Universitat Oberta de Catalunya

uoc.edu

El Despertar

Robert Crespo Valero
Grado: Multimedia
Temática: Videojuegos

Consultora: Ester Arroyo Garriguez
Profesor: Javier Luis Cánovas Izquierdo

9 de Junio de 2019

FICHA DEL TRABAJO FINAL

Título del trabajo:	*El Despertar*
Nombre del autor:	*Robert Crespo Valero*
Nombre de la consultora:	*Ester Arroyo Garriguez*
Nombre del PRA:	*Javier Luis Cánovas Izquierdo*
Fecha de entrega:	*01/09/19*
Titulación:	*Grado Multimedia*
Área del Trabajo Final:	*Videojuegos*
Idioma del trabajo:	*Castellano*
Palabras clave:	*Aventura gráfica, 2D, Point&Click*

Resumen del Trabajo (castellano)

El Despertar es un videojuego Point&Click de aventura gráfica en el que nuestro protagonista deberá intentar avanzar en la historia para poder despertar de su sueño.

El videojuego se planteaba como un videojuego clásico, con las interfaces clásicas de los años 90 y con las mismas interacciones. El aspecto deseado inicial era el de una aventura de 8 bits en pixeles.

Para la elaboración del trabajo se ha utilizado el motor gráfico Unity, especializado para desarrollos primerizos gracias a sus facilidades y propiedades únicas. Para la realización de lo deseado se dividirá el trabajo en arte, programación, sonido e historia.

A lo largo del desarrollo los límites humanos y la falta de tiempo ha jugado en su contra consiguiendo que el desarrollo se viera obligado a realizar al final de todo lo que hizo no cumplir con el planteamiento inicial ni con el diagrama de Gantt con el desarrollo de tareas por tiempo.

Al finalizar y ver como el día de la entrega llegaba pisando los talones se intentó conseguir deprisa y corriendo una prueba funcional de los aspectos fundamentales que el juego debería ofrecer. Pese a tener que realizarlo en tiempo récord pueden llegar a apreciarse las diferentes interacciones y en que se basa la mecánica del videojuego.

Abstract

El Despertar is a Point & Click videogame adventure game in which our protagonist must try to advance in history to wake up from his dream.

The videogame was considered as a classic videogame, with the classic interfaces of the 90s and with the same interactions. The initial desired aspect was an 8-bit pixel adventure.

For the elaboration of the work, the Unity graphic engine has been used, specialized for new developments thanks to its facilities and unique properties. For the realization of what is desired, work will be divided into art, programming, sound and history.

Throughout the development the human limits and the lack of time have played against it, getting development to be forced to perform at the end of everything that did not comply with the initial approach or with the Gantt diagram with the development of tasks for time.

At the end and see how the day of the delivery arrived stepping on the heels tried to get quickly and running a functional test of the fundamental aspects that the game should offer. Despite having to do it in record time, the different interactions can be appreciated and the mechanics of the game are based.

Índice

1. Introducción

Tras un largo camino nos encontramos delante de la culminación del grado, entrando de lleno en el trabajo final, dónde pondremos en práctica todos los conceptos aprendido durante éstos años.

Cuando me pregunto como he llegado hasta aquí y el porqué estoy cogiendo la mención de Videojuegos del Grado Multimedia, todo se remonta a una sola respuesta y ésta está unos 25 años atrás, en mi primer contacto con ellos y en lo que daría comienzo mi pasión por los mismos. Y es que mis primeros pinitos con el octavo arte empezaban en un ordenador Windows 95 reproduciendo y disfrutando de lo que hoy por hoy es un género casi olvidado, las aventuras gráficas, por ello, quiero rendir homenaje con mi Trabajo Final de Grado a la causa por la cual hoy estoy escribiendo éstas palabras.

1.1 Contexto y justificación del Trabajo

Las aventuras gráficas son un género de videojuegos que tenían un éxito más que reconocido en la época de los 90 pero que actualmente no cuentan con él. Yo crecí con ellos y fueron mis primeros pasos en lo que al mundo de los videojuegos se refiere por consiguiente el trabajo quiere homenajear a modo de tributo a todo ello por lo que empecé a soñar y disfrutar y lo que ha hecho que esté realizando éste Grado Multimedia y este TFG en concreto. La aventura gráfica El Despertar está dotada de la interfaz clásica de estos juegos a modo de nostalgia, una interfaz que ha ido evolucionando con el tiempo y que actualmente está en desuso pero que se recupera para éste título.

El juego resultante quiere reflejar todas aquellas mecánicas clásicas dónde la esencia residía en los comentarios que iba realizando el protagonista que iban definiendo su personalidad. El poder avanzar en la historia a través de la observación e investigación del entorno y la interacción con el mismo. Pese a que el juego final no puede abarcar la totalidad de una aventura gráfica, en unas pocas pantallas quiere reflejar esa interacción y esa interfaz clásica que con tiempo y dedicación podrían derivar a un videojuego entero.

La historia te plantea a un protagonista que no es más que una heterónimo propio que está soñando y necesita despertarse porque debe presentar el TFG, el objetivo para llegar a conseguirlo es morir en el sueño para ello debe operar con su inventario y con el entorno que le rodea para poder conseguirlo.

1.2 Objetivos del Trabajo

El objetivo principal del trabajo es adquirir conocimiento dando unos pequeños pasos con la producción de un videojuego, viviendo la presión de los tiempos y la carga de trabajo realizado. Se quieren llegar a conseguir ciertos objetivos a lo largo del proyecto:

- Conseguir crear un diseño de personajes y animarlo
- Conseguir unos escenarios interactivos dotados de vida gracias a animaciones propias.
- Conseguir que cada interacción con los elementos posibles sea diferente lo que dará diversidad al juego.
- Conseguir realizar música de 8 bits, a menos una por cada escenario.
- Conseguir crear conversaciones entre personajes.
- Conseguir crear acciones causa-efecto en escenarios.
- Conseguir montar todos los elementos de manera correcta con el motor gráfico escogido (Unity).
- Conseguir crear un mini proyecto que pueda derivar en un videojuego completo con una mayor dedicación y una disponibilidad de mayores medios.
- Aprender.

1.3 Enfoque y método seguido

Siempre se dice aquello de que todo está ya inventado y que solo nos basamos en mejorar lo que ya existe pues en éste caso es algo a medias, cogemos un genero de videojuegos que ya existe, pese a que no esté en auge y se pretende crear algo nuevo bebiendo de la esencia de lo viejo. Con ésto podemos empezar con todo el conocimiento que hemos adquirido a través del ocio y disfrute de esos videojuegos pero a su vez podemos ayudarnos de el para utilizarlo de base y de sustento para crear algo propio y nuevo. Al tratarse de una aventura gráfica y al ser éste un género con un público muy especifico que aprecia la nostalgia se han incorporado elementos, comentarios y personajes de otros videojuegos de éstas características lo que puede ser un arma de doble filo pero que por nostalgia propia quiero añadir.

Teniendo en mente todos los aspectos relevantes del trabajo creo que la mejor estrategia es empezar a realizar los aspectos que tengo más por la mano y conocer mis limitaciones. Saber hasta donde puedo llegar y no querer abarcar más de lo imprescindible para poder entregar un videojuego que deje patente las interacciones básicas que definen los videojuegos clásicos de éste genero.

1.4 Planificación del Trabajo

Antes de lanzarnos a realizar cosas sin ton ni son debemos tener claro que contenidos serán necesarios a la hora de desarrollar el videojuego. Tenemos que tener en cuenta el apartado gráfico, el apartado sonoro, el textual y el montaje de todo junto, es decir, la lógica.

Apartado gráfico

Nuestro videojuego se basará en un producto de gráficos 2D dónde no habrá ningún tipo de modelo tridimensional, únicamente serán diseños de entorno y *sprites* de personajes para simular las animaciones de los mismos.

Como ya hemos adelantado previamente nuestra pequeña aventura gráfica estará compuesta de tres escenarios, por ello tendremos un diseño de escenario para cada uno de ellos, con un diseño común a modo de menú que será igual para los 3 escenarios. También tendremos diseños con animación de los personajes interactivos del juego, serán los elementos que más trabajo den puesto que al generar animación tendrán que tener una producción más detallada y elaborada. Por último también se realizará el diseño de los objetos dentro del menú de objetos.

Resumen:
- Gráficos del menú (compartido en todo el juego)
- Gráficos de entorno (para los 3 entornos de los que se compondrá)
- Gráficos para los *sprites* de los personajes (tanto protagonista como *pnj* interactivos)
- Gráficos objetos (apariencia que tendrán en el menú de objetos)

Apartado sonoro

Uno de los apartados que más dolores de cabeza nos va a traer puesto que no se por donde empezarlo. No quiero que la música sea de un banco de sonidos y música sino que por el contrario quiero crear desde 0 una melodía propia del juego para cada una de sus 3 estancias, por lo que deberemos componer 3 canciones en total. No solo eso, las canciones deben ser en formato MIDI el característico de los videojuegos de la época lo que le dará un extra de dificultad a la elaboración. Pese a ello y las claras limitaciones humanas de las que se basan en ésta parte si no podemos realizar 3 o se prevé con tiempo que no se cumplirán los plazos estimados deberá realizarse una única melodía para todo el videojuego.

Resumen:
- Melodía para la plaza principal

- Melodía para la carpa del mago
- Melodía para el circo

Apartado textual (guión)

Aquí entra un peso pesado del videojuego, puesto que las líneas de texto serán extensas. Cada interacción posible con los elementos del entorno debe contener unas líneas de texto. Si es cierto que muchas de ellas se repetirán para algunas de las acciones, un recurso muy empleado en éste tipo de videojuegos, como el “Eso no es posible”, pero ya que se trata de un pequeño trozo del videojuego real sería crucial que cada interacción entre objetos tuviera una frase propia. La estimación del tiempo y la producción definirán realmente como acabará siendo.
Otro punto muy importante son las conversaciones y las respuestas entre personajes. Éstas definirán la personalidad de cada una de ellos y son las que darán peso argumental al desarrollo de la historia por lo que el guión debe ser estudiado con detenimiento para conseguir crear una coherencia global. Si se creara un personaje que contestara de maneras muy diferentes quedaría un personaje mal planteado y mal definido por lo que debemos tener especial cuidado con éste tipo de detalles.

