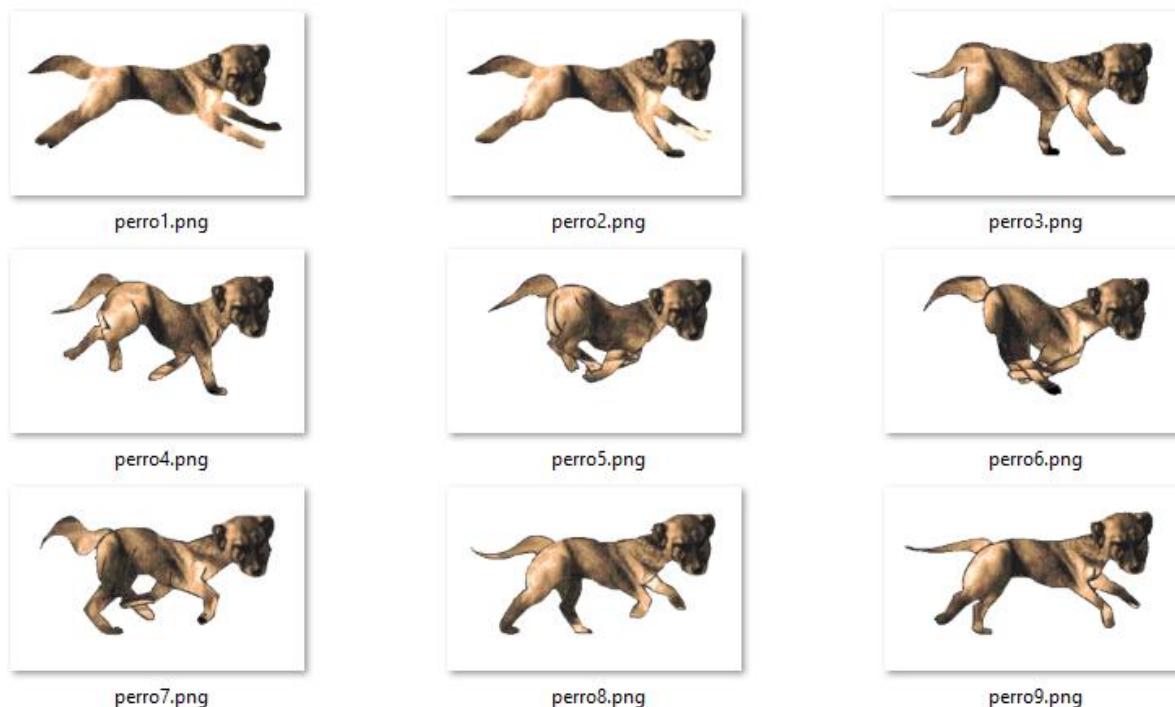


Figura 6.2.4B. Fotogramas de la animación de correr del perro

6.2.5 Enemigo volador

Este enemigo se encuentra en la fase del Renacentismo, y nos perseguirá en todo momento en modo kamikaze, es decir, si nos toca explotará, nos dañará y él morirá, pero en su lugar aparecerán otros.



Figura 6.2.5. Enemigo volador

Éste enemigo no dispone de animaciones estáticas, lo que sí hace es orientarse hacia donde nosotros estemos y venir directos hacia nosotros.

Que no tenga animaciones predefinidas no significa que no se mueva, pues lo hace constantemente, siguiendo el algoritmo A*, implementado en un paquete que nos hemos descargado (véase referencias) donde se crea un mapa de posiciones y mediante el algoritmo el enemigo encuentra la ruta óptima para llegar hacia nosotros.



Figura 6.2.5B. Ejemplo de algoritmo de búsqueda A*.

6.2.6 Animaciones - Transiciones

Fundamentos básicos de un state machine

Las animaciones comentadas en el anterior punto deben ser unidas entre sí a través de transiciones. Éstas transiciones se logran con la herramienta Animator, que nos proporciona el sistema de animaciones Mecanism de Unity.

La herramienta Animator tiene un funcionamiento de máquina de estados para organizar las animaciones.

La idea básica es que un personaje de alguna manera está vinculado en alguna acción en particular en cualquier momento. Las acciones típicas incluyen cosas como parado en estático (idle), caminar, correr, saltar, etc. Estas acciones son referenciadas como states (estados), en el sentido en que el personaje está en un “estado” donde éste camina, está saltando o cualquier acción disponible.

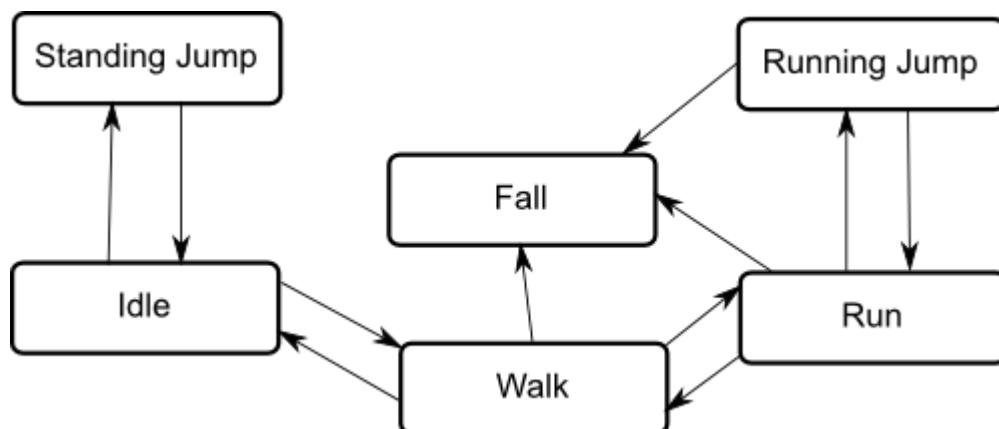


Figura 6.2.6. Estados del Animator

En general, el personaje tendrá unas restricciones para ir al siguiente estado al cual puede ir en vez de ser capaz de cambiar inmediatamente de un estado a otro. Por ejemplo, un “salto corriendo” puede solamente realizarse cuando el personaje ya está corriendo y no cuando está quieto, entonces nunca debería cambiar directamente de un estado sin actividad (idle) al estado saltando corriendo.

Las opciones para el siguiente estado el cual un personaje puede entrar desde su estado actual es referenciado como state transitions. En conjunto, el conjunto de estados, el conjunto de transiciones y la variable para acordarse del estado actual forman un state machine (máquina de estados).

Los estados y transiciones de un estado de maquina(state machine) pueden ser representados utilizando un diagrama gráfico, donde los nodos representan los estados y los arcos (flechas entre nodos) representan transiciones.

Éstas transiciones cuentan con condiciones (booleanas, comparación de valores numéricos, etc.).

Mecanim State Machines (Máquina de estados - Mecanim)

Las Maquinas de Estado de Mecanim proporcionan una manera de ver en general a todos los clips de animación relacionados a un personaje en particular y permitir varios eventos en el juego (por ejemplo el input del usuario) para activar diferentes animaciones.

- Personaje principal En nuestro personaje, su máquina de estados con Mecanism tiene el siguiente resultado:

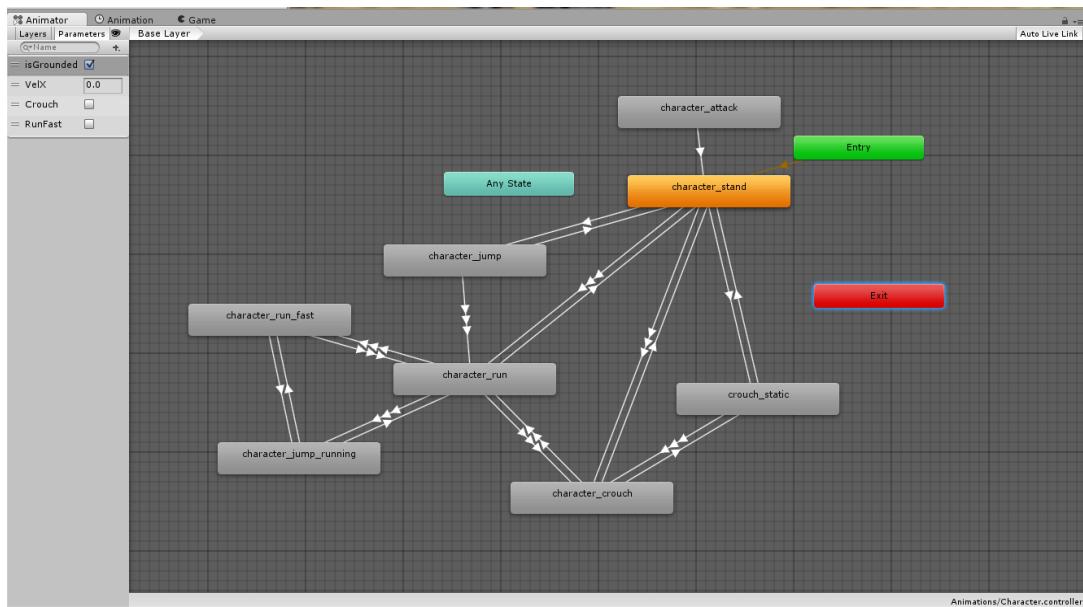


Figura 6.2.6B. Máquina de estados

Podemos apreciar en la imagen que la actividad del personaje comienza con la animación “character_stand”.

A partir de ese estado, podemos ver todas las posibles transiciones a las siguientes animaciones. Los arcos (transiciones) entre estados traen consigo una serie de condiciones (que funcionan entre sí con la lógica AND).

En caso de que entre un estado y otro queramos condiciones OR debemos aplicar un bloque distinto de condiciones en el mismo arco, como es el caso en los arcos en que vemos 3 flechas en la misma línea.

En el caso del arco de transición seleccionado en la siguiente imagen podemos ver a la derecha uno de los dos bloques de condiciones seleccionado (el primer elemento seleccionado en “Transitions”) con sus condiciones.

Én este ejemplo, estamos contemplando la transición entre el estado estático (parado -> character_stand) a la animación de correr (character_run). Y las condiciones de ése primer bloque son: que la velocidad del personaje sea mayor que 0 y que esté tocando el suelo.

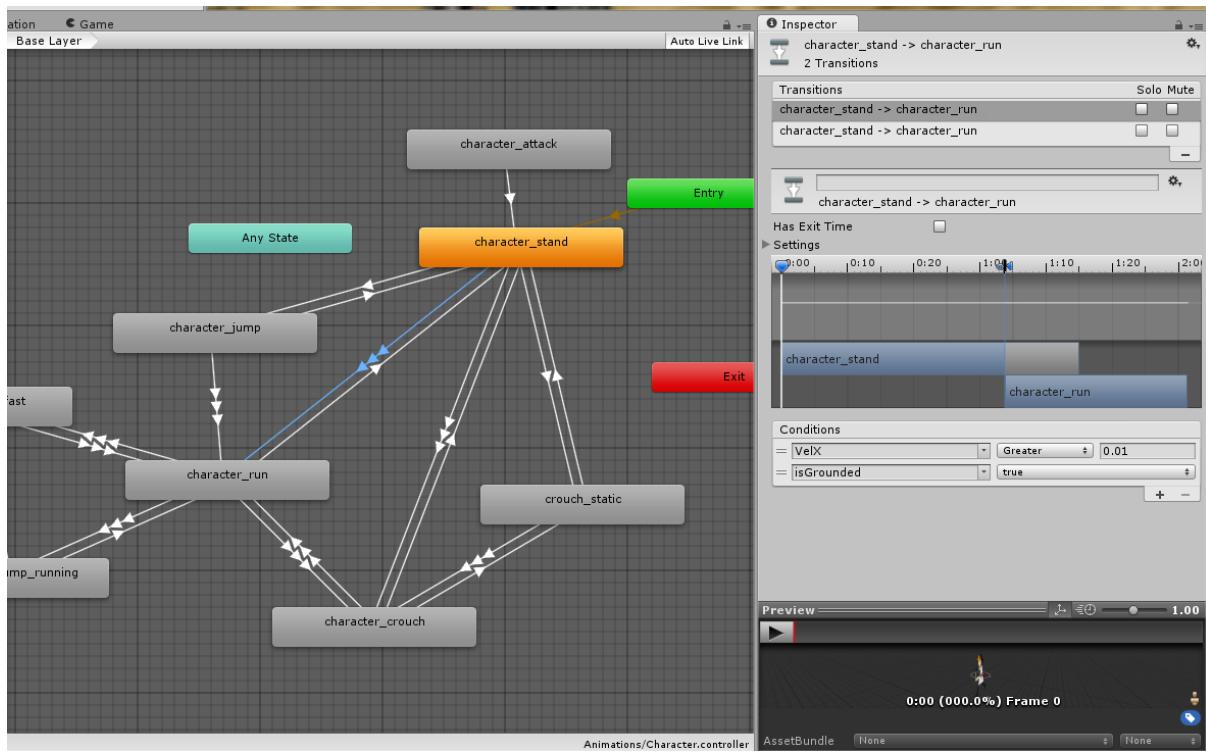


Figura 49. Ejemplo de transición entre dos animaciones

Para el segundo bloque de condiciones que no se muestra, la diferencia es que la velocidad sea menor que 0 en vez de mayor, contemplando así el movimiento del personaje hacia la derecha y a la izquierda.

6.2.7 Escenarios

Efecto Parallax Scrolling

Lo que en español sería desplazamiento con paralelaje, ésta técnica se utiliza para engañar a nuestro ojo para apreciar profundidades 3D en un renderizado 2D.

Su funcionamiento es simple, se basa en un principio de óptica en el que los objetos más cercanos sufren una desviación angular superior a los lejano. Es decir, los objetos que están en primer plano se desplazan más rápido que los que están en el fondo.



Figura 6.2.7. Implementación Scroll Parallax con Unity

Su implementación, pues, se realiza dividiendo las imágenes background por capas, clasificándolas en cercanas, distancia media y la capa lejana. Las de distancia medias son opcionales, pero dan una aún mejor sensación de profundidad.

Nuestro código para implementar esta técnica ha sido el siguiente:

```
previousCameraPosition = transform.position;

parallaxScales = new float[backgrounds.Length];
for (int i=0; i< parallaxScales.Length; i++)
{
    parallaxScales[i] = backgrounds[i].position.z * -1;
}

void LateUpdate () {
    for (int i=0; i<backgrounds.Length; i++)
    {
```

```

        Vector3 parallax = (previousCameraPosition -
transform.position) * (parallaxScales[i] / smoothing);
        backgrounds[i].position = new
Vector3(backgrounds[i].position.x + parallax.x, backgrounds[i].position.y
+ parallax.y, backgrounds[i].position.z);
    }
    previousCameraPosition = transform.position;
}

```

Ésta función (LateUpdate) se llama automáticamente antes de dibujar la cámara. Tendremos un vector backgrounds, que contendrá tantas capas (cercanía y lejanía) como queramos.

Esas capas, como vemos en la captura anterior, se colocan en el eje z, a distinta distancia, tanto como queramos que sea mayor el efecto scroll parallax.

Lo que hace el algoritmo es editar la posición de dibujado de las imágenes del vector, según la posición propia de la cámara y el valor del eje Z de cada una. (contenidos en parallaxScales).

Por lo tanto, cuanto más lejos o más cercano coloquemos una capa del fondo, más lento o más rápido irá acorde a su eje Z, teniendo como punto de referencia la cámara principal de renderizado.

En el video de la referencia [22] podemos ver un ejemplo visual sencillo del uso de ésta técnica.

6.2.8 Fases

En ésta sección hablaremos sobre los diferentes niveles de juego, su contexto en cuanto al personaje

6.2.8.1 Video de Introducción

Hemos realizado un video inicial nada más arrancar el juego, donde se muestra el logotipo de la UGR, los nombres de los creadores y se narra brevemente la introducción al argumento del juego.

El video ha sido montado usando Sony Vegas, montando el video y exportándolo a Unity, donde es necesario crear un objeto cualquiera, en este caso un plano, y añadir el video como textura de video.

Además es necesario añadir el componente de audio de forma separada a algún objeto de la escena e indicar que se reproduzca al inicio de la escena. Hecho esto, además de los cambios de escena implementados, tendríamos nuestro video montado en una escena independiente de Unity.

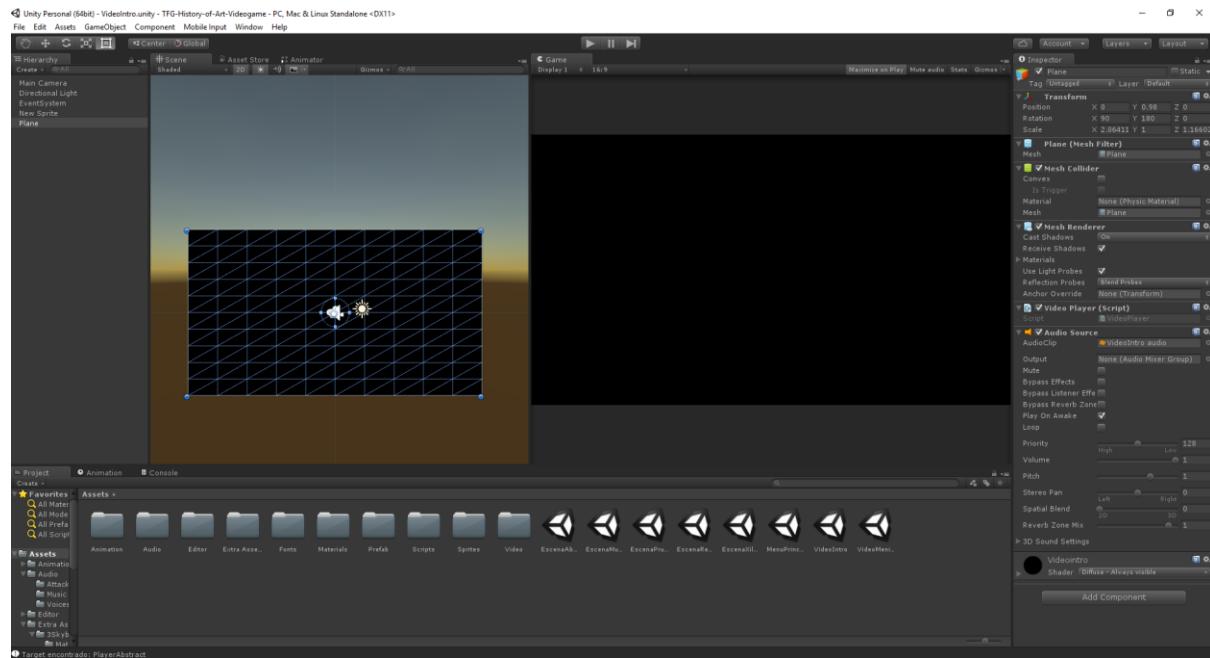


Figura 6.2.8.1. Plano con textura de video asignada.

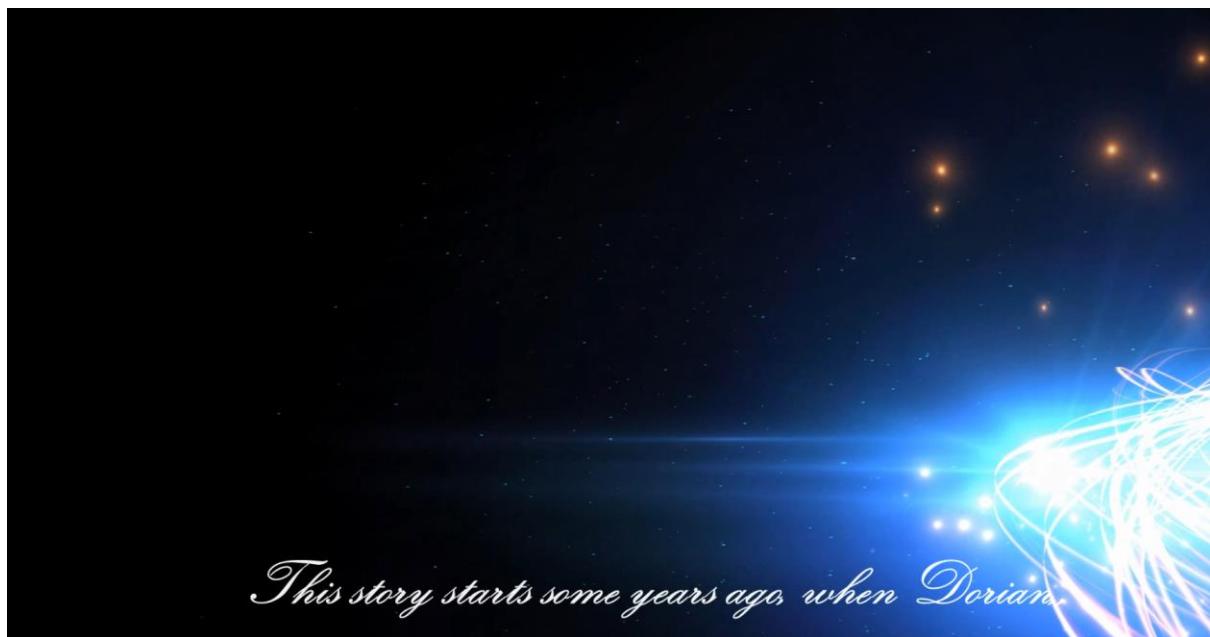


Figura 6.2.8.1B. Captura del video inicial

6.2.8.2 Menú Principal

Pantalla principal del juego , en la que se puede ver el título del mismo y referencias sobre lo que veremos durante la historia y niveles del juego: renacimiento, estilo xilográfico y abstracción, con toques de pintura.



Figura 6.2.8.2. Menú inicial

Tenemos dos opciones en el menú, “Play” para empezar a jugar y “Quit” para cerrar el juego.

6.2.8.3 Introducción del juego - Museo

Aquí nos encontraremos con el primer momento en el que empezamos a manejar a nuestro protagonista.

El lugar será el museo donde comienza la historia, el personaje empezará colándose de incógnito con el objetivo de poder contemplar algunas de las obras que el museo alberga y escapar de nuevo de allí.

El personaje caminará por el museo, en el que podremos ver de fondo varias obras colgadas en la pared.

Durante el recorrido del personaje, se irá pasando por delante de diversos cuadros, sobre los cuales el personaje soltará algunos comentarios.

Para estos comentarios, además de escuchar a la voz del protagonista veremos una pantalla de diálogo con la foto de nuestro personaje y el texto mostrando a modo de subtítulos el audio escuchado.



Figura 6.2.8.3. Primer diálogo de nuestro personaje

El **diálogo** mostrado en la anterior captura con un objeto canvas, con 3 objetos del mismo tipo por debajo en su nivel de jerarquía.

El Canvas es un área donde los elementos UI deben estar.

El Canvas es un Game Object con un componente Canvas en él, y todos los elementos UI deben ser hijos de dicho Canvas.

Para éste caso, los elementos UI, hijos del canvas de diálogo, son dos objetos para texto (nombre del personaje que habla y otro para el texto del diálogo) y un objeto imagen con la foto del personaje. Los tres, a su vez, contienen un componente canvas.

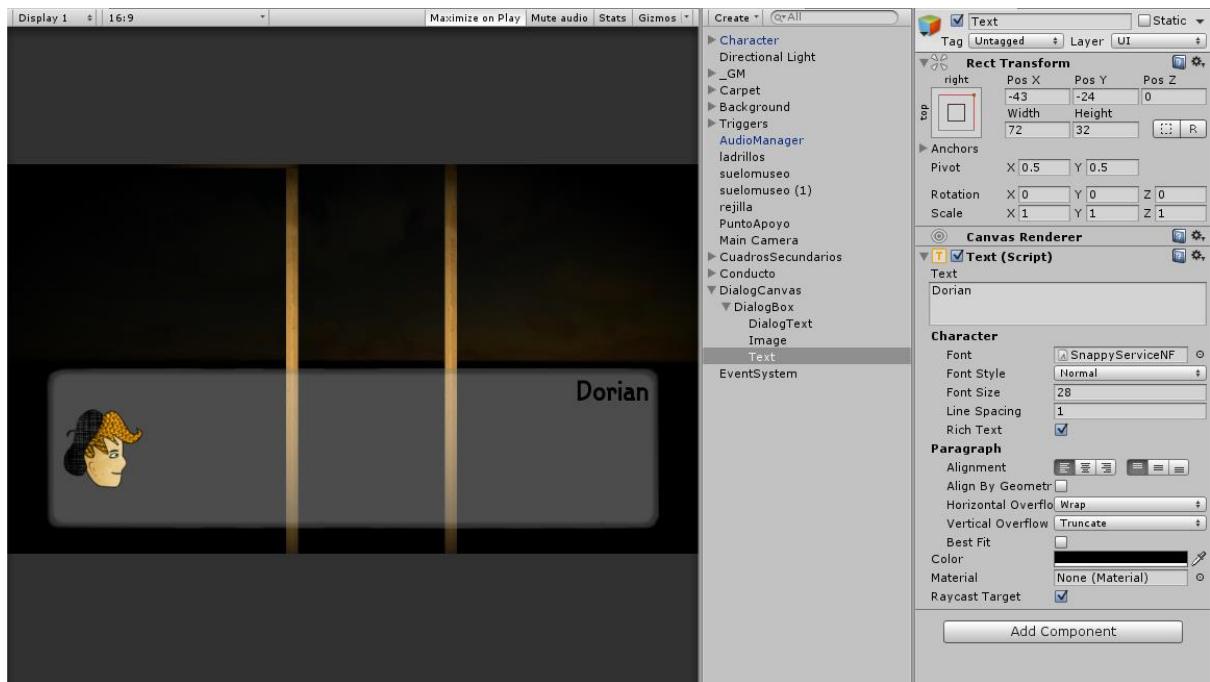


Figura 6.2.8.4. Objeto DialogBox

En la captura anterior podemos observar el **objeto DialogBox**, contenido como hijos a los 3 objetos nombrados. A la derecha podemos ver la configuración del objeto text para el nombre del personaje. Y a la izquierda podemos ver la preview en la escena del juego.

Para ésta fase, para mostrar en pantalla los diálogos y reproducir los comentarios del personaje hemos creado unos triggers, utilizando para ello unos colisionadores, colocados estratégicamente por el escenario, para que se reproduzcan cuando el personaje camina sobre esos puntos.

Como vemos en el siguiente mapa, usamos un objeto (invisible mientras jugamos), que contiene un BoxCollider2D, que es una zona por la que podemos controlar si está colisionando con el jugador. Y en caso de colisión lanzar el script que tenemos para el diálogo.

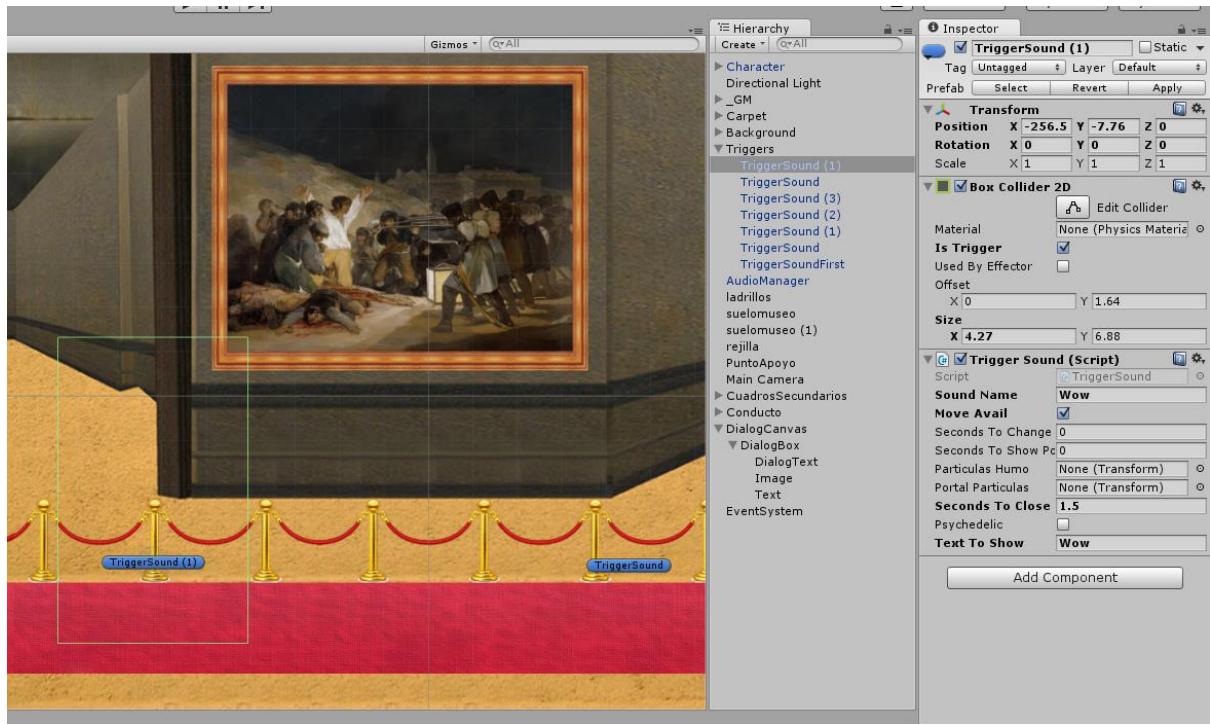


Figura 6.2.8.5. Ejemplo de BoxCollider2D

Como vemos en la imagen, el objeto con el BoxCollider2D contiene un script llamado "Trigger Sound" con sus variables públicas visibles desde Unity.

En este script le podemos indicar el nombre del audio a reproducir, el texto a mostrar en el diálogo y el tiempo que tardará en cerrarse, entre otras opciones.

Entre el código de dicho script, activaremos el canvas de diálogo (que por defecto estará desactivado), activamos sus hijos con los parámetros adecuados, se ejecutará el audio seleccionado y después de los segundos indicados, se desactivaran la vista del diálogo para que desaparezca de la pantalla.

Al final del nivel, nos encontraremos con el cuadro de Las Meninas, que dará lugar al comienzo paranormal de la historia.



Figura 6.2.8.6. Efecto de aberración cromática.

Como vemos en la imagen anterior, empezaremos a ver un efecto psicodélico en la pantalla, dando pie a pensar de que algo raro está comenzando a suceder.

Éste efecto se consigue desde el script “Trigger Sound” nombrado antes, utilizando el componente VignetteAndChromaticAberration de la cámara, y asignándole un valor de aberración cromática.

El código es simplemente la siguiente línea:

```
if (psychedelic)
{
    Camera.main.GetComponent<VignetteAndChromaticAberration>().chromaticAberration
= 50f;
}
```

A continuación, mientras el personaje se está preguntando qué es lo que está sucediendo en el cuadro, hemos colocado unos objetos con sistema de partículas para simular que se está abriendo un portal paranormal desde el cuadro hacia el personaje.