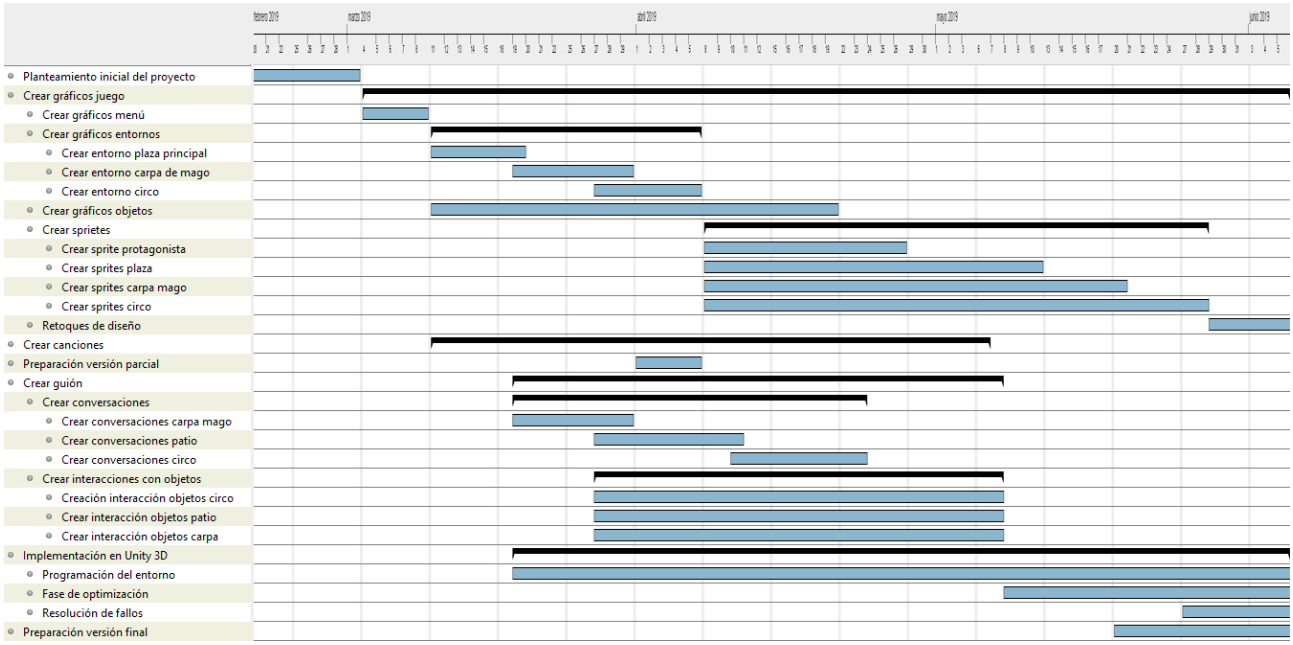
Resumen:
- Escribir conversaciones con los *pnj*
- Escribir iteraciones con los objetos del entorno

Implementación con el motor gráfico
La parte que más tiempo nos va a llevar puesto que se trata de unir todas las piezas con un motor gráfico el cual desconocemos e iremos experimentando a lo largo de este proyecto. No solo eso, también hay que tener tiempo para poder definir cada proceso y ver que no puede llegarse a un punto si no se pasa previamente por el anterior. Limpieza de posibles bugs y fallos en el desarrollo para poder conseguir un producto cerrado, limpio y acorde con la propuesta actual.

Resumen general
Lo que queda claro es que hay que ser realista y el tiempo es muy limitado y pese a que no se trate de un juego de una duración muy extensa, la cantidad de contenido que se pretende abordar hace que sea una producción muy ambiciosa que puede quedar en sueños rotos, para prever que eso no ocurra durante lo que dure el desarrollo habrá que tomar decisiones de recorte en el caso necesario siempre y cuando no se altere en una medida extrema el resultado final que se quiere conseguir.

Para poder planificar bien el tiempo empleado en cada una de los contenidos a desarrollar nombrados en el punto anterior procedemos a

crear un diagrama de *Gantt* que nos dará una mejor visión del tiempo a emplear en cada una de las tareas.



Como queda claro el tiempo es muy limitado y el nivel de trabajo es muy elevado. Siendo conscientes del planteamiento del semestre y personal va a ser todo un reto puesto que comparto el TFG con tres asignaturas más y trabajo jornada completa pero con perseverancia y constancia puede llegarse a cumplir los objetivos e intentar llegar a buen cauce a la finalización del semestre. Hay que tener especial atención a los tiempos que las tareas se superponen puesto que no hay que dejar descolgado ninguna de las tareas.

1.5 Breve sumario de los productos obtenidos

Al finalizar el TFG se obtendrá un ejecutable .exe con el producto final en el cual se podrá apreciar todo lo descrito anteriormente. De igual modo también estarán disponibles todos los *Assets* creados desde cero para el desarrollo. Interfaz gráfica, diseño de personajes, diseño de escenarios, animaciones, música, etc. Así como todos los archivos *script* que hagan posible las interacciones del videojuego.

1.6 Breve descripción del resto de capítulos de la memoria

A lo largo de la memoria se expondrán todos los elementos que se han ido desarrollando a lo largo del trabajo y se explicarán con detalle cada uno:

- Evaluación de diferentes *engines kits:* En el se hace un análisis de los motores gráficos principales para la iniciación en el desarrollo de videojuegos y se escoge en el que se va a trabajar.

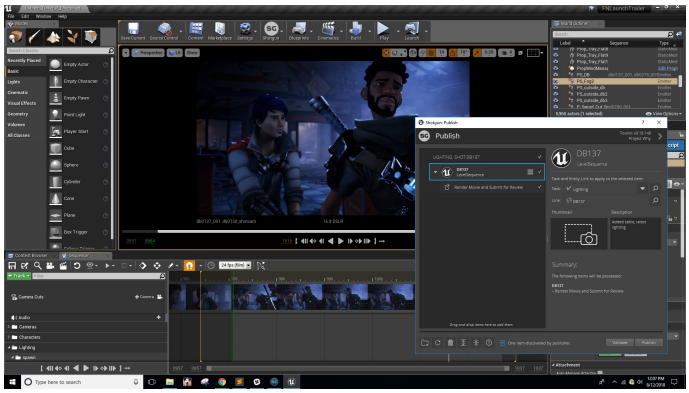
- Conceptualización: Se expone la idea original, la historia, la interacción con el jugador y la sinopsis del progreso.
- Subgéneros y referencias: Expone los videojuegos de los que bebe la idea y muestra los modelos a seguir.
- Arte: Se muestra el diseño de todo el videojuego, desde los primeros pasos hasta la evolución final, desde la interfaz hasta los personajes.
- Animaciones: Se muestra las diferentes animaciones que se pueden encontrar en el entorno.
- Apartado sonoro: Se expone como se ha realizado la producción de todas las músicas del videojuego.
- Montaje de la escena en Unity: Se explica como se utiliza la jerarquía de las capas y como se distribuyen para el correcto funcionamiento del juego.
- Programación: Se explica en detalle todos los comportamientos y las mecánicas del juego.
- Metas logradas y metas perdidas por el camino: Tal y como lo explica, se exponen todos los objetivos logrados o los que no se han podido llegar a alcanzar.
- *Bugs* y carencias por falta de tiempo: Expone los errores corregibles que no se han podido solventar por falta de tiempo o conocimientos.

2. Evaluación de diferentes *engines kits*

Antes de nada debemos concretar que motor gráfico hará posible la existencia del videojuego y para ello debemos analizar los que utilizaron los predecesores y las herramientas más accesibles actualmente.

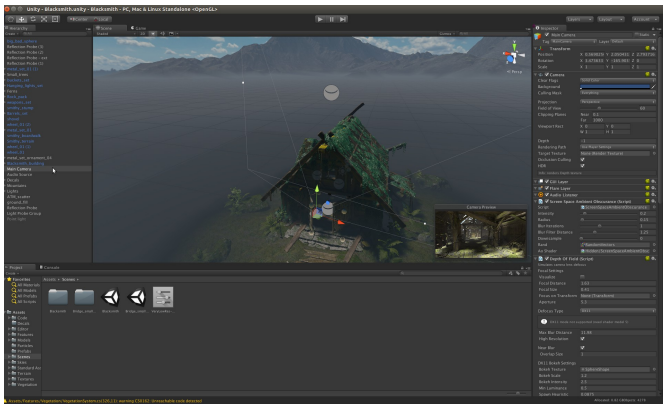
2.1 Unreal Engine 4

Un motor gráfico de calidad muy utilizado en la creación de videojuegos gracias a la versatilidad que ofrece y la facilidad a la hora de programar sobre él. Es un motor que demuestra una gran optimización en los recursos sin necesidad de una potencia muy elevada. Su código está escrito en C++ lo que lo hace de fácil acceso al igual que es un motor gratuito para la creación de obras propias por lo que lo hace una opción muy atrayente para experimentar y para poder implementar nuestro juego con él.



2.2 Unity 3D

Motor que ha ganado fama entre los desarrolladores más principiantes, para empezar es gratuito siempre que compilemos el juego para ser ejecutable en Windows, lo que como hemos expuesto en el primero de los puntos, nos viene al caso ya que será nuestra plataforma de destino. Lo bueno del motor es la cantidad de tutoriales y cursos que pueden encontrarse a la vez de que es un motor con el que se pueden conseguir resultados increíbles con poco conocimiento. Es un motor óptimo para principiantes que quieren crecer profesionalmente a la par que adquieren habilidades. En su última versión se puede disfrutar gratuitamente de todas las funciones que tendríamos con la versión de pago.



2.3 Shiva Engine

Un motor gráfico no tan conocido como los dos anteriores pero que dispone de unas prestaciones similares a Unity gracias a sus constantes actualizaciones, cierto es que no es un motor escuchado ya que no ha salido ningún proyecto de “calidad” que lo haya utilizado pero no por eso lo hace menos operativo. Por parte negativa su interfaz se aleja de la típica que utilizan los dos anteriores lo que hace que al principio pueda resultar confuso pero que si es utilizado con esmero puede dar muy buenos resultados.

Es un motor gratuito y puede desarrollar juegos para MAC, Linux, Windows, IOS, Android, etc...

Pese a que todos los motores gráficos tienen sus virtudes y sus defectos, por el tipo de versatilidad que ofrece y por la facilidad de encontrar información del mismo en Internet en caso de que nos encallásemos a la hora de programar algún aspecto, **escogemos Unity 3D para el desarrollo del TFG**. Pero aunque tengamos escogido el pegamento que lo unirá todo todavía nos queda concretar que va a ser ese todo.

3. Conceptualización

Para un videojuego de aventura gráfica su punto fuerte es la creación y personalización de cada uno de los elementos que se disponen y poder hacer sentir al jugador que es el responsable de avanzar en la historia y que los sucesos ocurran como ocurren. Un buen entorno, unos personajes con una personalidad muy marcadas y un fuerte grado de humor son elementos clave para conseguir una buena aventura gráfica, por ello el guión cobra un peso tan importante donde la situación más absurda puede resultar la más gratificante.