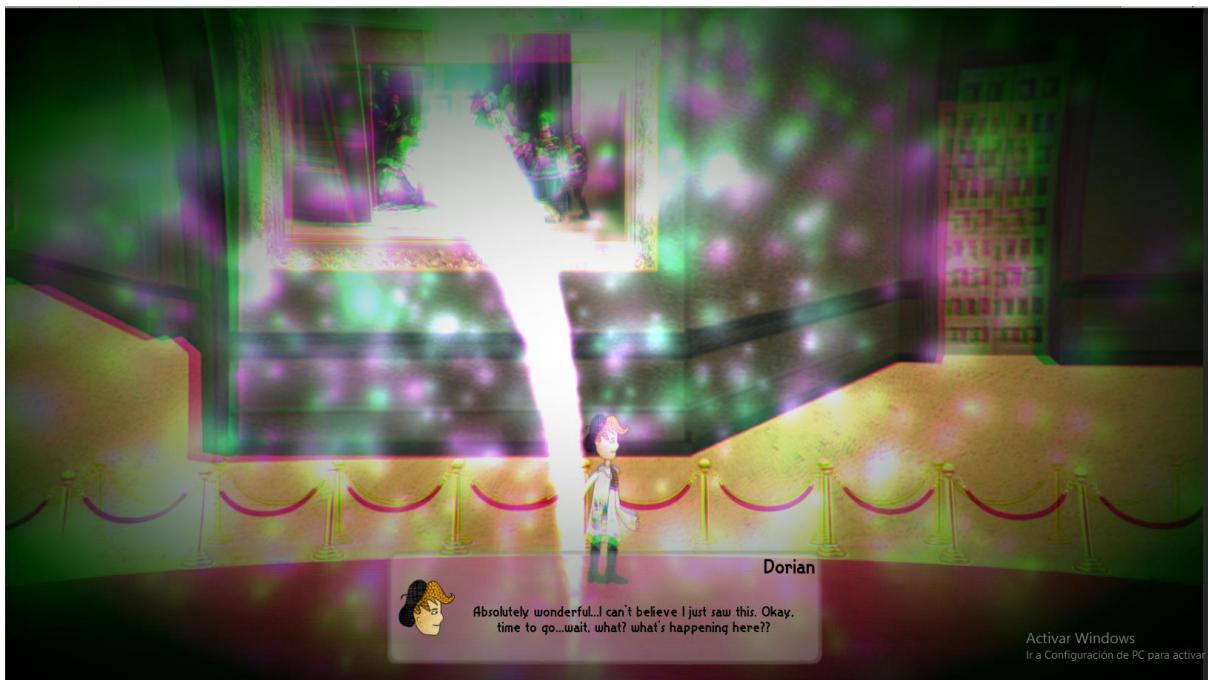


Figura 6.2.8.E. Efectos de iluminación

Después de dicha escena, daremos paso al video de introducción.

6.2.8.4 Video de Introducción

Este video aplica las mismas reglas que el video de introducción en cuanto a montarlo en Unity, y también ha sido creado en Sony Vegas y exportado previamente.

Lo que muestra este video es lo que sucede después de que nuestro protagonista, Dorian, se sitúe en frente del cuadro de Las Meninas, donde todo empieza a cobrar un sentido algo psicodélico y extraños sucesos pasan.

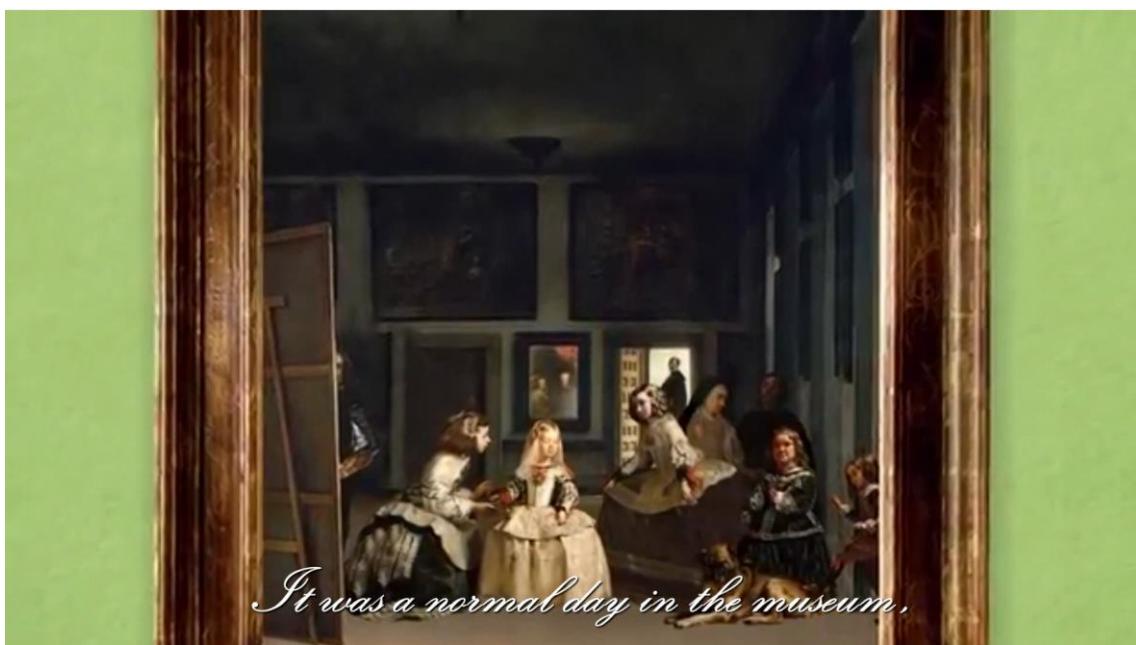


Figura 6.2.8F. Escena del video de introducción

Se muestra pues una escena donde los personajes del cuadro se dan cuenta de la presencia de Dorian y empiezan a desaparecer del cuadro, excepto Velázquez, el creador, y el perro que aparece en el cuadro.



Figura 6.2.8.G. Escena con el cuadro vacío

Tras esto Velázquez se dirige directamente a nosotros encomendándonos la tarea de ir a buscar a los personajes perdidos a través del mundo artístico, desde dentro del mismo. Es desde aquí desde donde realmente empieza la aventura.



Figura 6.2.8.H. Velázquez se enoja tras el suceso.

6.2.8.5 Época clásica: Renacimiento

En esta fase nos encontramos inmersos en un mundo donde es claro ver que la temática es renacentista, visible en los fondos y sus respectivos diseños.

Ésta fase tiene un enfoque de juego 2D clásico, plataformeo con estilo de corte clásico, enemigos y recolección de objetos, algunos de los cuales están algo escondidos .

Nuestro protagonista empezará en el nivel, observando como el perro empieza a correr siguiendo el rastro de un personaje del cuadro de Las Meninas, dejando unas huellas detrás para que podamos seguirlo.

El personaje perderá al perro de vista, el cual nos lo encontraremos más adelante.

Se mostrará un diálogo indicativo del objetivo actual del protagonista, además de escuchar en ese momento los ladridos del perro mientras se aleja. Éste diálogo sigue el modelo de la fase anterior, con un collider para lanzarlo en pantalla.



Figura 6.2.8.I. Comienzo de la fase renacentista

Podemos ver en la esquina superior que se encuentran 3 corazones, los cuales indicarán la vida del protagonista.

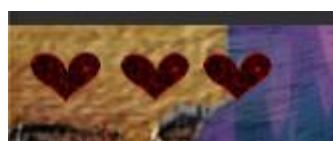


Figura 6.2.8.J. Salud del personaje

Al comienzo podremos ver también una especie de insecto negro con alas, una criatura que como veremos no es agresivo y no nos atacará, simplemente estará pululando por el escenario. Si lo tocamos veremos que podemos empujarlo y desplazarlo.



Figura 6.2.8K. Insecto inofensivo.

Un poco más adelante veremos un enemigo, el cual nos disparará bolas de pintura. Si el enemigo nos toca o nos golpea con una bola de pintura, perderemos uno de los corazones de salud.



Figura 6.2.8L. Enemigos en la fase Renacentista

Si el personaje llega a 0 corazones de salud, morirá y comenzará de nuevo en la zona de respawn, que suele ser el comienzo de nivel.

Durante el nivel veremos algunas zonas en las que el personaje debe arrastrarse por el suelo, para poder llegar a ellas.



Figura 6.2.8M. Personaje arrastrándose por el suelo

Se ha tenido en cuenta que el jugador suelte el botón de mantenerse agachado mientras está debajo de un bloque/obstáculo, para así mantener al personaje acostado y no permitir que se levante debido al conflicto de colisión con el obstáculo.

Para ello, se ha colocado un espacio de colisión en el objeto padre en la jerarquía del personaje, éste objeto tendrá un objeto BoxCollider2D, que lo utilizaremos para comprobar si está colisionando con un objeto.

En caso de colisión, el personaje se mantendrá acostado en el suelo, hasta que se salga de la zona de colisión y pueda levantarse.

```
void OnTriggerEnter2D(Collider2D collider)
{
    if (collider.gameObject.CompareTag("Obstacle"))
    {
        can_uncrouch = false;
    }
    else can_uncrouch = true;
}
```

En ésta fase, nos encontraremos con plataformas móviles, sobre las que nos podremos montar, y nos llevará a lugares que no podremos llegar con el salto del personaje, o hacia lugares donde se encuentran escondidos algunos los colecciónables de la aventura (huesos para el perro).

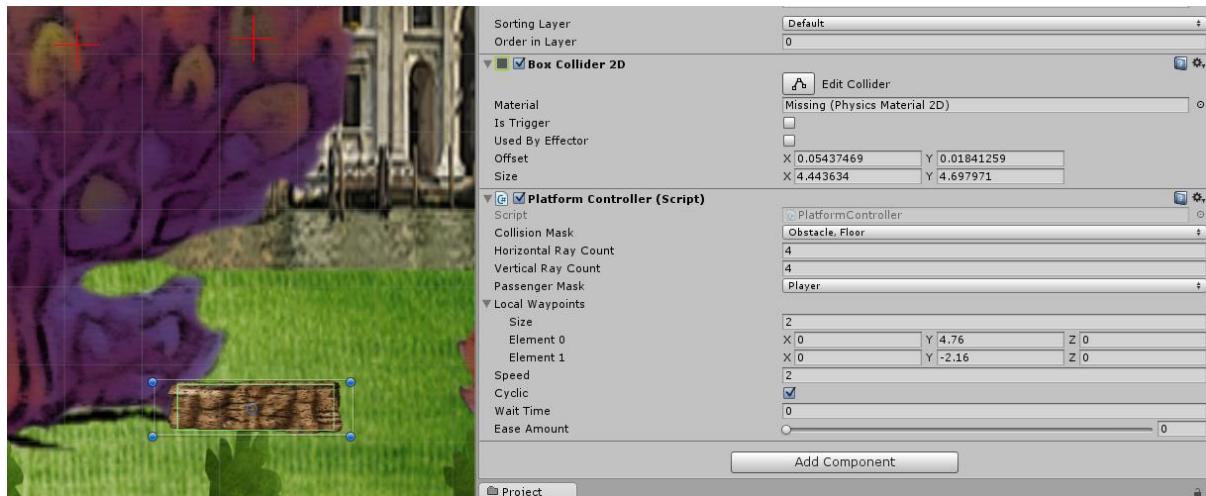


Figura 6.2.8N. Script de plataforma móvil

En éste script se han definido dos puntos (local Waypoints) que son los límites por los que se desplaza la plataforma.

Sobre la mitad del nivel se ha colocado una cueva, en la que mediante un script por colisión, cuando entremos dentro se nos mostrará el interior, en el cual se encontrará la Menina que buscamos en este escenario.



Figura 6.2.8O. Vista de cueva exterior en la fase renacentista

Para mostrar el interior, hemos usando la siguiente función:

```
public GameObject[] inner_parts;

// Use this for initialization
```

```

void Start () {

}

void OnTriggerEnter2D(Collider2D collider)
{
    if (collider.tag == "Player")
    {
        set_active_inner();
    }
}

void set_active_church_inner()
{
    foreach (GameObject element in inner_parts)
    {
        element.SetActive(true);
    }
    gameObject.SetActive(false);
}

```



Figura 6.2.8P. Interior Cueva con una de las Meninas perdidas

Más adelante, se ha apliado la misma técnica a una iglesia, mostarndo su interior si entramos dentro.



Figura 6.2.8Q. Exterior Iglesia - Fase Renacimiento

En la imagen anterior podemos ver el exterior de la iglesia, y en la siguiente imagen su interior, con algunas plataformas móviles:



Figura 6.2.8R. Interior Iglesia - Fase Renacimiento



Figura 6.2.8S. Zona de plataformeo con plataformas móviles

Se han añadido también plataformas giratorias, con las que se han creado varias zonas de saltos con coordinación, para acceder a colecciónables escondidos.

Éstas plataformas giratorias se han realizado creando animaciones para las plataformas. Modificando su posición de giro en cada frame de las mismas.

Cuando hayamos encontrado los huesos necesarios y a la Menina perdida en éste nivel, podremos ver de nuevo al perro al final del recorrido, que nos guiará hacia la puerta donde se encuentra el portal a la siguiente época.

En caso de no haber encontrado las meninas, los huesos y la llave, cuando hablemos con el perro el personaje dirá con un diálogo que aún no tiene lo necesario para continuar.



Figura 6.2.8T. Diálogo del personaje frente al perro

Este camino por el que nos guiará el perro consiste en una caída al vacío, donde durante la caída, veremos de fondo algunas obras famosas del Renacimiento.

Para que el personaje pueda apreciar mejor bien las obras, se han implementado un script, donde, la fuerza gravitatoria personaje será 0, y caerá por la inercia que lleve en ese momento.

El script se activará a partir de una colisión del personaje con el dibujo de las obras de época.

```
void OnTriggerEnter2D(Collider2D collider)
{
    if (collider.gameObject == character)
    {
        character.GetComponent<Rigidbody2D>().gravityScale = new_gravity;
        character.GetComponent<Rigidbody2D>().velocity = new
Vector2(character.GetComponent<Rigidbody2D>().velocity.x, 0);

        if (!can_doubleJump)
        {
            character.GetComponent<MainCharacterController>().doubleJump = true;
            //The character can't double jump now
        }
        else
        {
            character.GetComponent<MainCharacterController>().doubleJump = false;
            //The character can double jump now
        }
        if (can_destroy)
            Destroy(gameObject);
    }
}
```

Como se denota en el código, como la fuerza gravitatoria del personaje es 0, no se permitirá por defecto, que el personaje pueda realizar un doble salto en el aire durante la caída, ya que ello provocaría que se empezara a elevar con gravedad 0.

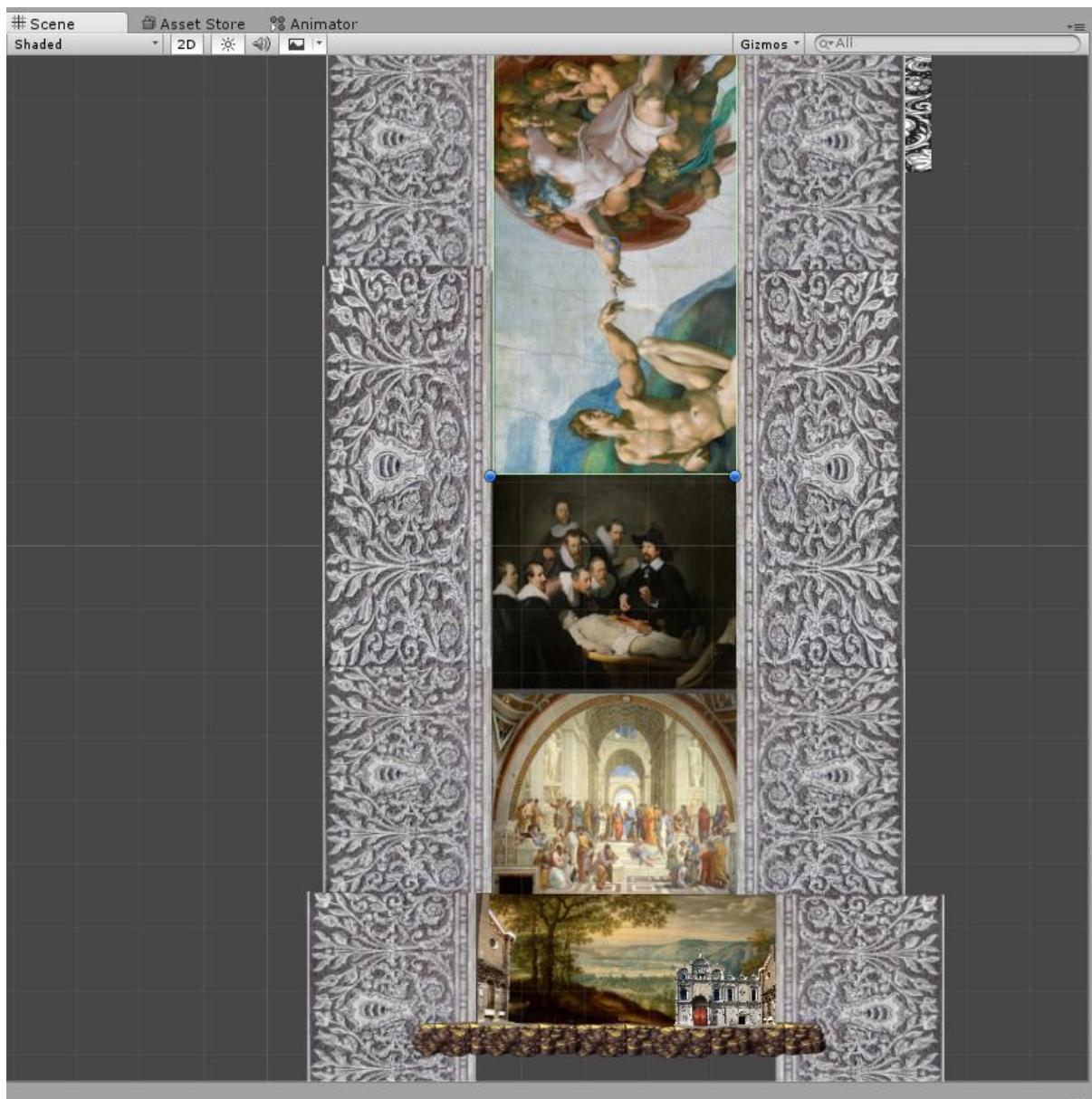


Figura 6.2.8U. Vista lejana de las obras visibles durante la caída del personaje al final de la fase renacentista

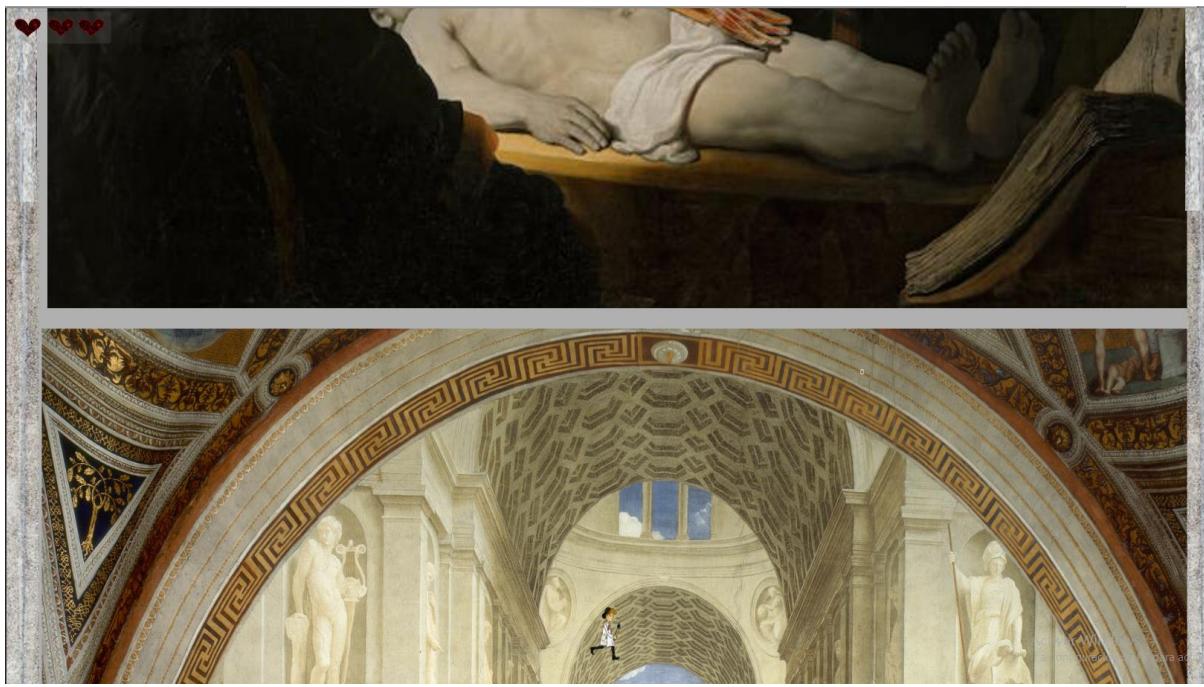


Figura 6.2.8V. Fotograma de caída del protagonista en gravedad 0.

Antes de llegar a la superficie del fondo, el personaje recobrará su gravedad habitual.

Al final veremos de nuevo al perro y finalmente nos guiará hacia el portal al siguiente mundo:



Figura 6.2.8W. Final de la época renacentista. Portal al siguiente mundo

6.2.8.6 Época vanguardista: Abstracción

En esta fase nos encontramos en un mundo dominado por la abstracción y de ambientación tranquila, ya que no hay enemigos alrededor y la música es más chillout.

En esta fase prima el plataformeo y el uso de nuestras habilidades para superar los obstáculos. El personaje dispone de tres transformaciones, entre las cuales se puede cambiar pulsando dos teclas distintas: una para el cambio de forma abstracta (cubo y esfera) y otra para el cambio de plano material (2D y 3D).

Las formas que Dorian puede adoptar en esta fase son, pues:

- **Forma base (con toque abstracto):** Esta forma es como la forma base normal pero con apariencia adaptada a la fase, concretamente algo cubista. Las habilidades de las que dispone el personaje estando en esta forma son solamente saltar, correr y atacar (aunque esta última no resulta necesaria).

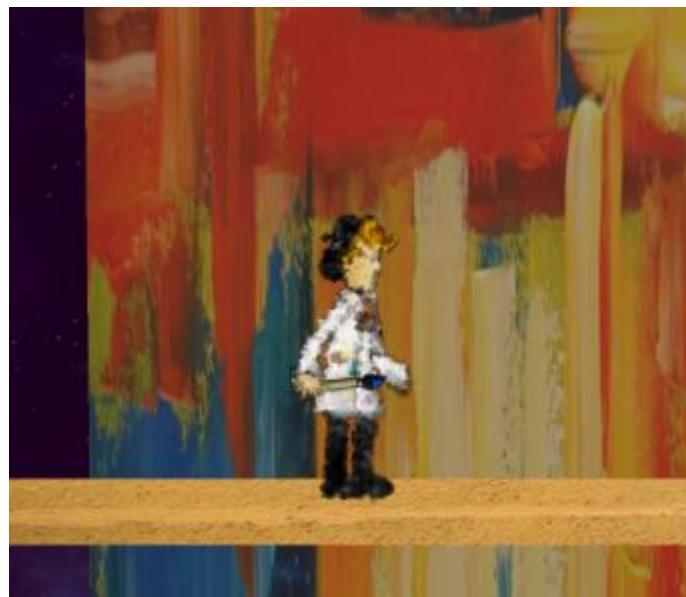


Figura 6.2.8.6A. Forma base de Dorian

- **Forma cúbica:** estando en esta forma Dorian es capaz de realizar saltos de pared a pared, siendo así capaz de llegar a zonas con las que las demás formas no podrían.

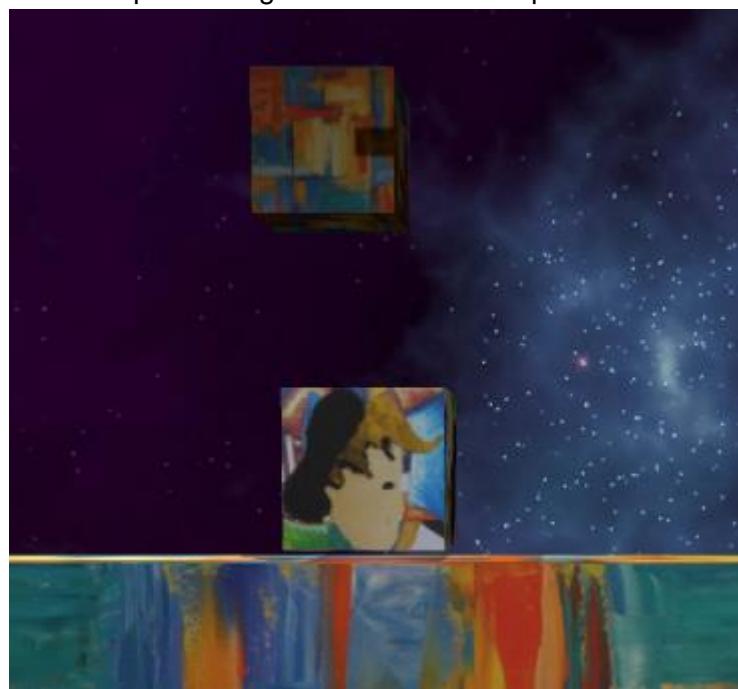


Figura 6.2.8.6B. Forma cúbica de Dorian

- **Forma esférica:** la peculiaridad de esta bola rebotante es que es capaz de realizar saltos más altos, y si pulsamos el botón de salto justo al tocar el suelo...el siguiente salto será más alto aún. Esto nos ayudará a saltar a plataformas que de otra forma no llegaríamos.



Figura 6.2.8.6C. Forma esférica de Dorian.

Entre las mecánicas disponibles en esta fase, una de ellas es la de saltar de pared en pared con nuestra forma cúbica, para lo cual tenemos que pulsar el botón de salto justo en el momento en el que estemos tocando una pared con uno de los lados del cubo. En la siguiente figura se ilustra dicha mecánica:



Figura 6.2.8.6D. Mecánica de saltar entre paredes

Otra mecánica, la más atractiva de esta fase, es la alternar entre 2D y 3D, alternando también el plano visual donde nos encontramos, situándonos al fondo o al frente. Pulsando una tecla podemos alternar entre estos dos planos para pasar por sitios donde solo está disponible uno de ellos, y volver al otro plano cuando el actual no se pueda seguir, y viceversa.

Tras cambiar del plano 2D al 3D se generará una estela en forma de generador de partículas brillantes indicando donde se ha producido el último cambio, y tal y como se puede observar en esta figura.



Figura 6.2.8.6E. Estela dejada tras cambiar de plano.

Al cambiar de plano habrá situaciones en las que no haya suelo donde pisar al estar en modo 2D, es aquí cuando tendremos que hacer uso de nuestra habilidad del cambio de plano pero esta vez de forma que nos quedemos adheridos a las paredes, como si de un dibujo se tratase. Los movimientos son los mismos solo que estaremos limitados por la superficie en la que nos encontramos “dibujados”.

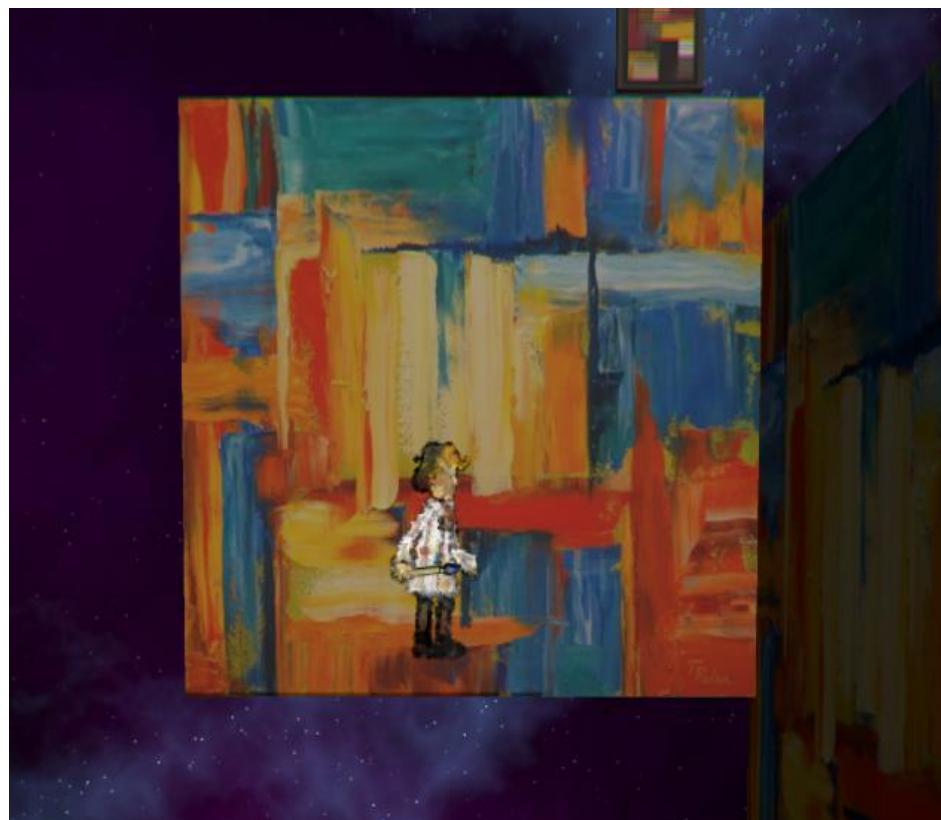


Figura 6.2.8.6F. Adhesión a paredes

6.2.8.7 Época técnica: Grabado xilográfico

En esta fase nos encontramos inmersos en un mundo donde desde el primer momento veremos unos escenarios con estilo xilográfico.



Figura 6.2.8.7A. Apariencia del protagonista en fase xilográfica.

Empezaremos con nuestro protagonista, con un aspecto cambiado, adecuado a la fase en al que se encuentra.

El protagonista tendrá exactamente el mismo modelo que en la fase anterior, pero con los sprites de cada parte de su cuerpo modificados.

Comenzaremos el nivel en un pasillo, donde por el camino nos encontraremos una prensa de encuadernación y arte gráfico. Cuando nuestro personaje pase a través de la prensa, adquirirá la habilidad de convertirse en la criatura xilográfica que se mostró anteriormente.



Figura 6.2.8.7B. Prensa para adquirir transformación xilográfica

Ésta criatura tendrá unas mecánicas de salto en las que es capaz de incrementar su altura si pulsamos el botón justo al tocar el suelo, siendo el siguiente salto mucho más alto que el anterior. Esto nos ayudará a saltar a plataformas que de otra forma no llegaríamos.