3.1 Historia, ambientación y/o trama

Dado el tiempo y las limitaciones temporales y humanas el proyecto no podrá adquirir las
dimensiones que en un principio nos gustaría. Por lo que la historia vendría siendo un prologo a un hipotético videojuego completo.
La trama es simple, el protagonista se encuentra en un sitio que desconoce, parecido a un bazar fantástico cuando de repente suena de fondo un sonido estridente, no es otro que una alarma, la alarma de su despertador. Si, se encuentra dormido y tiene que despertarse porque debe presentar su trabajo final de Grado a primera hora de la mañana, pero no hay ,manera de despertarse, no le queda otra que poner todos los medios a su alcance para intentar conseguirlo. Sabe que para despertar en los sueños hay algo que no falla, morir, por lo cual tendrá que intentar a toda costa morir en el sueño para poder llegar a tiempo a la presentación del TFG y no suspenderlo.

3.2 Definición de personajes/elementos

El personaje principal (protagonista) se irá definiendo según las interacciones con los elementos de su entorno y las conversaciones con los PNJ (personajes no jugables) del videojuego. La gracia y el planteamiento que tiene éste tipo de videojuego es que a través, sobretodo de las conversaciones el protagonista va definiendo su personalidad y así el jugador puede hacerse una imagen de como es. Los personajes no jugables también tendrán una personalidad muy definida que es lo que le da dinamismo y viveza a la aventura. El videojuego, o el prologo, como hemos comentado anteriormente, estará definido por 3 entornos por los que el personaje podrá moverse libremente. **La plaza principal, la carpa del mago y el circo**, en cada una de ellas habrá varios elementos interactivos, todavía por definir, muchos de ellos sin valor real en el avance de la aventura pero que darán solidez al juego final al igual que añaden dificultad a la hora de resolver los puzles que van apareciendo puesto que hay mas combinaciones posibles.

Personajes que nos encontraremos en los diferentes escenarios (sujeto a cambios):

Plaza Principal:

- Tendero

Carpa del mago:

- Mago

Circo:

- Animales enjaulados
- Hombre forzudo
- Domador
- Payaso

3.3 Interacción entre los actores del juego

La interacción con los personajes que hemos adelantado en el punto anterior serán meramente conversacionales, algunas añadirán contenido y harán conseguir al protagonista tanto objetos como información que le harán capaz de poder avanzar en la historia.
El peso de las conversaciones en el videojuego serán muy importante por ese motivo debe haber atrás de el un trabajo elaborado de guión y de posibilidades dado el peso que tiene éste en el transcurso de la historia. Por ese motivo tendremos que definir y escribir durante mucho tiempo las diferentes líneas de los personajes por lo que tendremos que tener muy en cuenta éste punto a la hora de la organización del tiempo.
La gracia del juego será que pese a que el objetivo quedará claro desde el principio, morir, y a más de que en pantalla habrá elementos

claramente referenciados para ello, como armas afiladas, acantilados profundos, etc; ninguno de éstos elementos nos harán llevar al fin, sino que por el contrario acabaremos llegando al objetivo de la manera más cómica y rara posible.

3.4 Objetivos planteados al jugador

La peculiaridad que tienen las aventuras gráficas es que si supieras lo que hay que hacer son videojuegos que podrían terminarse en tan solo unos pocos minutos, y éste es el mismo caso, por el contrario las posibilidades de combinación de objetos y conversa son altos para darle un grado de dificultad al jugador para que pueda inspeccionar y tenga que exprimirse la cabeza para llegar a la conclusión de como avanzar.

En el videojuego a conseguir se añadirán muchos *easter eggs* a videojuegos de la época y referencias a modo nostálgico. Plantearemos el juego para poder avanzar en **3 secuencias**. Tras el primer discurso y poniendo al jugador en situación y dejando claro el objetivo a cumplir estaremos en la plaza principal, desde ella podremos acceder a la carpa del mago o al circo. El personaje solo tendrá en su inventario dinero.

- **1ª Secuencia:** El jugador tendrá que entrar en la carpa del mago (Timon the sorcerer) y entablar puzle conversacional con él. Al final se pondrá nostálgico y nos pedirá si puede probar a volver a usar el control, al asentir, utilizara magia para controlar el puntero y pulsará Dar Dinero a Timon the sorcerer, lo que hará que el personaje automáticamente le de su dinero sin poder evitarlo. A partir de ello y de exponerle las quejas justificadas él nos recompensará con polvos de levitación mágicos. También cogeremos un libro de chistes malos antes de salir de la carpa.

El planteamiento quedará de la siguiente manera:

-Dinero +Polvos de levitación mágicos +Libro de chistes malos

- **2ª Secuencia:** De vuelta en la plaza principal el jugador deberá conseguir fruta de un puesto que hay al final de la misma pero el tendero no permitirá al jugador hacerse con ella, para ello deberá crear una distracción. Justo detrás del puesto de fruta habrá un club de alterne vallado, el jugador deberá utilizar los polvos obtenidos anteriormente en las cortinas de la ventana del club lo que hará que el tendero se quede mirando tras ella sin echar cuentas de la tienda, en ese momento el jugador podrá coger la fruta que quiera.

El planteamiento quedará de la siguiente manera:

-Polvos de levitación mágicos +Fruta

- **3ª Secuencia:** Ésta vez el jugador deberá ir a la zona de circo, ésta será la zona donde podrá morir, para ello debe provocar al forzudo para que le pegue, pero ni con esas será suficiente, por ello deberá darle la fruta (plátano) a los monos de la jaula y éstos se lo comerán y dejarán la piel en el suelo. Después debemos hacer enfurecer mucho al forzudo para ello deberá averiguar su debilidad, para ello tendrá que hacer que el bufón se lo cuente, para ello tendrá que hacerle reír antes, para poder conseguirlo el jugador deberá leer previamente el libro para aprender algunos chistes, de ésta manera podrá contarle alguno que le haga gracia y le contará que el forzudo usa peluca y que su melena es falsa. Una vez el jugador tiene toda la información éste tendrá que tirar del pelo al forzudo lo que hará enfurecerle y que pegue tan fuerte al protagonista que hará que retroceda mucho lo que le hará tropezarse con la piel de plátano y morir desnucado cumpliendo así el objetivo planteado al jugador.

El planteamiento quedará de la siguiente manera:

-Fruta **Objetivo Cumplido**

4. Subgéneros y referencias

Como se exponía anteriormente el juego estaría englobado en el genero de aventura gráfica, un género de gran auge en los años 90 en el cual las aventuras del subgénero conocido como *point & click* estaban en boca de todos. En éste tipo de videojuegos no premiaba la habilidad y destreza milimétrica a la hora de tomar decisiones sino que eran pequeñas obras de arte donde el desarrollo de la historia era el plato fuerte del juego. Los Game Designer podían desarrollar la personalidad de los personajes única con una libertad y un mimo que quedaba palpable en el resultado final. Creo que es muy necesario destacar en éste punto el papel que tuvo la empresa LucasFilm Games, posterior Lucas Arts que Disney decidió su cierre cuando la adquirió en 2012. Ésta nos dejó joyas como Maniac Mansion, Monkey Island o Sam & Max. Justo en algunas de éstas obras nos referenciaremos a la hora de producir nuestro proyecto.

Pero en un trabajo como el que tenemos entre manos en el que la nostalgia tiene un peso tan importante no tienen que quedarse en el tintero otras grandes producciones de las que se va a beber mucho a la hora de tomar un camino u otro. Videojuegos con los que crecí y con los que pasé horas de diversión y quiero arrastrar un poco del humor característico que tenian éste tipo de videojuego en mi proyecto final. Sin duda alguna éstas son tres sagas de videojuegos; La saga Monkey Island (LucasFilms Games), La saga Simon the sorcerer (Adventure Soft) y Leisure Suit Larry (Sierra Online).

The Secret of Monkey Island (1990) *Simon the sorcerer (1993)*

Leisure Suit Larry 6 Shape Up or Slip Out! (1993)

5. Arte

Al tratarse de un videojuego en 2D todo el contenido serán gráficos, todos los elementos serán *Sprites.* Algunos de ellos únicamente estáticos y otros con diversas variantes para someterse a posteriores animaciones. Al final a lo largo del desarrollo del videojuego solo se han podido realizar dos de los tres escenarios que se tenían mente, por ese motivo todos los diseños completos al 100% se basan en los elementos que si hemos logrado plasmar en el videojuego final.

5.1 Interfaz

Tal y como hemos comentado con anterioridad una de las características típicas del genero de aventura gráfica de los años 90 era su interfaz. Una interfaz que venía heredada de las antiguas historias escritas y por ese motivo el contenido textual tenía un mayor peso que el simbolismo, caso totalmente contrario al de hoy en día.

Para la interfaz el menú de acciones es el elemento que se comparte en todas las escenas del videojuego (excepto del menú principal) éste recoge todas las acciones posibles que puede realizar el protagonista

del juego con los elementos del escenario. Ésta engloba una tercera parte de la totalidad de la pantalla de juego por lo que es un elemento muy importante en él. Ésta se divide en dos partes, la parte izquierda está comprendida por las acciones anteriormente nombradas y la parte derecha contiene los espacios para los objetos que el protagonista lleva encima, es decir su inventario. Éstas dos partes están separadas por dos flechas que se activarán cuando existan suficientes objetos en el inventario para tener que pasar para arriba y para abajo los elementos, dado de que en nuestro caso no se dará no tendrán funcionalidad. La parte de arriba del menú se completa con una franja negra donde saldrán escritas las interacciones que se pueden realizar.

Tal y como podemos observar se distribuyen los elementos en el menú de acciones (izquierda: acciones, medio: flechas, derecha: objetos y arriba: descripción de las acciones)

La idea principal y uno de los objetivos que no se ha podido llevar a cabo al final era conseguir que el aspecto de nuestro videojuego fuera pixelado, para ello queríamos utilizar un filtro que pixelara cada elemento individualmente y que a la hora de combinarlos todos por capas diera el efecto deseado. Por falta de tiempo y algunas pruebas con efectos raros ha hecho que se desestimara la idea aún así el elemento de la interfaz comparte éste aspecto gracias a la fuente utilizada en el texto, lo que le da ese toque.

Cada botón del menú de acciones tiene tres modos, el modo normal (el predefinido y como aparecería con nuestro menú sin tocar), el modo *hover* (el aspecto que coge cuando entra el ratón dentro de él) y el estado de pulsación (el estado que toma el elemento cuando es pulsado (éste último es casi imperceptible y podría haber sido eliminado sin crear un gran efecto)).