También, en el aire, podemos realizar un impulso hacia el suelo, lo que hará que rebotemos contra él, (que podemos combinar con el incremento de salto anterior)

En un salto normal, podrá realizar un segundo salto en aire.

Se han añadido plataformas móviles para sortear los elementos, poder coger colecciónables y avanzar por el escenario.

Éstas plataformas, irán cambiando de color entre blanco y negro. Se han elegido éstos colores para situarlos en corcordancia con todos los escenarios de ésta fase.

Para ello, se han configurado un material que contiene dos shaders distintos, para poder pintarlo de blanco y negro.

El código implementado irá cambiando dicho material cada X segundos, siendo X un rango entre 1 y 5 segundos.

```

// Update is called once per frame
void Update () {
    if (Time.time > nextFlick)
    {
        if (is_black)
        {
            change_to_white();
            is_black = false;
        } else
        {
            change_to_black();
            is_black = true;
        }
        nextFlick = Time.time + Mathf.Floor(Random.Range(1, 5)); ;
    }
}

void change_to_white()
{
    Shader myShader;

    myShader = Shader.Find("Standard"); //Shader configured for white color

    mySpriteRenderer.material.shader = myShader;
}

void change_to_black()
{
    Shader myShader;

    myShader = Shader.Find("Sprites/Default"); //Shader configured for black color

    mySpriteRenderer.material.shader = myShader;
}

```

En el siguiente fotograma podemos ver a la criatura xilográfica entre plataformas móviles y estáticas utilizando el script de cambio de color.



Figura 6.2.8.7C. Criatura xilográfica junto con plataformas móviles y estáticas

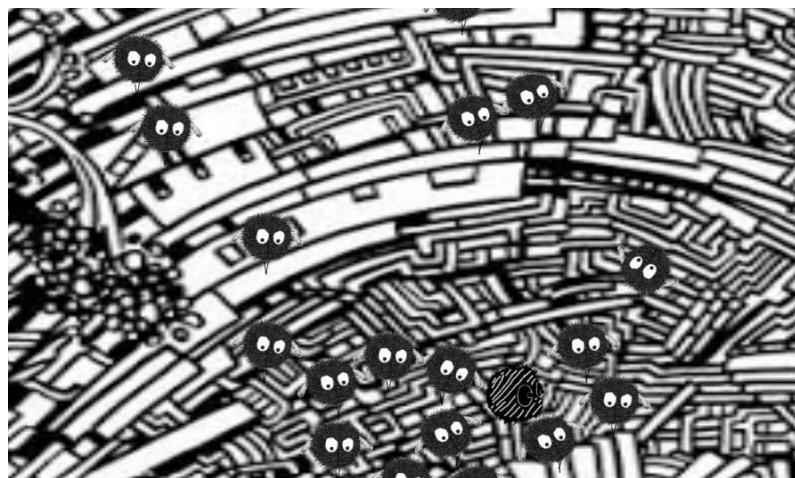


Figura 6.2.8.7D. Criatura sobre insectos voladores

A mitad de nivel se ha hecho uso de los insectos voladores infensivos que vimos en la fase renacentista. Tendremos que cruzar un trecho de vacío, con un puente hecho por éstos insectos, los cuales, al estar el aire, servirán de apoyo para que nuestra criatura pueda correr sobre ellos. Pero deberá hacerlo rápido porque desplazaremos poco a poco a los insectos y podremos caernos al vacío.

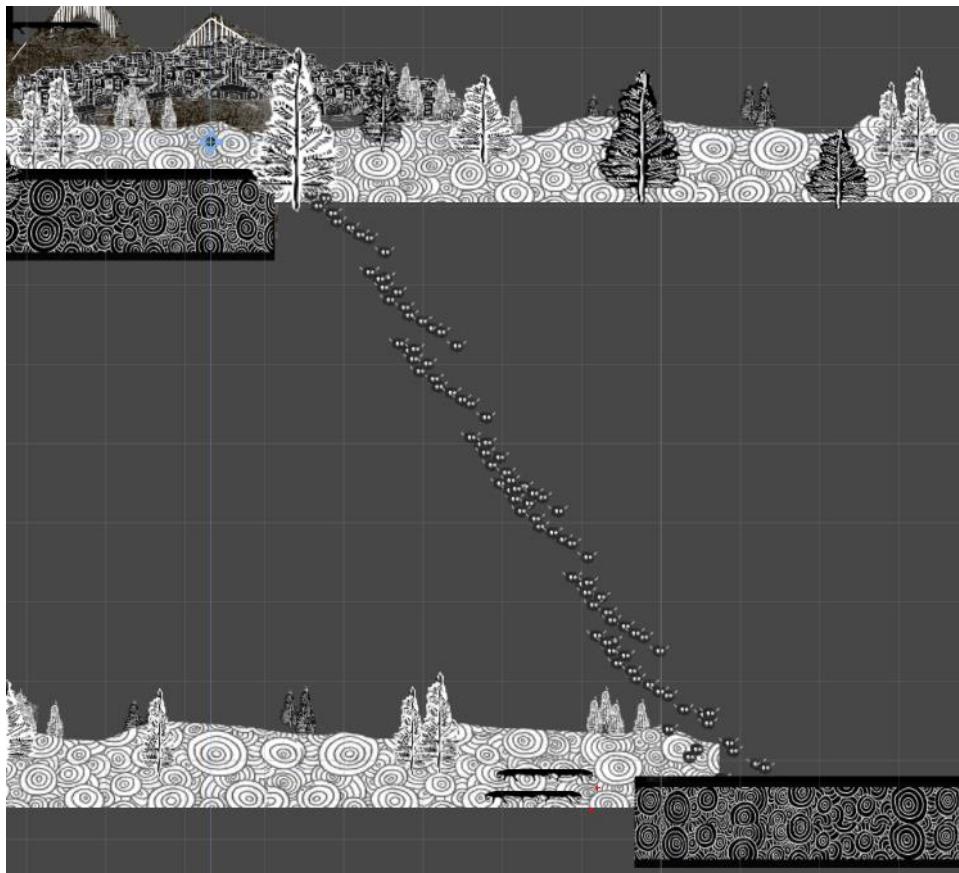


Figura 6.2.8.7E. Zona intermedia de la fase xilográfica en el editor de Unity

La última parte del nivel consiste en un plataformeo con habilidad. Consiste en el uso de las plataformas estáticas y móviles, que cambian de color (blanco y negro) sobre un mosaico de cuadros negros y blancos (como un tablero de ajedrez).

En esta parte se encontrará un personaje perdido del cuadro, otra Menina. Para llegar a ella debemos ir saltando de plataforma en plataforma, fijándonos bien en cada momento en donde se encuentran las plataformas ya que se irán fusionando con el fondo.

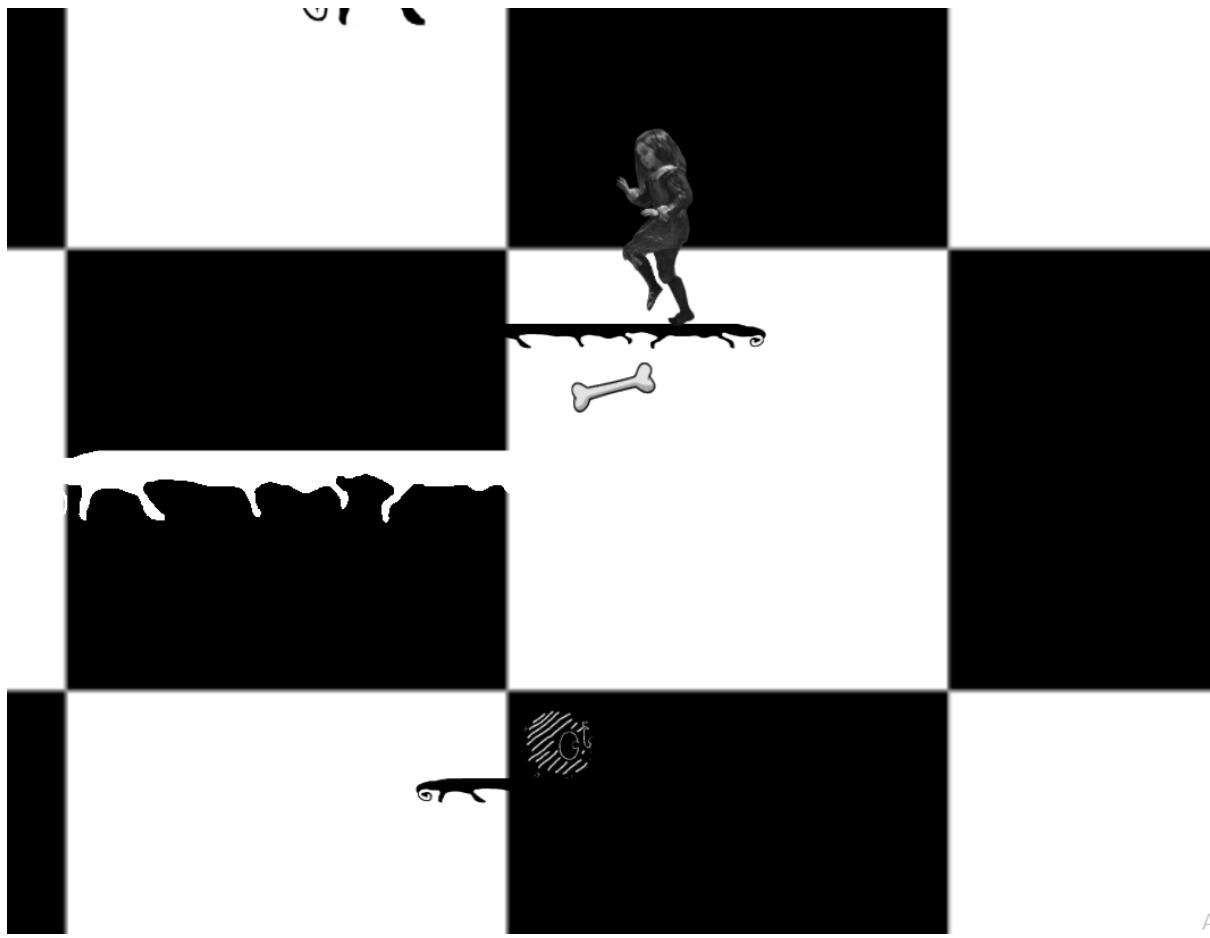


Figura 6.2.8.7F. Plataformeo con mosaico en blanco y negro.

6.2.9 Sonido

6.2.9.1 Música

Hemos seleccionado distintas pistas musicales acordes a la fase correspondiente, todas ellas obtenidas de sitios web de música open-source, es decir, libre de licencias privativas (véase las referencias).

Para la introducción hemos seleccionado un tema suave con aire misterioso y místico, el cual crea cierta ambientación al iniciar el juego.

Para la fase de selección de fase hemos puesto una música calmada, como la que habría en un museo, del estilo de la música clásica.

Para las distintas fases del juego hemos seleccionado diversas pistas musicales que recuerdan a la música chillout, lo que nos transmite una sensación de tranquilidad, ya que este no es un juego de acción puro. El objetivo es jugar tranquilamente y proporcionar una sensación de calma mientras se superan las distintas fases.

Para el final del juego se ha introducido una pista algo más emotiva y lenta, a modo de despedida de nuestra aventura.

6.2.9.2 Narrativa

Se ha usado un procesador TTS (text to speech) con voz de persona mayor, la cual parecía la más apropiada ya que transmitía profundidad y misterio.

Se han procesado los siguientes textos:

- Introducción del juego: se explica cómo empieza la historia de nuestro personaje y qué le sucede.
- Inicio de fase: se explica brevemente el objetivo a seguir y se describe un poco la fase de manera que suene misterioso.
- Fin de fase: breve narrativa indicando que hemos conseguido nuestro objetivo y breve descripción de lo que nos queda por hacer.
- Fin del juego: narrativa a modo de ending del juego, donde se relata cómo acaba la historia, y posible continuación (trabajo futuro).

6.2.9.3 Sonidos del juego

Los sonidos usados para las acciones que ocurren durante el juego, como saltos, explosiones, muertes de enemigos, etc... han sido obtenidas de sitios web de assets gratuitos (véase las referencias), y hemos seleccionado los más convenientes.

7. Código

Todo la estructura que se ha comentado anteriormente está formada por las clases, algunas de las cuales se nombrarán a continuación a modo de ejemplo. Habrán usualmente dos tipos de variable, privadas, las cuales usaremos solo en esa clase, o públicas. Las públicas las declaramos así para poder acceder a ellas desde otras clases, o simplemente para asignar desde la propia interfaz de Unity un objeto a esa variable, arrastrando el objeto a la variable dentro del script añadido como componente a un objeto en concreto.

Primero, hay que dejar claro, que una sola clase no suele definir un objeto de la escena de juego completo. Los objetos suelen tener más de un script asociado, y cada script contendrá una clase a la que dará nombre.

Por ejemplo, el objeto `GameObject` padre del personaje principal, contiene 4 scripts con sus 4 clases. Y algunos de los hijos del padre, también contendrán scripts, como por ejemplo la mano junto con el pincel del protagonista contiene un script con una clase para realizar el ataque y detectar enemigos.

Así pues, en la siguiente captura podemos ver el objeto padre del personaje principal con sus scripts asociados a la derecha.

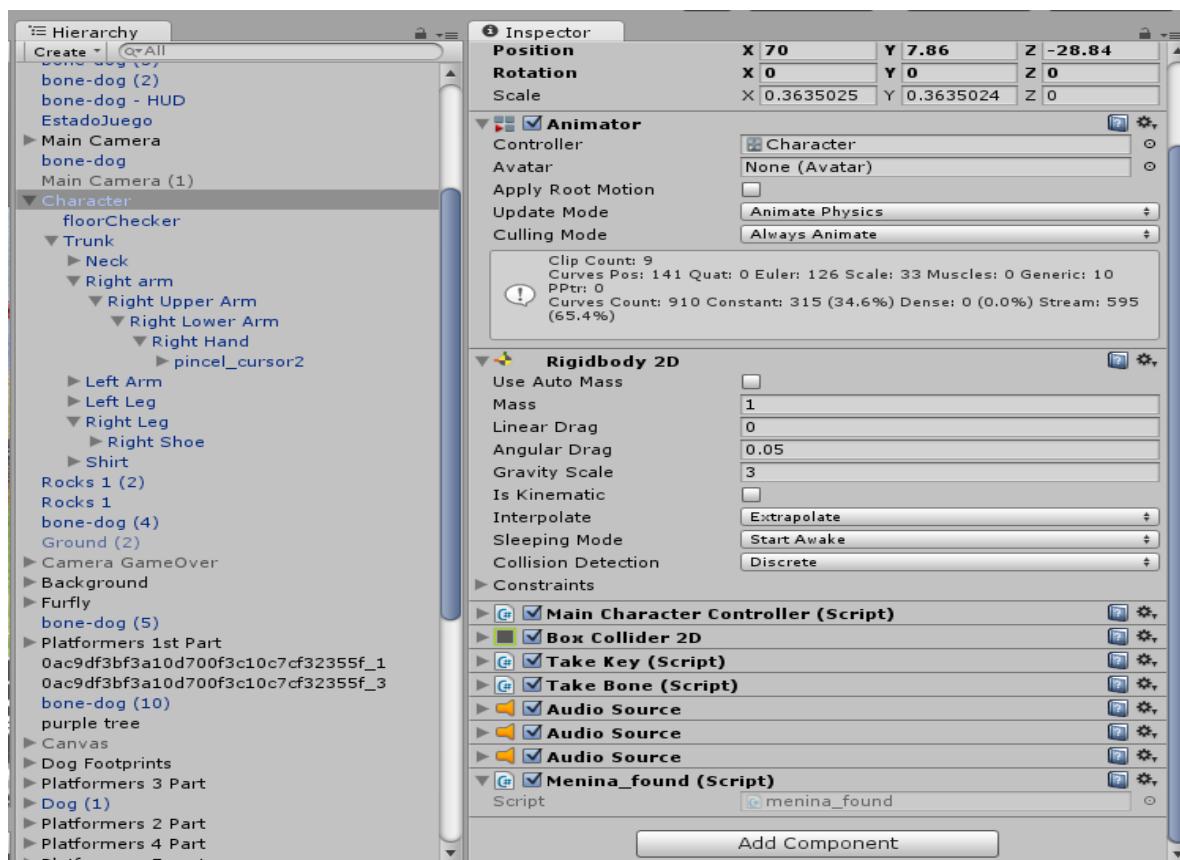


Figura 7A. Scripts asociados al objeto *Character*

Primero nombraremos con detalle algunos componentes genéricos que poseerán casi todos los objetos:

- El componente transform, es proporcionado por Unity y configurado por el maquetador del nivel, este componente indica la posición, orientación y escala del objeto dentro de la escena.
- El componente Animator que aunque no todos los objetos lo posean dado que es un componente de animación del modelo, cualquier objeto de la escena que sea capaz de moverse realizando una animación es probable que lo tenga, este componente está implementado por Unity pero necesita una base en el sistema Mecanim que incorpora la propia interfaz de Unity.
- Otro componente proporcionado por Unity y que debe ser configurado por el maquetador es el componente RigidBody que especifica el cuerpo físico del objeto para aplicarle la física, cualquier objeto dentro de la escena al que se le quieran aplicar las leyes de la física debe contener este componente.
- Un componente necesario si deseamos procesar colisiones es el colisionador, normalmente será un Box Collider 2D, aunque también usaremos algún Polygon Collider 2D o Circle Collider 2D.
- Finalmente un componente opcional es AudioSource, el cual simula los sonidos que puede emitir el objeto.

7.1 Ejemplos de clases - MainCharacterController

Detallaremos como primer ejemplo una clase bastante importante en el juego, la clase encargada de controlar casi todos los parámetros asociados al movimiento del protagonista principal, la clase MainCharacterController.

7.1.1 Atributos:

- myRigidbody, es una variable privada de tipo Rigidbody (Clase de Unity) que almacena el componente Rigidbody (Componente de física) del personaje cuando éste es cargado en la escena.
- animator, es una variable privada de tipo Animator (Clase de Unity) que almacena el componente Animator (componente Mecanim) del personaje cuando este es cargado en la escena.
- jumpStrength, es una variable pública que almacena el valor que le queremos asignar a la altura del salto del personaje. Cuanto mayor sea su valor, mayor será la altura máxima del salto.
- onFloor, es una variable privada de tipo booleana, previamente declarada en el sistema de animaciones Mecanim. Ésta variable la usa dicho sistema para realizar las transiciones de una animación a otra. Desde nuestro script modificaremos esta booleana según esté tocando (colisionando con los pies) o no una superficie.
- floorChecker, es una variable pública de tipo Transform (Clase de Unity). Ésta variable se coloca en la posición de los pies del personaje y se utiliza para comprobar si existe una colisión, teniendo en cuenta el valor radio de la siguiente variable, con un objeto que tenga aplicada una capa “suelo”.
- radiusChecker, es una variable privada de tipo Float, donde se asignará un valor de radio para su implementación junto a la variable anterior y su utilidad.
- floorMask, es una variable de tipo LayerMask (Clase de Unity), donde asignaremos a través de la interfaz de Unity la capa, la cual asignaremos a los objetos que queremos que entren en juego con las dos anteriores variables y su utilidad con la variable onFloor.

- doubleJump, es una variable privada de tipo booleana, usada para realizar un segundo salto en el aire.
- m_FacingRight, es una variable privada de tipo booleana para saber si el personaje está mirando hacia la derecha o hacia la izquierda, para manejar su movimiento correctamente.
- running, es una variable privada de tipo booleana para conocer si se está presionando el botón para que el personaje corra a más velocidad.
- crouch, es una variable privada de tipo booleana para conocer si se está presionando el botón asignado para que el personaje se agache.
- can_uncrouch, variable privada de tipo booleana que nos permite saber, en caso de que el personaje esté agachado, si puede levantarse o no. (en el caso de que tenga un objeto por encima y no tenga espacio para levantarse, no podrá hacerlo)
- speed, es una variable pública de tipo Float para indicar la velocidad de movimiento del personaje.
- numer_bones, es una variable pública de tipo Int, para indicar cuantos huesos de perro ha conseguido el personaje en el nivel en el que se encuentra.
- max_bones, es una variable pública de tipo Int, para indicar cuantos huesos de perro en total puede el personaje conseguir.
- is_attacking, es una variable de tipo booleana que nos indica si en ese momento el personaje está atacando.
- ink_attack_sprite, variable pública de tipo gameobject (clase de Unity), al que nos permite desde la interfaz de Unity asignarle un sprite que se mostrará en el ataque de nuestro personaje con su pincel.
- position_ink_attack, variable pública de tipo Transform (clase de Unity), que nos permite desde la interfaz de Unity asignarle una posición donde colocar el sprite de ataque previamente comentado.
- PlayerStats, clase creada dentro de la clase MainCharacterController, contiene el número de salud máxima, y la actual que tiene el personaje.

- stats, instancia de PlayerStats.
- audios, variable privada que contiene un array de AudioSource, que serán audios que el personaje podrá reproducir.
- fallBoundary, variable pública de tipo entero, que determinará la altura negativa a la que el personaje morirá. (en caso de caída)
- deathSoundName, variable pública de tipo string, que contiene el nombre del audio reproducido cuando muere el personaje.
- damageSoundName, variable pública de tipo string, que contiene el nombre del audio cuando el personaje recibe daño.
- lifeIndicator, variable privada de tipo LifeIndicator, que será el indicador de corazones de salud en pantalla, representando la salud del personaje.
- spawnPosition, variable privada de tipo Transform, contendrá la posición inicial para el respawn del personaje en caso de muerte.
- stats, variable pública de tipo PlayerStats, clase que contiene los valores de salud del personaje y funciones para su manejo.
- able_to_run, variable pública de tipo booleana, se utiliza para activar o no la capacidad de correr (con el botón de correr a más velocidad) del protagonista.

7.1.2 Métodos:

- Start(), método con visibilidad de paquete, retorna void, este método se encarga de inicializar todos los atributos de la clase o de realizar alguna operación previa como obtener los componentes necesarios es el constructor que proporciona Unity. (Llamado por Unity automáticamente)
- Update(), método con visibilidad de paquete, retorna void, se encarga de realizar los análisis o de actualizar las variables del personaje en cada frame renderizada. (Llamado por Unity automáticamente)
- FixedUpdate(), método con visibilidad de paquete, retorna void, se encarga de realizar los análisis o de actualizar las variables del personaje cada cierto tiempo, es llamada por Unity en cada frame. (Llamado por Unity automáticamente)
- OnTriggerEnter2D(Collider collider), método con visibilidad de paquete, retorna void, este método se encarga de procesar las posibles entradas de objetos dentro del trigger (Colisionador sin física) del objeto y realizar acciones según el tipo de objeto que ha entrado en su trigger, el parámetro collider contiene todos los

parámetros relacionados con el colisionador que ha entrado en el trigger. (Llamado por Unity automáticamente)

- OnTriggerExit2D(Collider collider), método con visibilidad de paquete, retorna void, este método se encarga de procesar las salidas de objetos procedentes de un trigger (Colisionador sin física) previo del objeto y realizar acciones según el tipo de objeto que ha salido de su trigger, el parámetro collider contiene todos los parámetros relacionados con el colisionador del que se ha salido en su colisión. (Llamado por Unity automáticamente).
- attack(), método con visibilidad privada, retorna void, este método se utiliza para realizar un ataque cuerpo a cuerpo, y se encarga de llamar a la animación correspondiente.
- Flip(), método con visibilidad privada, retorna void, este método se utiliza para cambiar la dirección en la que se apunta el personaje, izquierda o derecha, para así acometer las acciones con la orientación correspondiente.
- HandleMovement(float horizontal, bool crouch, bool running), método con visibilidad privada, retorna void, este método se utiliza para mover al personaje. La variable horizontal se utiliza para calcular su velocidad en el eje x, crouch se utiliza para saber si se está en un estado agachado o no (lo que haría reducir su velocidad), y running se utiliza para comprobar si se está corriendo para así aumentar la velocidad de movimiento.
- takeBone(), método con visibilidad pública, retorna void, se utiliza para coger un hueso, y aumentar el número de ellos que tiene el personaje en su poder.
- DamagePlayer(int damage), método con visibilidad pública, retorna void, se utilizar para restar al total de vida del personaje el valor de damage, el daño recibido.
- setToSpawn(), método con visibilidad pública, retorna void, se utiliza para colocar al personaje en la zona de respawn, a causa de una muerte.

Colocaremos en esta parte sólamente el código de un par de funciones, para mostrar la tónica del refresco de frames de Unity:

- **FixedUpdate() - Código :**

```
void FixedUpdate()
{
    if (Input.GetKeyDown(KeyCode.LeftControl))
    {
        crouch = true;
    }
    else
    {
        if (crouch)
```

```

        {
            if (can_uncrouch)
            {
                crouch = false;
            }
        }
    animator.SetBool("Crouch", crouch);
    if (running)
    {
        GetComponent<Rigidbody2D>().velocity = new
Vector2(GetComponent<Rigidbody2D>().velocity.x, GetComponent<Rigidbody2D>().velocity.y);
    }
    animator.SetFloat("VelX", GetComponent<Rigidbody2D>().velocity.x);
    onFloor = Physics2D.OverlapCircle(floorChecker.position, radiusChecker, floorMask);
    animator.SetBool("isGrounded", onFloor);

    if (onFloor)
    {
        doubleJump = false;
    }
}

```

- **Update() - Código :**

```

// Update is called once per frame
void Update()
{
    if (transform.position.y <= fallBoundary)
        DamagePlayer(9999999);

    float horizontal = Input.GetAxis("Horizontal");
    if (horizontal > 0 || horizontal < 0)
    {
        if ((horizontal < 0 && m_FacingRight) || (horizontal > 0 && !m_FacingRight))
        {
            Flip();
        }

        if (!running)
        {
            //NotificationCenter.DefaultCenter().PostNotification(this,
"PersonajeEmpiezaACorrer");
        }
        running = true;
    }
    else {
        running = false;
        //NotificationCenter.DefaultCenter().PostNotification(this,
"PersonajeHaParado");
    }

    bool running_fast = Input.GetKey(KeyCode.LeftShift);

```

```

if (able_to_run)
    animator.SetBool("RunFast", running_fast);
HandleMovement(horizontal, crouch, running_fast);

if (Input.GetKeyDown(KeyCode.Space))
{
    if ((onFloor || !doubleJump) && !crouch)
    {
        if (!doubleJump && !onFloor)
        {

            doubleJump = true;
        }
        int randomSound = (int)Random.Range(0, audios.Length);
        audios[randomSound].Play();
        GetComponent<Rigidbody2D>().velocity = new
Vector2(GetComponent<Rigidbody2D>().velocity.x, jumpStrength);

    }
}

if (Input.GetKeyDown(KeyCode.E) || Input.GetMouseButtonDown(0))

{
    //int randomSound = (int)Random.Range(0, audios.Length);
    //audios[randomSound].Play();
    if (SceneManager.GetActiveScene().name != "EscenaXilografia")
    {
        audioManager.PlaySound("Attack");
        attack();
    }
}
}

```

7.2 Ejemplos de clases - TakeBone

Ésta es una pequeña clase que también contiene el objeto del personaje principal. Se ha detallado aquí para poner también un ejemplo de una clase sencilla.

7.2.1 Atributos

No contiene ningún atributo, más allá del propio objeto en sí mismo. (this)

7.2.2 Métodos

- OnTriggerEnter2D(Collider2D collider), método con visibilidad de paquete, retorna void, este método se encarga de procesar las posibles entradas de objetos dentro del trigger (Colisionador sin física) del objeto y realizar acciones según el tipo de objeto que ha entrado en su trigger, el parámetro collider contiene todos los parámetros relacionados con el colisionador que ha entrado en el trigger. (Llamado por Unity automáticamente)

Al ser un código de pocas líneas, mostraremos su función en ésta sección:

```
void OnTriggerEnter2D(Collider2D collider)
{
    if (collider.tag == "bone")
    {
        Destroy(collider.gameObject);

        this.GetComponent<MainCharacterController>().takeBone();
    }
}
```

7.3 Ejemplos de clases - DumbEnemy

Es la clase asociada al enemigo azul.

7.3.1 Atributos

- myRigidbody, es una variable privada de tipo Rigidbody (Clase de Unity) que almacena el componente Rigidbody (Componente de física) de éste enemigo cuando éste es cargado en la escena.
- speed, es una variable de tipo float, que se utiliza para la velocidad de movimiento del personaje
- direction, es una variable privada tipo float, que se utiliza para indicar la orientación en la que mira y se mueve el enemigo
- MaxDist, es una variable privada de tipo float, que se utiliza para calcular si el personaje principal está en un radio menor al valor asociado, con respecto a este objeto enemigo en cuestión. De estar en el radio, el enemigo se moverá y atacará.
- wait_to_shoot, es una variable privada de tipo float que se utiliza para que el enemigo espere después de realizar un tiro de pintura, para realizar el siguiente tiro.
- gunTip, es una variable pública de tipo Transform, utilizada para indicar en qué posición inicial estará la bola de pintura que se dispara.
- bullet, es una variable pública de tipo GameObject, utilizada para referenciar al objeto de pintura (con su sprite correspondiente) que se disparará.
- nextFire, es una variable privada de tipo Time, que se utiliza para configurar el momento siguiente en el que se puede realizar un tiro de pintura.

7.3.2 Métodos

- Start(), método con visibilidad de paquete, retorna void, este método se encarga de inicializar todos los atributos de la clase o de realizar alguna operación previa como obtener los componentes necesarios es el constructor que proporciona Unity. (Llamado por Unity automáticamente)
- Update(), método con visibilidad de paquete, retorna void, se encarga de realizar los análisis o de actualizar las variables del personaje en cada frame renderizada. (Llamado por Unity automáticamente)
- OnTriggerEnter2D(Collider collider), método con visibilidad de paquete, retorna void, este método se encarga de procesar las posibles entradas de objetos dentro del trigger (Colisionador sin física) del objeto y realizar acciones según el tipo de

objeto que ha entrado en su trigger, el parámetro collider contiene todos los parámetros relacionados con el colisionador que ha entrado en el trigger. (Llamado por Unity automáticamente)

- FireInkBullet(), método con visibilidad privada, retorna void, este método se utiliza para realizar un disparo de pintura, y se encarga de instanciar el sprite de pintura correspondiente.
- Flip(), método con visibilidad privada, retorna void, este método se utiliza para cambiar la dirección en la que se apunta el sprite del personaje.
- Flip_and_change_direction(), método con visibilidad privada, retorna void, este método se utiliza para cambiar la dirección del objeto y del sprite, izquierda o derecha, para así acometer las acciones con la orientación correspondiente.

Como ejemplo, mostraremos el código de dos funciones.