Los diferentes estados en los que aparecerán las acciones (De izquierda a derecha: normal, *hover* y pulsado).

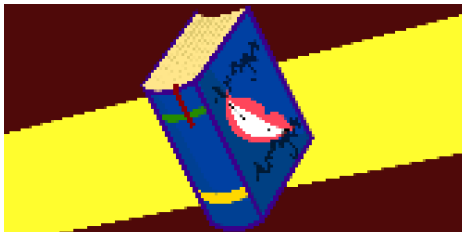
5.2 Objetos inventario

Desde el principio del desarrollo tenía claro cuales iban a ser los objetos interactivos del inventario por ese motivo fueron los primeros elementos

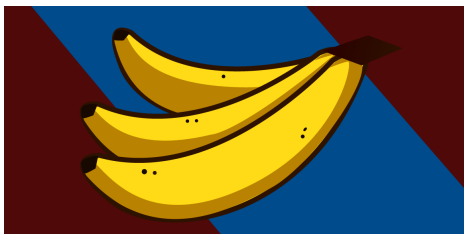
en realizarse. Éstos están realizados con el programario de ilustración vectorial Adobe Illustrator. Al ser elementos realizados al principio del TFG comparten el aspecto pixelado del menú de acciones. Los elementos a realizar han sido el dinero, el libro de chistes, el plátano y los polvos mágicos. Todos ellos se presentan en sus dos variantes, vectorizados y pixelados.

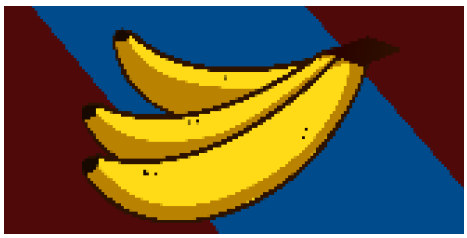
Dinero

Libro de chistes

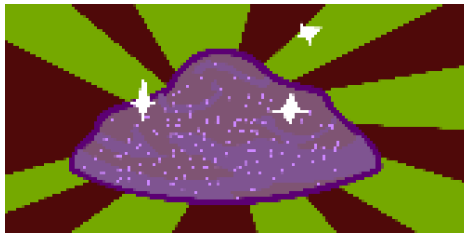


Plátano





Polvos mágicos



Una vez tenemos todos los elementos del menú de acciones y los objetos de inventario juntamos los elementos para visualizar como

quedan unidos. Éste resultado es semejante a los juegos de antaño gracias al efecto pixel.

5.3 Escenarios del juego

Por desgracia el tiempo ha jugado durante todo el desarrollo en mi contra y la meta principal de poder cubrir 3 escenarios se vio reducida a 2. Pese a ello se ha querido llegar al máximo de calidad de cada uno y se ha tenido que ir modificando a lo largo del desarrollo. Para poder subsanar esa carencia del tercer escenario se desarrolló un menú principal que aunque pueda parecer obvio no se había planteado en el inicio de desarrollo puesto que los videojuegos antiguos no contaban con él sino que te lanzaban a la aventura gráfica sin más.

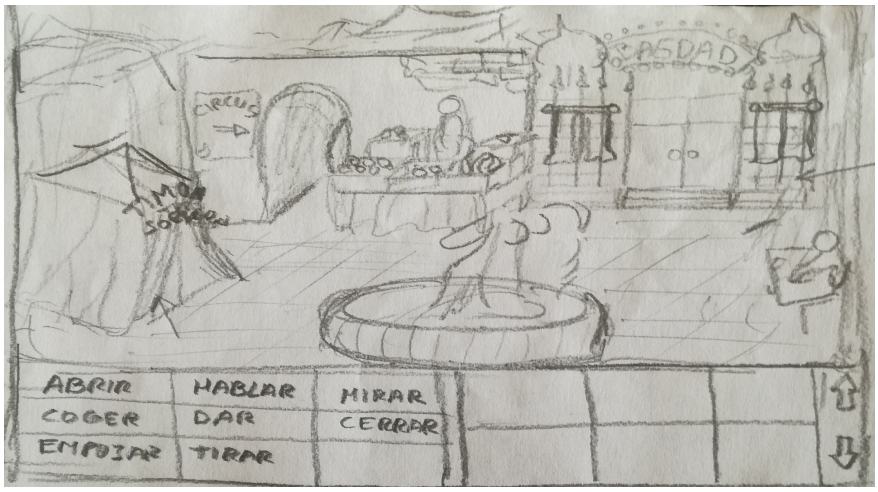
Volviendo a los escenarios, el primero que se concibió fue el principal, la Plaza, el diseño se inició con un pequeño croquis en papel que fue evolucionando hasta el diseño final. Durante el desarrollo se intentó realizar el diseño con Photoshop. El resultado desastroso que quedó obligo a rehacer la idea desde el inicio con el programa que más acostumbrado estaba y que había empleado para crear los objetos de inventario, Adobe Illustrator. Con ésto abandonaba la idea de crear un juego pixelado, puesto que a la hora de realizarlo, algunas capas creaban efectos extraños que solapaban colores. Los diferentes elementos están generados en diferentes capas porque a la hora de montarlo todo en Unity nuestro personaje puede caminar dejando algún elemento por encima y otros por debajo. Uno de éstos casos fue el elemento fuente, el cual creaba un efecto raro y no quedaba bien, por no perder un tiempo que era muy valioso desestimé la idea para implementarla más adelante si se pudiera dar el caso.

Para el segundo escenario, la tienda del mago, ya veníamos de generar el diseño de la plaza, habíamos aprendido de los errores y por ello el tiempo empleado entendíamos que iba a ser menor y así fue. La no necesidad de hacerlo antes en otro programa y el tener la idea en la cabeza de lo que deseaba hizo que pudiera realizarlo más eficazmente. Podemos apreciar que en cada uno de los dos escenarios, los elementos no responden a unas proporciones reales, creando malformaciones y construcciones poco lógicas, éste efecto es intencionado y quiere asemejarse al encanto que tenían los diseños de escenarios del juego Day of the Tentacle, dónde pese a que los elementos no tienen sus dimensiones típicas, se puede vislumbrar con todo detalle que es cada elemento.

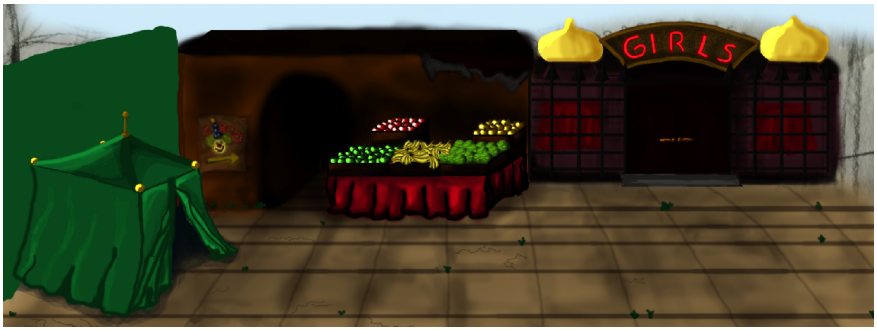
Por desgracia el tercero de los escenarios se tuvo que quedar en el tintero con lo que únicamente podemos compartir de él el boceto inicial. Éste también está incluido en el juego a modo de broma, puesto que en el planteamiento inicial si estaba estipulado que estaría.

Evolución a lo largo del proceso de los escenarios:

- Plaza



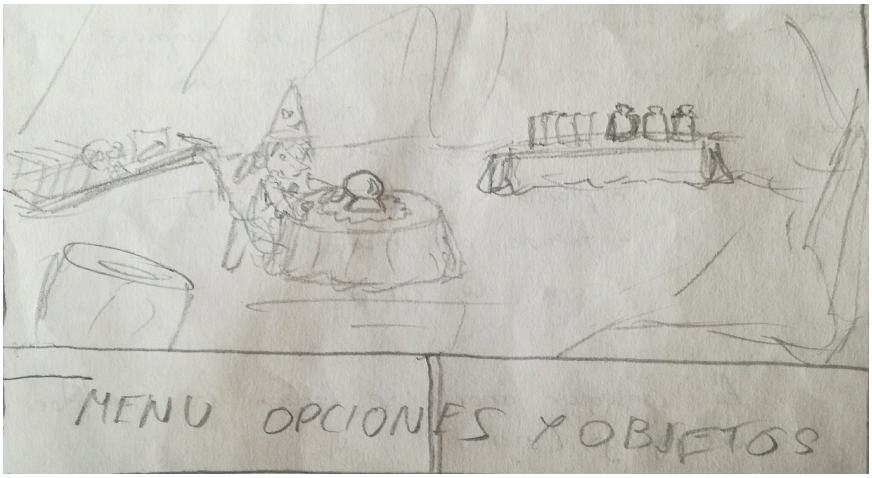
Boceto inicial



Primer diseño con Photoshop

Diseño final

- Tienda del mago



Boceto inicial

Diseño final

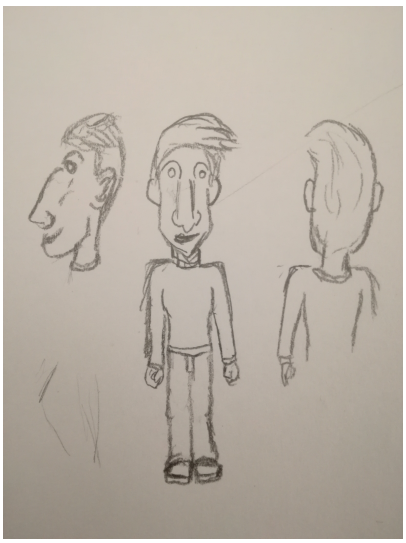
- Circo

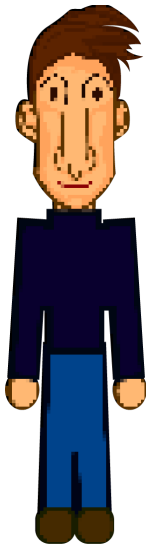
Boceto inicial (no realizado en el proyecto final)

5.4 Diseño de personajes

En el juego final existen 3 personajes destacables, uno de ellos y el que más animaciones tiene es el protagonista. Siguiendo la línea *cartoon* quería darle malformaciones y proporciones agresivas destacando las imperfecciones. Dado que el personaje principal es como una caricatura mía propia le he pronunciado lo que más grande tengo, la nariz. Dejando a un lado ya el intento del aspecto pixel me he dedicado a realizar un personaje de aspecto cómico vectorizado el cual me resultara fácil de animar posteriormente. Originalmente todos los diseños, como en el caso de los escenarios, están basados en bocetos lo cual facilita encarecidamente la realización del diseño directamente en el programario.

Protagonista del juego, sin nombre, pero semejante a mi; al igual que en el caso de la plaza, el diseño fue elaborado para una de las primeras entregas por tal de tener un objeto que funcionara a modo de personaje, por ese motivo es el diseño que ha pasado por tres fases distintas.





Evolución del diseño del personaje (de izquierda a derecha; izquierda: boceto inicial; en medio: diseño de entrega para la segunda PAC, derecha: diseño final)

El tendero es el segundo de los personajes principales del juego, el diseño se basa en un tendero árabe del desierto con complexión delgada y cabeza grande escondida bajo el turbante, éste es el vendedor de la tienda de fruta, la cual inicialmente estará cerrada. El diseño de éste personaje no está dotado de pies ni manos puesto que como está tras de un mostrador éstos elementos no se muestran en ningún momento y a modo ahorrativo de tiempo no se han realizado. Éste diseño pasó directamente del boceto al diseño final empleando la misma técnica que en el personaje protagonista, se pasó el boceto al ordenador y se empezó a vectorizar sobre él.

Del boceto inicial al diseño final del tendero

Por último tenemos el mago Timon, que aparece en la tienda. Éste personaje hace referencia al personaje Simon the Sorcerer de las aventuras gráficas con homónimo nombre. Es uno de los personajes que se han diseñado con más cariño dado la nostalgia que se tenía sobe él y pese a que el boceto no refleja el estado que se quiere conseguir, al final se consigue el diseño deseado. Como en el caso del tendero, a Timon no se le han dibujado las piernas puesto que en su ubicación no se le ven puesto que está sentado tras una mesa. Para la realización de sus animaciones se le ha juntado con la mesa porque en un principio una de las animaciones(que al final no se ha podido crear) la necesitaba.

Del boceto inicial al diseño final de Timon el hechicero

5.5 Otros diseños

Para completar el entorno y los escenarios se han realizado otros elementos puramente decorativos, mucho de ellos realizados para ser animados posteriormente desde el mismo Unity. Entraremos en detalle en ellos en el apartado animaciones.

6. Animaciones

Cualquier juego que se precie contiene múltiples animaciones, éstas otorgan dinamismo al videojuego y juegan un papel muy importante en el desarrollo de los mismos. En nuestro videojuego las animaciones cambian mínimamente, al tratarse de un videojuego en 2 dimensiones, las animaciones serán *Sprites* de los elementos gráficos que varían con el tiempo. Ésto conlleva que el desarrollo gráfico juega un papel más importante que en otros juegos puesto que el número de imágenes empleadas aumenta exponencialmente. Cuando tenemos un videojuego en 3 dimensiones, el modelado del personaje se mueve a través de unos huesos y estos son los que llevan la animación a través de la linea de tiempo, para éste caso serán todas imágenes.

6.1 Animaciones del menú

Anteriormente ya habíamos adelantado que como decisión final hemos incorporado el diseño de un menú principal. Éste no lo hemos adelantado en el apartado del arte porque es un diseño enteramente animado, con él se intenta dar un poco de viveza al inicio del juego y ya que al iniciar la partida te lleva directamente al escenario de plaza sin

ejecutar la animación de inicio, que pese a estar creada no ha podido estar implementada, éste funciona a modo de primer filtro para que la sensación no sea extraña, o más extraña aún.

El menú está compuesto a base de capas, cada una con una animación propia que se mueve hacia direcciones diferentes. La primera de las capas, o la más cercana al usuario es la capa del logo, ésta varia su tamaño y decrece en un factor de tiempo determinado. Todas las animaciones, cabe decir, que se han realizado internamente dentro del mismo kit de desarrollo Unity, utilizando la linea de animación y variándola. Para poder apreciar la animación debemos ser conscientes que el botón de grabación debe estar activado. Cada vez que queremos que un objeto varíe añadimos una *key* con el valor del cambio, en nuestro caso como hemos comentado hemos escogido agrandamiento y encogimiento sucesivo, para ello solo debemos hacer que el agrandamiento y el decrecimiento se repita una vez por *keys* y especificar que esa animación debe estar ejecutándose siempre (es la opción que viene por defecto).

Para poder modificar el tamaño a antojo hay que ir cambiando entre la ventana de animación y *scene* así podemos tener un retrato real de la animación.

Una vez animado el logo se anima la segunda de las capas, el marco, que queda alrededor de la pantalla y éste se moverá girando y recuperando su posición inicial. Todas las animaciones las generamos del mismo modo siempre desde la ventana animation de Unity.

En tercer lugar tenemos una animación que es un poco diferente al resto, tenemos una animación que no se mueve sino que cambia de sprite 2d pasado un determinado tiempo. Ésto nos crea un efecto de que bombilla que está iluminando se enciende y apaga intermitentemente. Todo ésto está maquillado con una animación falsa de aumento de texto al pasar el ratón por encima de las opciones que se genera a partir de un archivo *script.*

El menú también tiene un objeto más que está animado, lo que éste elemento no aparece hasta pulsar la opción de créditos. Éste, como especifica el botón, son los títulos de crédito, los cuales aparecen por abajo y se pierden por la parte superior. Siguen el proceso de la generación de animaciones explicadas anteriormente.

6.2 Animaciones de los elementos del escenario

Una de las cosas que tenía claro a la hora de realizar éste Trabajo es que quería que los escenarios tuviesen vida. Con vida me refiero a que no fueran puramente estáticos y con falta de dinamismo, por éste motivo quería que aparecieran varios elementos animados que dieran la sensación de que el escenario corría y que el tiempo pasaba. Éstos elementos no tenían que tener repercusión en la historia ni tener mayor trascendencia, es más, no serían ni interactivos, pero por el contrario si que darían calidez a la escena.

En el escenario de la plaza encontramos distintos objetos animados empezando por una de las capas por la cual está compuesto el escenario. Ésta tiene el letrero luminoso del club de alterne el cual cambia de color cada x tiempo. La animación la tiene la propia capa la cual cambia de imagen. Por otro lado podemos encontrar la fuente, la cual crea el efecto de que corre el agua. Éste efecto se consigue a través de 3 imágenes distintas que se cambian sucesivamente y generan dicho efecto. Por último tenemos las nubes que se generan por el cielo. Son elementos estáticos pero que a través de una animación por *keys* se desplazan de derecha a izquierda haciendo que cuando estemos en el escenario se vean pasar las nubes.

A parte de las animaciones que tiene el escenario para dotarlo de vida también hay animaciones que se dan por realizar diferentes acciones. Éstas permanecen inmóviles y desconectadas y al realizar dicha acción se activan, como por ejemplo la cortina que se eleva cuando utilizamos los polvos mágicos. Cuando lo hacemos la cortina se eleva y muestra la animación del elemento *censored* por encima como cubriendo un acto sexual que se está dando dentro de el club de alterne. Todas éstas animaciones se consiguen a través de diferentes imágenes que cambian con la linea de tiempo y se activan a través de *scripts.*

Hay algunos elementos animados en el escenario Plaza que se han quedado en el tintero y al final no han sido utilizados en la versión final del juego en ellos profundizaremos en el apartado metas logradas y metas perdidas por el camino.

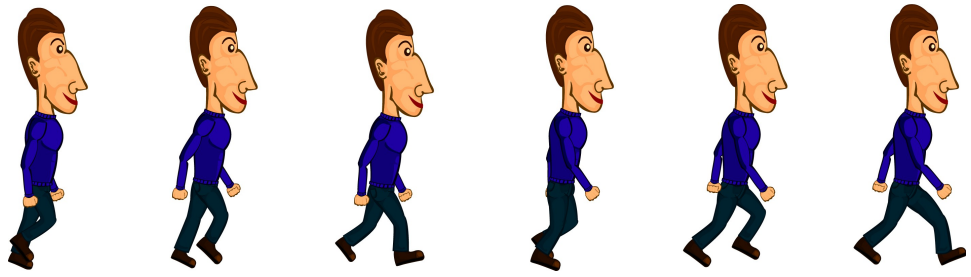
En el escenario Tienda del mago podemos encontrar diferentes elementos animados, pese a que a primeras su apreciación es mínima, ésta ayuda a ambientar el escenario. La primera de ellas es la vela, o velas, las dos velas tienen una animación secuencial compuesta por tres imágenes donde se ve como la llama cambia de forma. También y ligado a ésta animación tenemos el resplandor de las velas el cual está directamente pegado a las capas que componen el escenario, éstas capas cambian de imagen en un periodo muy corto de tiempo generando el efecto de la luz ampliándose y encogiéndose por el movimiento de las llamas.

Para conseguir que las llamas no realicen el mismo movimiento simultáneamente se ha empezado la animación de la vela izquierda y la vela derecha por imágenes diferentes.

También podemos encontrar una araña que se descuelga por su telaraña. Éste elemento está en la capa más cercana al usuario por lo que tapa al protagonista y da la sensación que está en la pared más cercana al espectador. Ésta animación es una sola imagen en la cual se ha modificado su posición a lo largo del tiempo.

6.3 Animaciones de los personajes

Otra de las cosas que dan fluidez a la escena y sobretodo son muy importantes porque son las que nos ayudan a comprender lo que está ocurriendo en escena son las animaciones de los personajes, tanto el protagonista principal como los secundarios que aparecen en escena. En nuestro videojuego hemos realizado varias animaciones según sus características empezando con el protagonista.

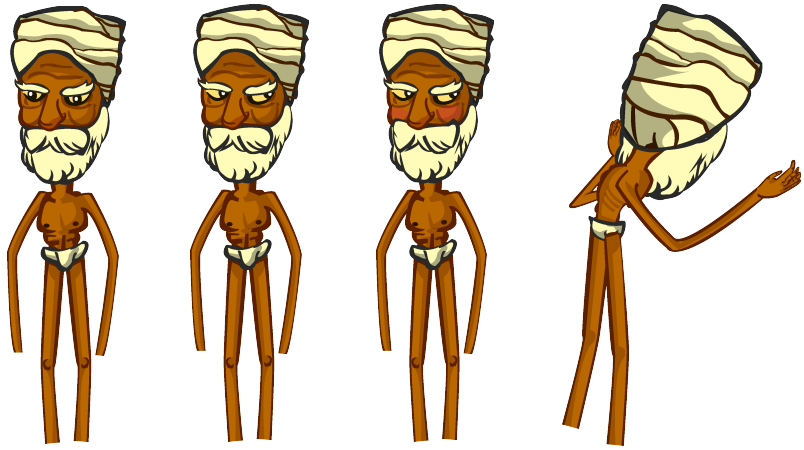


El protagonista tiene 7 animaciones en total, éstas definen su comportamiento en todo momento:

- Caminar izquierda
- Caminar derecha
- Hablar izquierda
- Hablar derecha
- Quieto izquierda
- Quieto derecha
- Inicio (no utilizada)

Por desgracia la animación de Inicio se debía inicializar cuando empezaba el juego y mostraba a modo de vídeo la presentación y explicación de lo que estaba ocurriendo pero no ha podido emplearse por falta de tiempo al igual que la depuración por *script* del resto, aun así están creadas y su funcionamiento es correcto.