- **Función update() - código:**

```
void Update()
{
    GameObject thePlayer = GameObject.Find("Character");
    if (thePlayer == null)
        return;
    float x_player = thePlayer.transform.position.x;
    float y_player = thePlayer.transform.position.y;
    float x_enemy = transform.position.x;
    float y_enemy = transform.position.y;
    float x_difference = x_player - x_enemy;

    if ((x_difference > 1 || x_difference < -1) && (x_difference < MaxDist) &&
(x_difference > -MaxDist))
    {
        myRigidbody.velocity = new Vector2(direction * speed, myRigidbody.velocity.y);
        //enemy shooting
        FireInkBullet();
    }
    else
    {
        myRigidbody.velocity = new Vector2(0, myRigidbody.velocity.y);
    }
}
```

- **Función FireInkBullet() - código:**

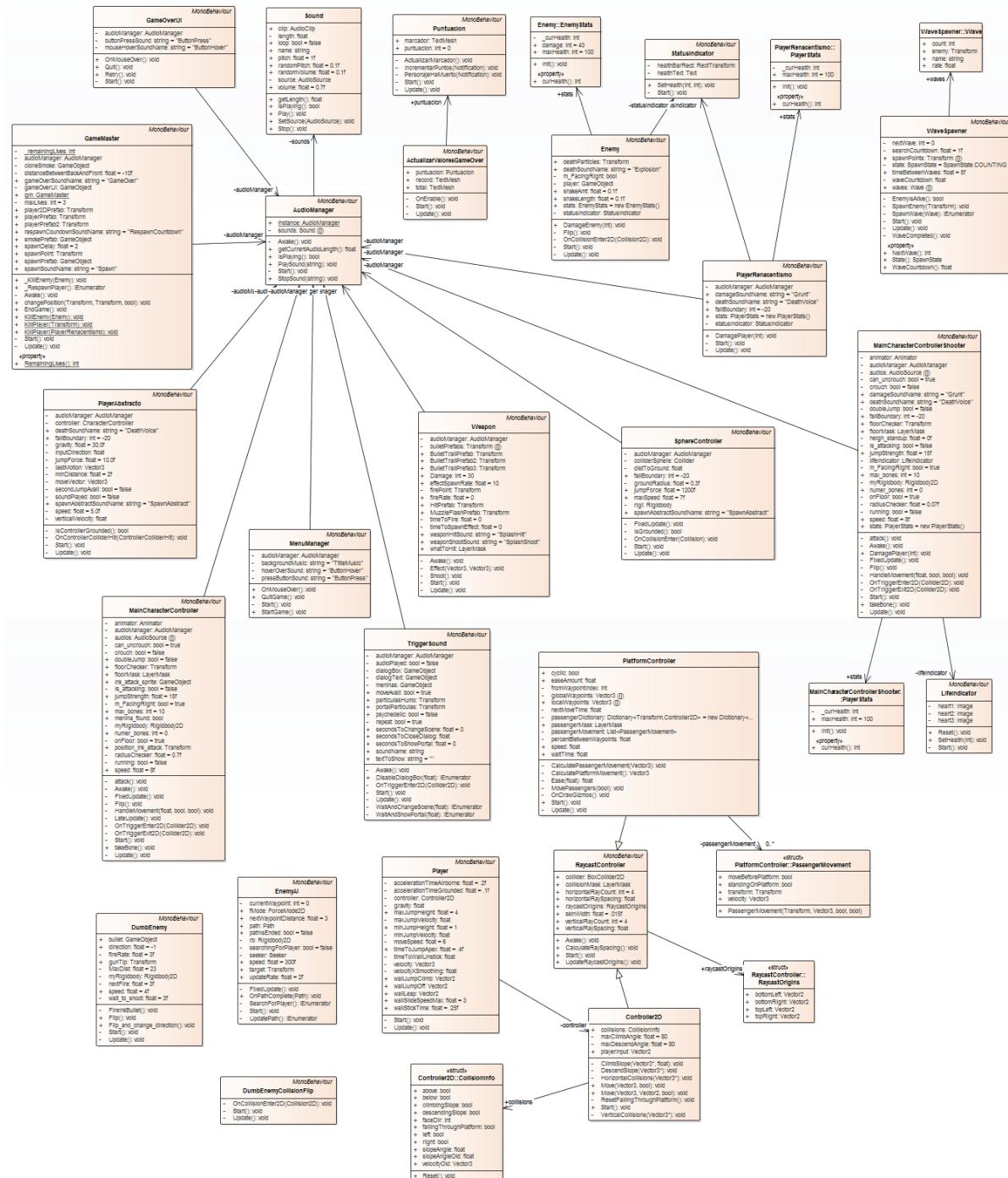
```
void FireInkBullet()
{
    if (Time.time > nextFire+ wait_to_shoot)
    {
        nextFire = Time.time + fireRate;
```

```
        if (direction > 0)
            Instantiate(bullet, gunTip.position, Quaternion.Euler (new Vector3(0, 0,
180f)));
        else Instantiate(bullet, gunTip.position, Quaternion.Euler(new Vector3(0, 0,
0)));
    }

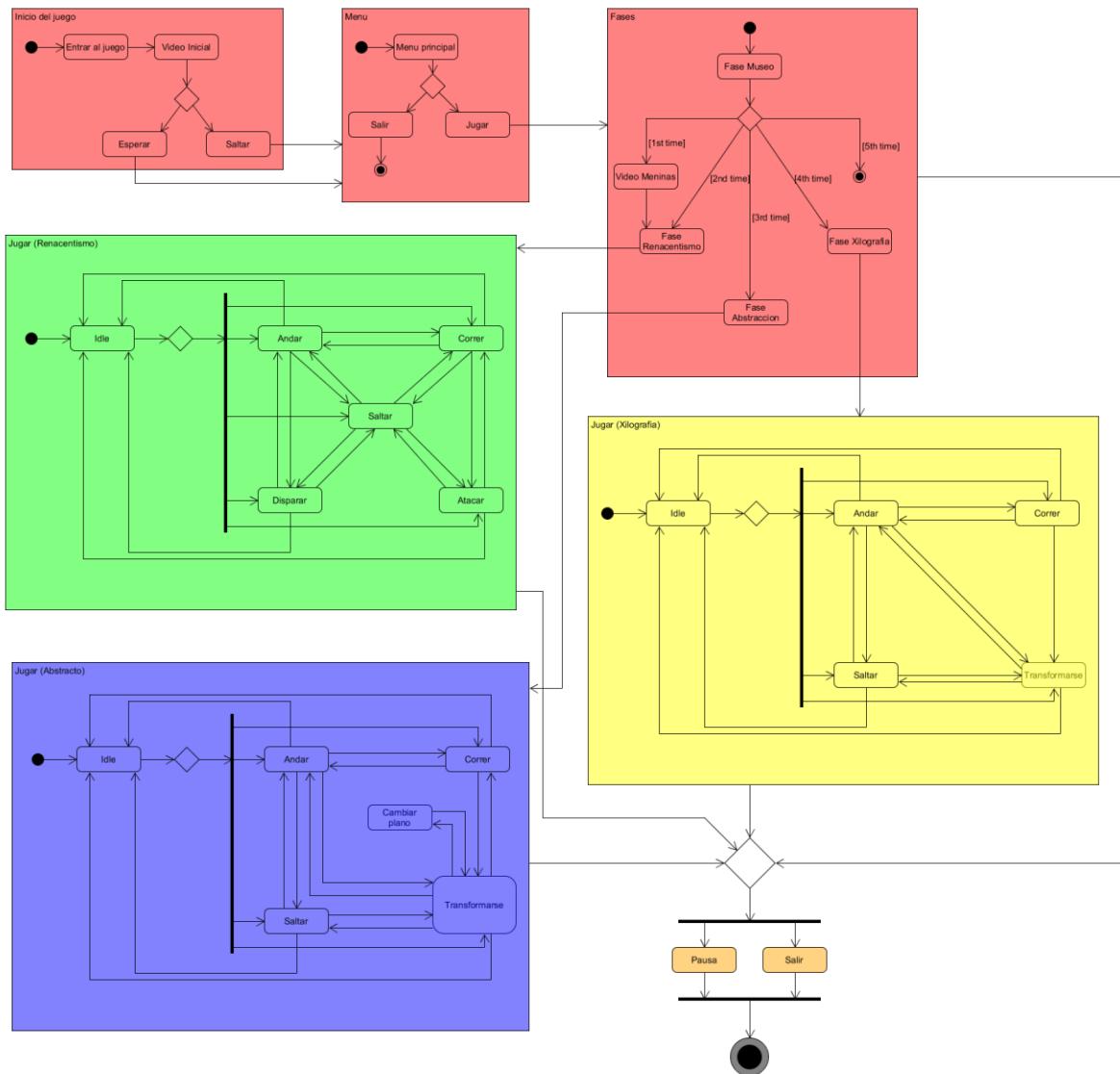
}
```