Por otro lado tenemos el personaje del tendero el cual tiene 4 animaciones las cuales se basan en las acciones que puede realizar, puesto que el tendero estará estático no haremos diferenciaciones de izquierda y derecha.



Las animaciones posibles con el tendero son:

- Pestañear(la animación por defecto)
- Hablar(cuando habla con el personaje)
- Mirar(cuando se ejecuta la acción de los polvos con la cortina)
- Mirando(la animación que se queda por defecto después de la acción)

Debo admitir que ésta animación me dio quebraderos de cabeza puesto que a la hora de ampliar la imagen se variaban los *anchor* y los posicionamientos de toda la imagen lo que hacia que se modificara toda la animación y no solo lo que quería. Hay que tener especial atención y cuidado a la hora de crear animaciones en el cual se modifique el tamaño de la imagen de un golpe a otro puesto que si está grabando se generarán *keys* donde no deseamos y el resultado final variará del esperado.

Por último tenemos las animaciones de Timon el hechicero, éstas están solapadas junto con la mesa por que originalmente iba a tener una animación extra que en la versión final no ha sido generada. Ésta iba a ser una en las cuales Timón agitaba sus manos delante de la bola, para poder robarte el dinero.

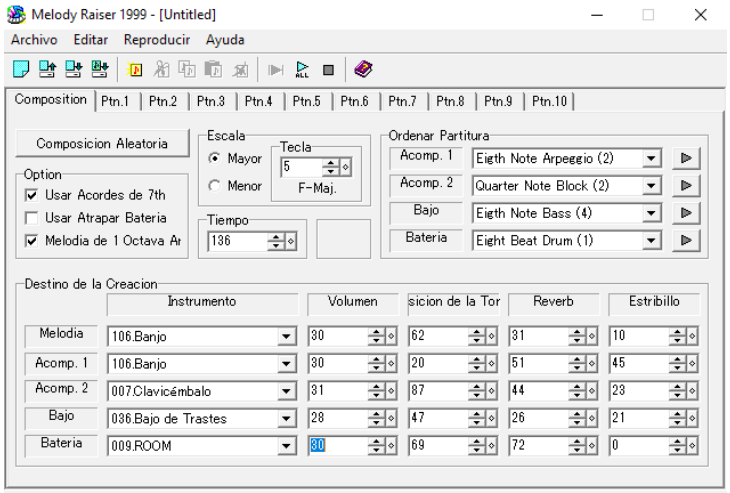
Timon está compuesto de dos animaciones:

- Pestañear(animación por defecto).
- Hablar(se ejecuta cuando hablamos con el personaje).

7. Apartado sonoro

Dado que el juego originalmente estaba pensado para ser un videojuego pixelado quería que las músicas fueran también de la época, es decir en formato midi y compuestas como antiguamente. Como mínimo quería generar una música para cada escenario que se fuera reproduciendo en bucle y que no dejara una sensación de vacío al escenario.

Mi conocimiento de música es limitado pero no quería coger algo que ya estuviera realizado y sin derechos, éste Trabajo lo he entendido como una obra entera mía y no quería concebir que ningún elemento era cogido externamente por eso todos los elementos de los que se compone el resultado final son propios y la música no iba a ser menos por ese motivo indague para conseguir un programario que generaba melodías en 8 bits. El programario en cuestión se llama Melody Raiser 1999. Es un programa de desarrollo japones con el que generar las melodías deseadas con acordes y acompañamientos deseados.



El funcionamiento es fácil e intuitivo se crea una composición dónde eliges Que instrumento o elemento generará la melodía, cual será los acompañantes (pudiendo elegir 2), el bajo y la batería, a cada elemento le especificas como debe tocarse y con que frecuencia y que importancia o ganancia va a tener en la música final. Después debemos definir el tempo al que será tocada y la escala en la que estará regida, después de ello debemos crear páginas de partituras las cuales se irán añadiendo una tras otra en la canción y en ella debemos especificar los acordes en los que sonará la melodía y especificar las notas que sonarán. Al ser un programa japonés las notas están en katakanas pero son fácilmente reconocibles. Después de estar usándolo y toqueteando unos días comprendí su funcionamiento y pude generar tres melodías diferentes, una para el menú, otra para la plaza y una última para la tienda.

7.1 Música del menú principal

Ésta canción es la que se ha creado con más consciencia y con más sentido en connotación con el videojuego. Al iniciar el juego suena el repicar de un reloj, como que el tiempo está pasando y haciendo la similitud con el sonido de los despertadores antiguos. Una vez pasadas las notas iniciales que van en progresión arranca la melodía con una serie de notas que evocan problemas y que hay que solventarlos, sabes que la cosa no está bien.

También se ha utilizado el sonido del reloj inicial para cortarlo y generar dos sonidos independientes el cual se ejecuta al entrar en las letras del menú principal.

Para reproducir la melodía del menú pulsar aquí: 

7.2 Música de la plaza

Para la música de la plaza se quería conseguir todo el efecto contrario que en el menú, se quería evocar tranquilidad y viveza, a modo más alegre, con notas más separadas y con un ritmo mucho más ameno, la duración de las músicas es corta y al ser sonidos MIDI reconvertidos en archivos mp3 ocupan realmente poco peso

Una vez terminado podemos decir que el resultado es muy mejorable en todos los aspectos pero como he comentado anteriormente deseaba que el 100% del juego fuera obra propia por consiguiente tenía que tener presente mis fortalezas pero sobretodo mis debilidades.

Para reproducir la melodía de la plaza pulsar aquí: 

7.3 Música de la tienda del hechicero

Quería seguir la pauta de la plaza, una música mucho más alegre pero con unos tonos más agudos simulando el sonido de la magia. Ésta el la canción que más dura puesto que al inicio del juego es el lugar donde más tiempo pasaremos porque hay más interacción y no quería que la melodía se hiciera más repetitiva de la cuenta por ese motivo las notas y los acordes van y vienen con una mayor frecuencia.

Para reproducir la melodía de la plaza pulsar aquí: 

Una vez terminados los elementos sonoros podemos apreciar que el programa disponía de ciertas limitaciones puesto que no se podían generar varias

composiciones diferentes y aunarlas en el mismo documento. Para la elaboración de la música del menú inclusive tuve que crear dos documentos y unirlos después con Adobe Audition.

También podemos decir que el videojuego está falto de sonidos ambientales. Un sonido de agua corriendo para la fuente, sonidos cuando se realizan algunas acciones, etc hubiera creado una sensación de acción real.

8. Montaje de la escena en Unity

Es la primera vez que me topaba de lleno con un motor gráfico con el que desarrollar videojuegos, había escuchado hablar maravillas de Unity por la facilidad de acceso y de desarrollo y estaba a punto de ver si era cierto o falso. El motor estaba en pleno auge y no solo pequeños desarrolladores lo habían empleado para crear sus obras sino que videojuegos encontrados en grandes videoconsolas lo emplean para su funcionamiento.

8.1 Escenas

Unity funciona con escenas o *scenes,* pero para entrar de lleno a hablar de ellas antes tenemos que especificar que son. Cada una de las escenas definía cada una de las pantallas del videojuego en ella se dispondrían los elementos imprescindibles para adquirir funcionamiento. Por consiguiente nuestro proyecto final está compuesto por 4 escenas con extensión .unity:

- Menú principal (la pantalla que sale por defecto).
- Plaza (la primera pantalla donde aparece nuestro protagonista).
- Tienda del hechicero.
- Circo (ésta última no ha podido incluirse pero está a modo de broma).

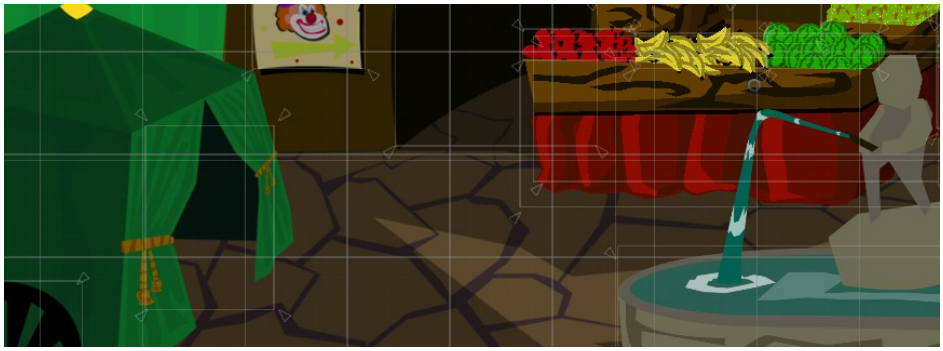
Se puede saltar de escena en escena siempre que éstas estén añadidas en el *Building Assets*, para añadirlas hay que ir a la opción *Build and settings* y *Add Open Scene.* De ésta manera añadiremos las escenas al mismo proyecto y podremos acceder de una a la otra a través de la programación por *scripts*.

8.2 Elementos imprescindibles para generar la escena

Para poder tener nuestras escenas funcionales debemos tener varios elementos imprescindibles. Por el tipo de videojuego que se trata tenemos que tener la ventana de trabajo definida como entorno 2D y

como mínimo una cámara, ésta sera la que nos muestre lo que ocurre en nuestro videojuego. En nuestro caso cada una de las pantallas contendrá los diferentes elementos pero estos estarán recogidos de un *canvas* donde se expondrán todos los objetos, el *canvas* funcionará como padre y de ella colgarán todos los elementos que estén contenidos dentro de la pantalla de juego. Ésto es muy importante porque de ello depende de que si ajustamos bien las *anchor*, todo se redimensione a la hora de hacer la pantalla más grande.

Vamos a explicar un poco que son las *anchor.* Las *anchor* no es la escala del *gameObject* en sí pero son las que modificarán su tamaño según la resolución del juego, por ese motivo es sumamente importante que estén bien posicionadas para que los diferentes elementos de juego actúen como queramos.



Podemos apreciar como las *anchor* están delimitando el tamaño de los *GameObject* de la escena.

Pese a que pueda parecer una tontería hay que poner cierta atención y mimo para que queden bien definidas. En las diferentes escenas hay varios elementos que no pueden regirse bien por las *anchor* porque no empiezan sus animaciones dentro del *canvas* y al ser éstos hijos de él no pueden tener las *anchor* fuera. Éstos son los elementos con los que tenemos que tener más cuidado y son los que nos han creado un efecto no deseado. Todos estos objetos se definen por una jerarquía que define su comportamiento y la cual tenemos que tener muy presente.

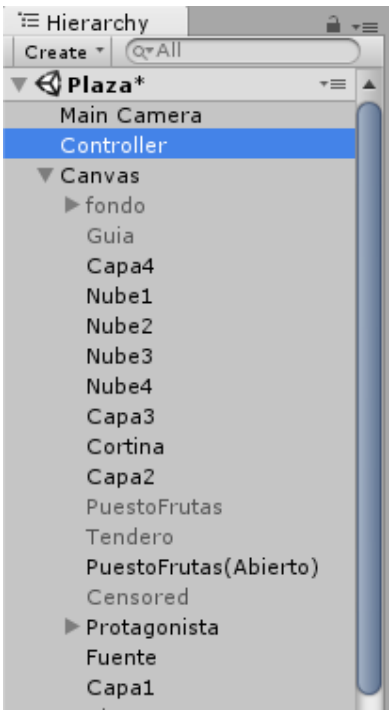
8.3 Jerarquía de los elementos de escena

Todos los elementos de la escena están regidos por una jerarquía, éstos se rigen por padres e hijos donde si algo afecta al padre inmediatamente también l hará con el hijo, ésto es algo muy importante que hay que tener presente a la hora de elaborar la escena y montarla puesto que para algunos comportamientos que deseemos posteriormente vía *script* tenemos que saber bien que causa puede tener en escena. Un ejemplo claro es que si un elemento padre está desactivado de la escena no se mostrará, pero inmediatamente repercutidos tampoco se mostraran los hijos pese a que no los

hayamos desactivado expresamente. Ésto no funciona de abajo a arriba, es decir, si tenemos desactivado un hijo, por contra el objeto padre si podría estar activado.

En el Trabajo ésta información es primordial porque se juega constantemente con la activación y desactivación de elementos los cuales tienen repercusión real en pantalla.

La jerarquía está definida en la lista de objetos que aparece a la izquierda de la escena. Ésta muestra en caída todos los objetos que están en ese momento en escena. En negrita remarcados los objetos activados, en gris los que no (éstos se pueden activar y desactivar pulsando el recuadro de objeto en el inspector de la derecha). Tenemos que entender la lista de la izquierda como si de capas de un editor de ilustración se tratara dónde los objetos más abajo de la lista se mostrarán más cercanos al usuario y las alejadas actuarán más alejadas. Ésto se da en los juegos en 2 dimensiones porque la información de las Z desaparece y se juega con la prioridad de capas.



Podemos observar como en la imagen de la izquierda, la capa *Puesto de Frutas* está desactivada entre otras, que *Canvas* es el padre de todas las demás capas que son utilizadas a modo de hijo y la importancia de la organización de las capas, dónde la *Capa4* se mostrará en el fondo, la *Capa3* más adelantada, la *Capa2* aún más y así hacia abajo. Todos los diseños de escenarios han sido creados con el fin de poder gestionar una jerarquía de capas con un sentido estético y funcional.

La jerarquía de las capas nos ha jugado malas pasadas, puesto que en el caso de la escena de la tienda del mago tenemos una capa que hace de primera (la que se vería más cercana al espectador) y nuestro personaje está entre ella y el fondo para que quede bien gráficamente, pero ésto hace que el texto de dialogo de nuestro personaje, al ser una capa hijo del mismo personaje (realizado así para que siga su movimiento gracias a las especificaciones de que el hijo se ve influido directamente por el padre comentadas anteriormente) se vea tapado por dicha primera capa, creando así un efecto no deseado.

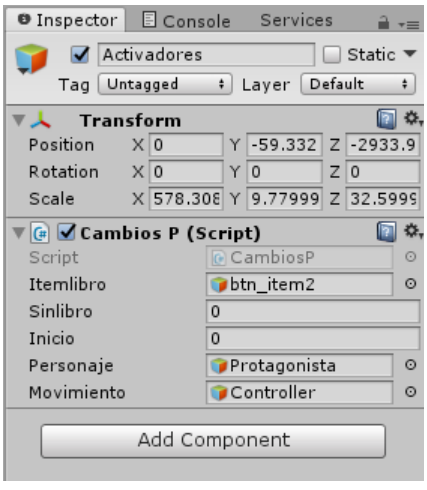
8.4 *GameObject*

Los *GameObjects* son todos aquellos objetos que completan la escena de nuestro videojuego. Éstos son los que aparecen en la lista de jerarquía nombrada en el punto anterior. Existen muchos tipos, de los cuales algunos ya tienen algunos complementos dedicados que Unity facilita para su fácil comprensión. *Sprites*, botones, imágenes; todo formarían *GameObjects*. Pero a parte de los que el mismo programario ya facilita con los complementos precisos para ser de un tipo u otro nosotros también podemos crear un objeto desde 0 con las características que deseemos.

La estructura de los objetos viene definida por los componentes que contenga, por ejemplo, al crear un objeto vacío éste estará exento de ningún componente el cual sabemos que estará en pantalla pero no estará dotado de ningún comportamiento. El crear un objeto desde 0 nos insta a investigar los diferentes componentes que contiene el programa e ir aplicándolo. En las escenas escogimos crear objetos vacíos y darles un *script* que se ejecutara tan solo inicializar la escena, de éste modo podríamos tener una lógica del juego que nos modificara la escena a antojo gracias a las variables que se quedan guardadas en *PlayerPrefs* (entraremos en detalle en el apartado de programación).

Nuestro trabajo está compuesto por objetos gráficos, como serían la interfaz y los fondos con su comportamiento único y objetos vacíos sin imágenes pero que son los encargados de ejecutar la lógica del juego gracias a que contienen *scripts*. Éstos están dispuestos por la pantalla posicionados por encima de los elementos de las capas del fondo que tienen interacción para que a la hora de pasar por encima de ellos nos cambie el contenido mostrado por pantalla textualmente.

Unity dispone de múltiples complementos que ayudan a que los diferentes objetos realicen los comportamientos deseados para poder añadir un comportamiento especifico al objeto lo añadiremos a través de la opción *Add component* de la ventana de inspector del *GameObject* allí podremos buscar en una lista el comportamiento deseado de nuestro objeto.



8.5 Estructuración propia del Trabajo

La estructura de objetos utilizada en el trabajo es siempre la misma para las diferentes escenas, primero de todo se define la cámara y el objeto *Canvas*, el que contendrá todos los elementos visuales. Dentro de éste *Canvas* lo primero que contiene son las capas que formarán el escenario y entre ellas está ubicado el personaje protagonista. Después de éstas capas están las que generan el menú de acciones, todos los botones y objetos de inventario más el elemento de menú vacío por detrás en el orden de capas para que todos los botones se muestren por arriba. Por ultimo al final de la estructura de capas tendríamos los *colliders*, estos deben estar por encima de todo, lo más cercano posible al jugador puesto que son los encargados de mostrarnos los diferentes objetos que tenemos en pantalla si éstos estuvieran más para arriba estarían tapados por las capas de fondo y no harían la función que les pertoca.

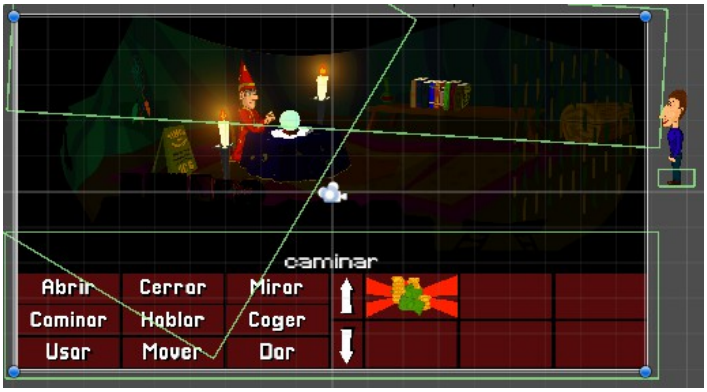
Hay otros elementos sueltos en la escena los cuales no influyen lo lejos o cerca que estén como los archivos de audio que están en constante funcionamiento. Éstos pueden modificarse de características en la pestaña de inspector para que se escuchen más o menos, o no se reproduzca automáticamente, en el caso del Trabajo se han dejado estáticos.

8.6 *Colliders*

Los *colliders* juegan un papel muy importante en el juego puesto que son los que vía *script* van a especificar por donde puede pasar el jugador y por dónde no. Al tratarse de un juego en 2 dimensiones éstos son rectangulos, los cuales están ubicados en los elementos gráficos donde el personaje protagonista no puede entrar, en el caso del escenario Plaza sería en las casas y paredes del fondo, mientras que en el escenario de Tienda del mago abarca una dimensión mucho mayor dejando el espacio de movimiento del personaje protagonista limitado al espacio del recibidor principal.

Para conseguir que éste efecto surja efecto a parte de la programación posterior también hemos tenido que dotar de un *collider* al personaje principal como un componente *rigidbody 2D* que le añade físicas al protagonista. El componente tuvo que ser modificado posteriormente puesto que al añadirlo desde el inspector aparecía por defecto con un valor de gravedad 1 lo que hacía que nuestro personaje cayera al vacío (hacia abajo), por ese motivo se cambió a valor 0.

Podemos apreciar en verde la de limitación de los *colliders* y con ello el espacio en el que nuestro personaje puede caminar



A través de la posterior programación donde se especifica que el protagonista no puede entrar en colisión con otros *colliders* también cambiamos el pivote del elemento protagonista a los pies para que cuando se pulsa a donde queremos caminar con el ratón se dirija allí con el eje centrado en ellos y no en el centro del cuerpo.

9. Programación

Nos encontramos delante de mi talón de Aquiles, la programación, pese a tener los conocimientos adquiridos durante lo largo del Grado, siempre ha sido la disciplina que más quebraderos de cabeza me ha hecho pasar y cuando se trata de poder realizar la programación para construir la lógico del videojuego todo parece indicar que no iba pues a mejorar sino a empeorar. Es la parte más costosa y la que más tiempo ha requerido.

La programación del videojuego y en definitiva toda la lógica está estructurada a partir de *scripts* en lenguaje C. Cada uno de ellos tiene un comportamiento dedicado expresamente para cada elemento y pueden ser clasificados en:

- *Script* controlador del movimiento del personaje.
- *Scripts* con la lógica de las interacciones con elementos.
- *Scripts* ejecutores de escenas.