8. Diagramas

8.1 Diagrama de clases

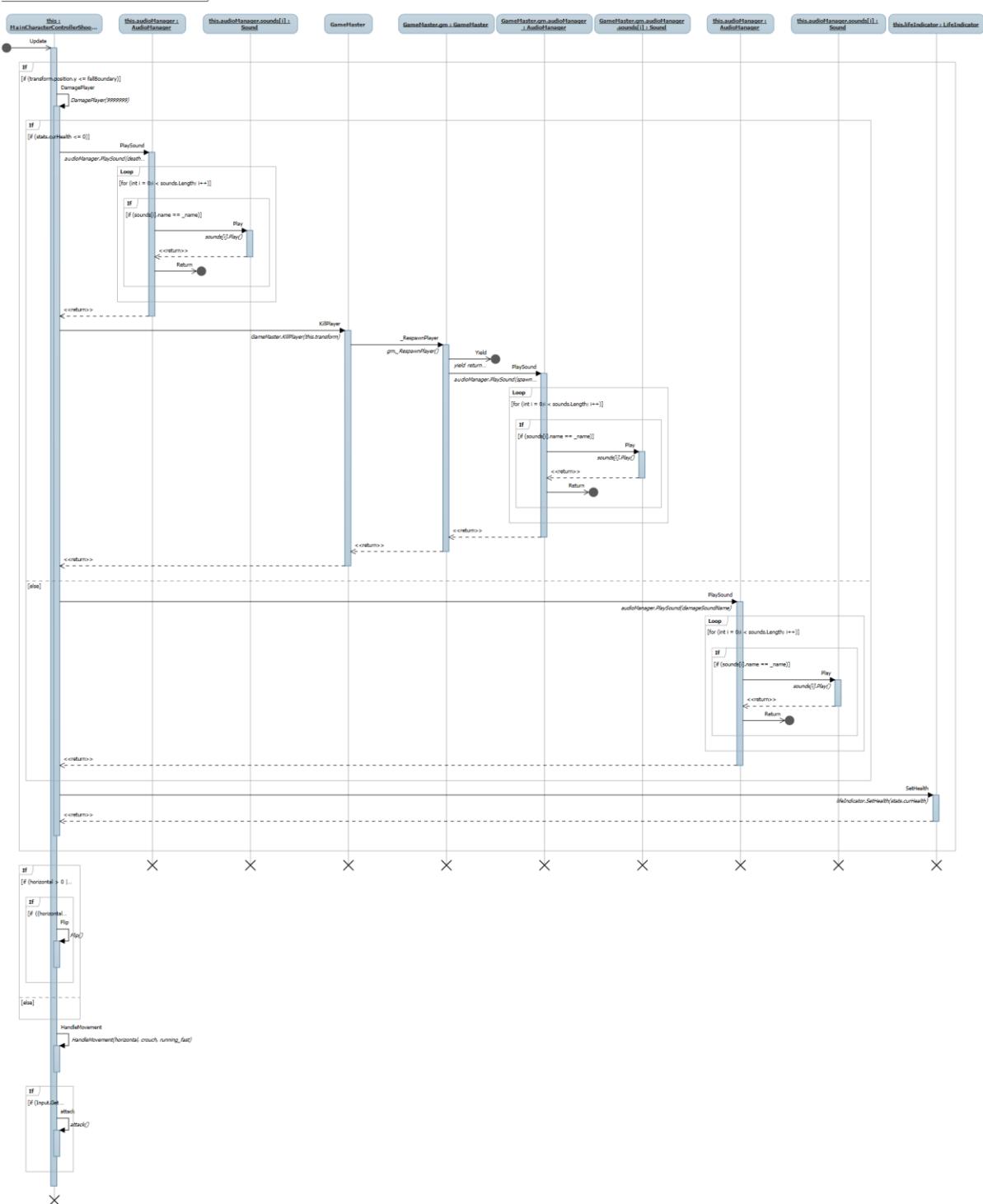


8.2 Diagrama de flujo

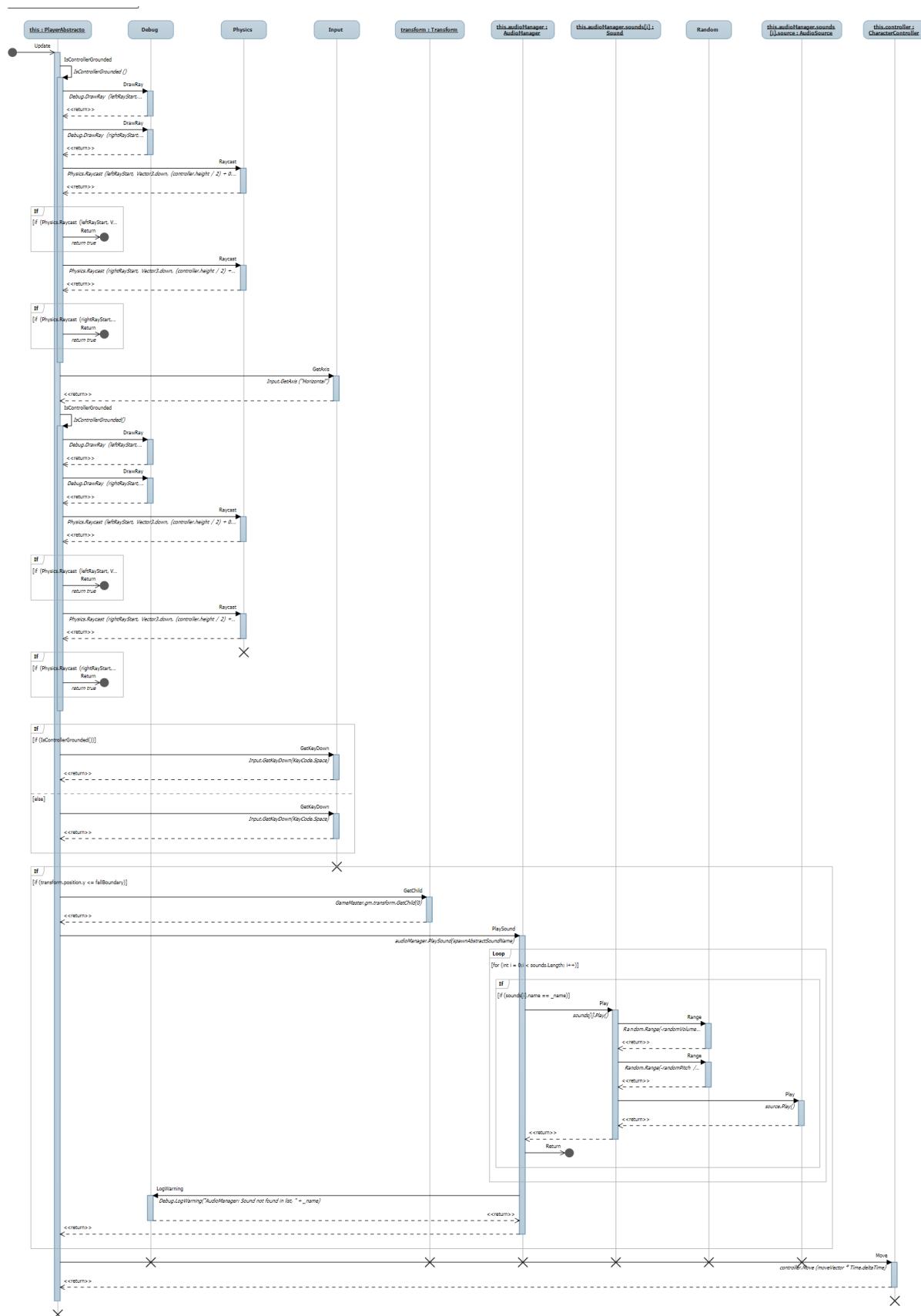


8.3 Ejemplos de diagramas de secuencia

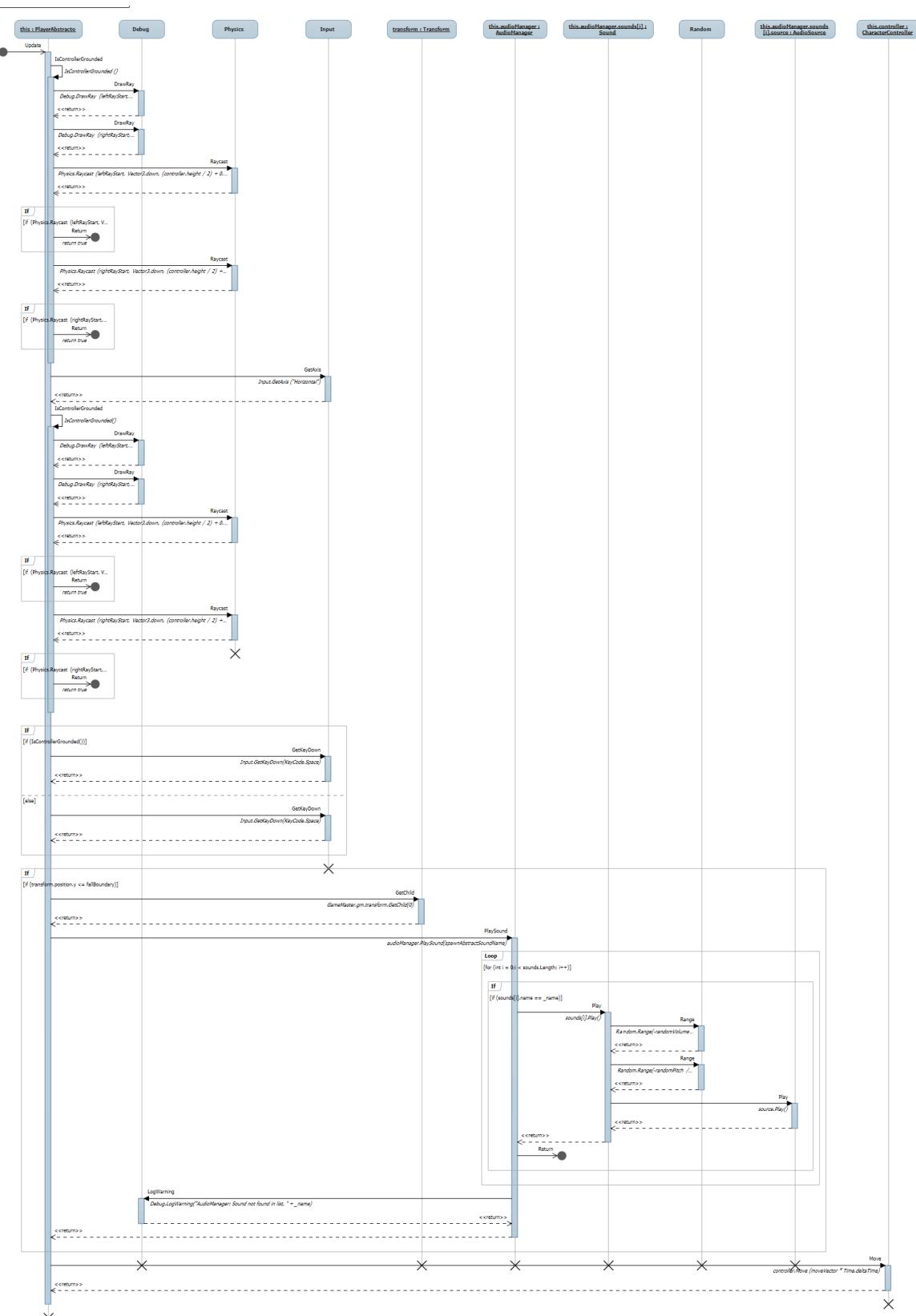
8.3.1 Diagrama de secuencia del seguimiento de la cámara



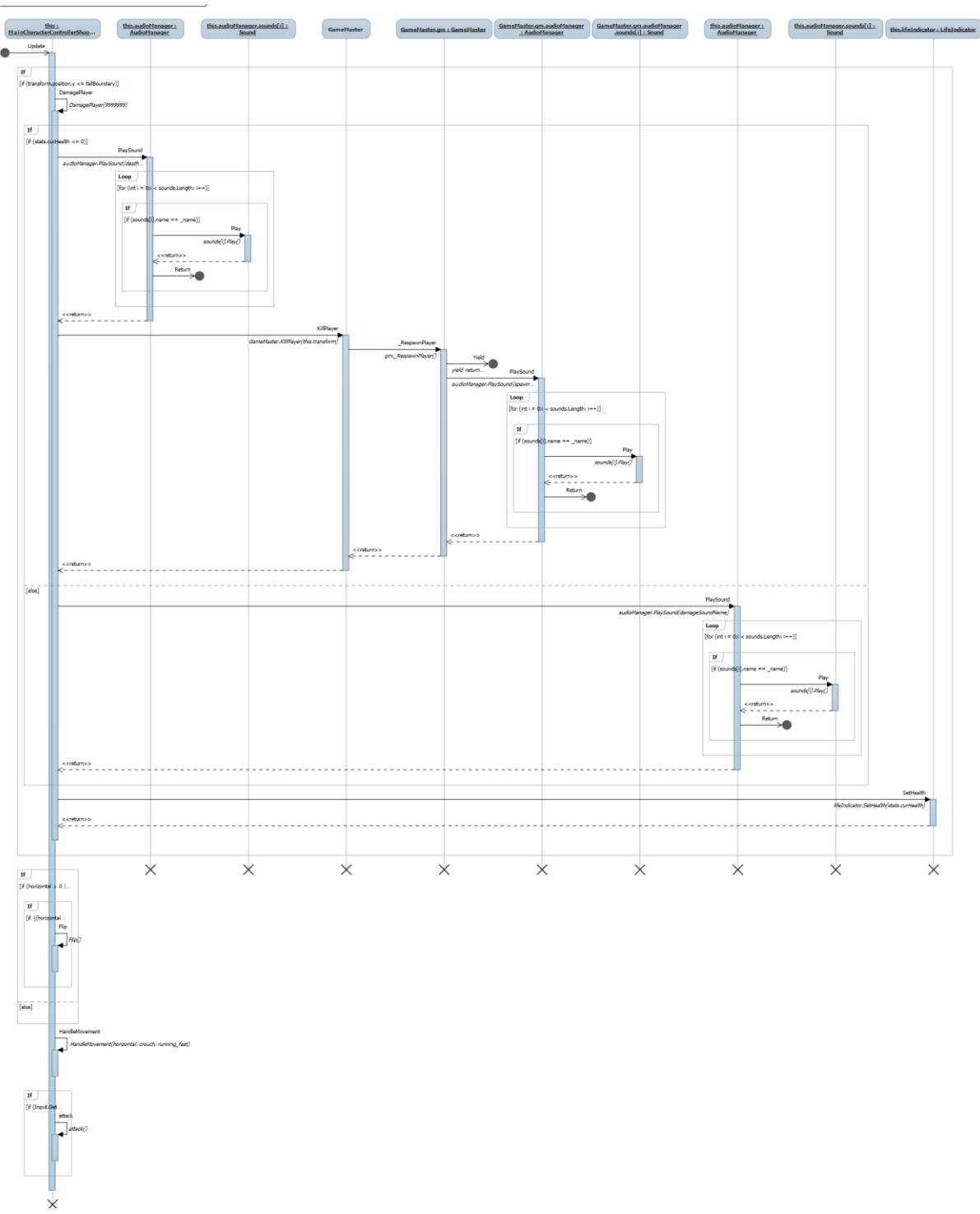
8.3.2 Diagrama de secuencia de cómo se asigna la salud al personaje



8.3.3 Diagrama de secuencia de la función Update (refresca cada frame) del personaje de la fase de abstracción



8.3.4 Diagrama de secuencia de la función Update del personaje de la fase del Renacimiento



9. Conclusiones y trabajo futuro

9.1 Conclusiones

La realización de este proyecto nos ha servido sobretodo para aprender desde cero a desarrollar un videojuego en 2D, además de aprender el uso de Unity, el cual posee una gran potencia y ofrece multitud de posibilidades.

Hemos aprendido también a colaborar con alguien externo a nosotros, en este caso la persona que ha diseñado todo el arte incluido en el juego, Ximena, la cual nos ha proporcionado los recursos necesarios, los cuales ha habido que ajustar acorde a las necesidades.

En general estamos satisfechos con el resultado, aunque sea a modo de demostración, y con las posibilidades que se pueden implementar en un supuesto futuro gracias al potencial explotable.

9.2 Trabajo futuro

- Mejora de las animaciones, usando sprites específicos de cada animación.
- Adicción / ampliación de las fases históricas.
- Implementación de puzzles más complejos.
- Integración de jefes finales.
- Multijugador
- Adición de música original para el juego.

10. Aspectos legales

La actual legislación sobre videojuegos establece que debe de clasificarse el título según su contenido, al ser un videojuego de plataformas y no violento, que contiene combates simples contra algunos enemigos, se ha decidido una clasificación PEGI 7/ESRB T que establece que solo deberían jugar a estos títulos adolescentes con 7 años o más, teniendo en cuenta el trasfondo del juego.

En cuanto a las reglas de copyright, algunos de los modelos y diseños son realizados por nosotros y por nuestra diseñadora, Ximena.

Por otra parte, otros modelos y sonidos se han extraído de fuentes públicas en las que los usuarios libremente dejan recursos de desarrollo gratuitos. Los cuales están nombrados en la lista de referencias.

11. Anexos

Manual de Usuario: <https://drive.google.com/open?id=0BxnBz1IWkfGPOFRkb0ZwcThVdjA>

Video de presentación + tráiler : <https://www.youtube.com/watch?v=L10i7XST4gc>

Repositorio GitHub: <https://github.com/iesmorc/TFG-History-of-Art-Videogame>

12. Bibliografía / Referencias

- [1] Facultat d'Informàtica de Barcelona . *Historia de los videojuegos*. <http://www.fib.upc.edu/retro-informatica/historia/videojocs.html>
- [2] Wikipedia. *Historia de los videojuegos*. https://es.wikipedia.org/wiki/Historia_de_los_videojuegos
- [3] Simone Belli y Cristian López Raventós (2008) [versión electrónica]. España: Universitat Autònoma de Barcelona. *Breve historia de los videojuegos*. Recuperado de: <https://dialnet.unirioja.es/descarga/articulo/2736172.pdf>
- [4] El otro lado. *Historia de los videojuegos: Década de los 80*. http://www.elotrolado.net/wiki/Historia_de_los_videojuegos:_Decada_de_los_80
- [5] Los videojuegos a ti. *La revolución de las 3D*. <https://losvideojuegosati.wordpress.com/la-revolucion-de-las-3d/>
- [6] Universidad da Coruña. *La quinta generación (1994-1997)*. <http://sabia.tic.udc.es/gc/Contenidos%20adicionales/trabajos/Juegos/videojuegos-1995-2004/videoconsolas/c1-quinta-generacion.html>
- [7] Steven L. Kent . *La gran historia de los videojuegos*. <https://books.google.es/books?isbn=8490695520>
- [8] El Arte blogspot. (2013) . *7 bellas artes*. <http://elartejmtt.blogspot.com.es/>
- [9] Historia del Arte blogspot (2008). *El Renacimiento*. <http://historiadelarte07.blogspot.com.es/2008/08/el-renacimiento.html>
- [10] Lluviadelv (2010). *Historia del arte*. Monografías. <http://www.monografias.com/trabajos13/histarte/histarte.shtml>
- [11] Marco Sdb. *El humanismo y el renacimiento* Facultad de ciencias Humanas y Sociales. <http://www.monografias.com/trabajos16/metodo-lecto-escritura/metodo-lecto-escritura.shtml>

- [12] El Arte Vanguardista blogspot. (2013). *El arte vanguardista*. <http://elartevanguardistaa.blogspot.com.es/>
- [13] Sara Lasso. About.com . *Arte abstracto: definición, características y origen*. <http://arte.about.com/od/Diccionario-De-Arte/f/Arte-abstracto.htm>
- [14] Wikipedia (2016). *Xilografía*. <https://es.wikipedia.org/wiki/Xilograf%C3%A3DA>
- [15] Ximena Hidalgo Vásquez (2011). *Videojuegos. Un arte para la historia del arte*. [Versión electrónica]. España: Universidad de Granada.
- [16] Disney Pixar (2015). Escena de fases de abstracción. *Inside Out*. <https://www.youtube.com/watch?v=l2eOUpUpjWg>
- [17] A* PathFindingProject: <http://arongranberg.com/astar/features>
- [18] Asbjørn Thirslund (2014). *How to make a 2D platformer*. <http://brackeys.com/preview/2d-platformer-course/#>
- [19] TTS (Text To Speech): <http://www.acapela-group.com/>
- [20] Assets gratuitos: <http://opengameart.org/>
- [21] Música Open-source: <http://incompetech.com/>
- [22] Clonix Impressions (2014), Youtube. *Just a simple scroll parallax example*. <https://www.youtube.com/watch?v=sGApLQTV-sI>
- [23] Fotografías gratis Renacentismo: <http://www.123rf.com/stock-photo/renaissance.html>
- [24] Fotografías gratis Renacentismo: <http://www.canstockphoto.com/images-photos/renaissance.html>
- [25] Fotografías gratis Renacentismo: <https://www.dreamstime.com/free-photos-images/renaissance.html>
- [26] Unity - Game Engine: <https://unity3d.com/es>
- [27] Dani Candil. *Unity, el motor de desarrollo capaz de partir la historia de los videojuegos en dos*. (2014): <http://www.vidaextra.com/industria/unity-el-motor-de-desarrollo-capaz-de-partir-la-historia-de-los-videojuegos-en-dos>
- [28] Wikipedia (2016) - Unity (software): [https://es.wikipedia.org/wiki/Unity_\(software\)](https://es.wikipedia.org/wiki/Unity_(software))
- [29] Andrés García Morro (2013) - Desarrollo de un sistema de información CAD 3D: <https://riunet.upv.es/bitstream/handle/10251/31899/Memoria.pdf?sequence=1>
- [30] Unity - Store, portal de adquisición: <https://store.unity.com/es>

[31] Guillermo Jiménez Díaz - Tutorial Unity: El paseo del astronauta:
<http://gaia.fdi.ucm.es/files/people/quille/tallerUnity2015/material/quion.pdf>