9.1 *Script* controlador del movimiento del personaje

Éste es el encargado de hacer que el personaje se mueva por pantalla. Cabe decir que toda la programación y todas las estructuras y funciones predefinidas dentro de la propia programación de Unity han sido buscadas por Internet por desconocimiento, así que me he tenido que someter a una búsqueda intensiva para poder ir imaginando como poder salir airoso de las complicaciones que me iban surgiendo por el camino.

El *script* va ligado a un objeto vacío de la escena el cual es llamado *Controller*, y éste en su funcionamiento especifica el comportamiento que tendrá que realizar el protagonista al pulsar en pantalla con el ratón. Si el ratón pulsa un elemento de la escena donde no exista un *collider* el personaje caminará hacía allí con un vector direccional. Los vectores creados en el *script* son vectores 3 pero podría haberse creado vectores 2 dado que se trata de un juego de 2 dimensiones.

Una vez se ha pulsado el ratón éste identifica si se ha pulsado a izquierda o a derecha del personaje y muestra la animación que pertoca en ese momento, es decir si pulsamos a la izquierda del personaje se activará la animación de caminar a la izquierda y al contrario si es a la derecha.

Cuando el protagonista impacte con un *collider* éste se forzará para salir de él caminando hacia fuera. Si el personaje va caminando e impacta se parará cambiando así su animación.

Por las características del *script* no se consiguió hacer que el personaje apareciera inmóvil cuando no cambiara de posicionamiento a pesar de realizar muchas pruebas, ésto se especificará en apartados posteriores.

9.2 *Scripts* con la lógica de las interacciones con los elementos

Éstos se llevan el peso y son realmente los encargados de ejecutar todas las interacciones del juego. Se basan en las funciones de ratón predefinidas dentro del propio Unity. Así se especifica cual será su comportamiento cuando el ratón entra en contacto con el objeto que contiene el *script*, cuando sale con el ratón, cuando es pulsado o cuando se levanta la pulsación.

Éstos *script* están asociados tanto a los botones del menú como a todos los elementos interactivos por igual y son los encargados de modificar los apartados textuales de la pantalla al igual que activar y desactivar elementos a antojo.

Éstos *scripts* en los botones del menú de interacciones hacen que al entrar o pulsar en ellos nuestras imágenes de los botones cambien al estado *hover* o pulsación, al igual que volver al estado normal si se sale de ellos (los estados comentados en el apartado Arte). Pero no solo eso, al ser pulsados también especificamos que el texto que aparece en la parte superior del menú debe ser modificado con la acción pertinente. Para poder reutilizar el *script* con todos los botones se especifica que el texto a cambiar será por el nombre del objeto que contiene así que solo se tuvo que definir posteriormente el nombre del objeto en Unity.

Los *scripts* para los *collider* que están posicionados en los diferentes elementos interactivos de la pantalla tienen una lógica similar pero a la vez diferente. Éstos leen el contenido que hay escrito en el objeto texto de encima del menú y depende lo que haya actúan de una manera u otra. Por ejemplo si el texto escrito es “caminar”, texto escrito por defecto, sabe que al pulsar encima del elemento solo mostrará lo que es, pero si por contrario previamente se ha pulsado una de las acciones éste mostrará la acción más el nombre del objeto.

Si se pulsa con una acción pulsada previamente ésta modifica el texto que debe decir el protagonista y lo activa para que se muestre en pantalla, adicionalmente también cambia la animación del personaje para que en el transcurso de tiempo en el cual aparece el texto encima del personaje también hable.

Al salir el ratón de encima de los elementos si no ha sido pulsado se volverá a la acción previamente marcada para que podamos pulsar encima del objeto interesado.

Dadas las características de éste *script* y como se ha comentado al principio de la memoria cada uno de los elementos dará comentarios diferentes dada que es la gracia de éste tipo de videojuegos el tener que ir explorando las posibilidades que ofrece el escenario, por ese motivo existirá un *script* específico para cada uno de los elementos interactivos de pantalla y los objetos del inventario ya que no pueden ser reutilizables como en el caso de los botones del menú interactivo.

9.3 *Scripts* ejecutores de escenas

Éstos son los encargados de que las acciones que se han llevado a cabo en una escena cambien en la siguiente, se ejecutan en objetos vacíos y tan solo entrar en la escena leen el valor que tienen y depende cual sea muestran una u otra cosa.

Por ejemplo, si el personaje ha cogido el libro dentro de la tienda al salir de ella hacia la plaza seguirá el libro en el inventario al igual que si volvemos a entrar en la tienda en vez de cargar el escenario con el libro de nuevo y activado los *colliders* pertinentes aparecerá el hueco y no existirá la interacción anterior con el.

9.4 Elementos claves en la programación

Durante la investigación y creación de los *scripts* se han utilizado varias funciones que hacen que las cosas funcionen tal y como se esperan, es importante remarcar las que son más clave para entender en profundidad en que se basan. Éstas aparecen a a lo largo de los *scripts* y se repiten para conseguir que los elementos actúen de la manera adecuada.

.SetActive()

Ésta llamada que va precedida de un *GameObject* especifica que debe activarse o desactivarse según lo que ponga dentro de sus paréntesis donde true se activa y false lo desactiva, siempre hay que tirar hacia atrás y recordar de nuevo en la jerarquía de éstos para saber que si desactivamos el padre también lo harán los hijos.

Se ha utilizado en muchas ocasiones para desactivar el movimiento del personaje cuando no queremos que pueda moverse, al igual que para hacer aparecer o desaparecer los elementos de dialogo sobre los personajes.

.GetComponent<>();

Ésta llamada que se ejecuta dentro del *void Start* del *script* especifica el elemento que queremos que coja una variable, es decir, será el encargado de especificar cual de los componentes de un *GameObject* debe modificar. Se ha utilizado para coger el componente de animación y el textual de los *GameObject* para posteriormente variarlos.

StartCoroutine(());

Una de las funciones más importantes, después de varias pruebas con el *TimeDeltaTime* sin éxito ni diferentes otros métodos se llegó a utilizar ésta para hacer que los siguientes elementos se demoraran en ejecutar un cierto tiempo. Ésta función es muy utilizada para especificar cuanto tiempo debe estar el dialogo de las interacciones activados y en las conversaciones es el encargado de ir pasando de una a otro sin pisarse.

PlayerPrefs();

Es el encargado de pasar información de variables entre *scripts* pudiéndola recuperar de un archivo que se genera externamente. Al llamarlo pasamos las variables al archivo que generamos y al llamarlo de nuevo la volvemos a recuperar. Se utiliza en los archivos de carga de las escenas para poder implementar las modificaciones antes de que se carguen de este modo el videojuego puede mostrar una progresión real. Hay que tener especial cuidado puesto que si no se especifica que se elimine al salir del juego al iniciar una nueva partida seguirán los archivos como se dejaron en la última sesión, por lo que se puede entender la función como un guardado de partida.

Application.Quit();

Lo que a priori puede parecer obvio no lo es. Ésta función quita el juego, si no se implementa el juego solo se cerrará con la abreviatura de teclado Alt+f4.

10. Metas logradas y perdidas por el camino

El Trabajo empezó con una idea clara y unas pretensiones fijas, empezando con el ideal que se ha mantenido inamovible hasta ahora de que todo el contenido sería propio y que no se cogería material externo de ningún tipo para la producción; ni de gráficos, ni de menús, ni sonoro, ni de programación. Por ese motivo la cantidad de trabajo generado era muy alta y quizás había que ser más consecuente con la realidad porque por el camino se quedaron en el tintero algunos de los elementos que se tenían en mente aplicar en la versión final del videojuego.

10.1 Elementos logrados

Se ha conseguido generar unos escenarios únicos con interactividad propia dotados de animaciones que hacen que den el efecto de que están en movimiento y están dotados de vida.

Se han conseguido generar animaciones a los personajes principales y sobretodo al protagonista para hacer que actúe con el entorno de manera eficiente.

Se ha llegado a hacer una interacción única para cada uno de los objetos de pantalla como los objetos del inventario, haciendo que estos tengan un comentario único si son utilizados con el entorno.

Las relaciones causa-efecto que hacen que si utilicemos un objeto concreto de nuestro inventario pueda cambiar la escena y sentir una progresión en el juego.

Conseguir músicas realizadas expresamente para el juego y que no pertenezcan a ningún banco de sonidos.

10.2 Elementos perdidos

Parte de la historia principal, no se puede completar dado a que falta uno de los escenarios, el Circo, el cual no se ha llegado a implementar.

Animaciones en el escenario, en el caso de la Plaza en la capa más cercana al jugador aparecía la sombra de una rata que caminaba por encima de los escombros.

Pantalla de carga, que estaba creada y que se ejecutaba cuando el videojuego debía esperar para entrar en algún escenario.

El menú de opciones. La interfaz se llegó a crear, dando la posibilidad de silenciar la música y poder hacer que el texto apareciese en inglés.

La estética pixelada deseada ha sido desestimada por problemas técnicos y limitaciones del tiempo

Pese a que haya mucho contenido que no se ha podido implementar ya que se han realizado todos los gráficos éstos estarán añadidos en el directorio de *GitHub* para poder examinarlos si se quisiera. Tanto las imágenes generadas como los documentos de *Adobe Illustrator* donde se aprecian los escenarios y las animaciones generados por capas.

11. *Bugs* y carencias por falta de tiempo

Me encantaría decir que el videojuego final ha quedado pulido como un diamante pero no es así. El Trabajo se ha visto sometido a una presión de tiempo muy grande ya que no se ha realizado solo sino que se ha visto compartido con otras 3 asignaturas que han hecho que la dedicación no haya podido ser toda la deseada. El juego cumple con los objetivos de interactividad

que se preveían al inicio del Trabajo pero por el camino algunos de los elementos no actúan como desearíamos.

Hay un *script* por cada elemento interactivo, el desarrollo empezó generando el escenario de la Plaza, posteriormente se realizó el Menú principal para acabar haciendo la Tienda. Una vez terminada la tienda se modificaron los *scripts* que previamente tenían los *colliders* de la plaza ya que la escena de la tienda se generó encima de la plaza como guía ésto hizo que cada vez que se generaba uno nuevo se fuera modificando para una mejor interacción. Una vez listos los elementos de la tienda y viendo que la interacción de estos *scripts* era mucho mejor que los de la plaza, se rehízo por completo el escenario de la plaza dándole a éstos los nuevos scripts a la vez que mejorándolos a los de la tienda por éste motivo las interacciones de la plaza varían el tiempo que se queda el dialogo dependiendo de lo largo del texto a mostrar o cambia la animación de estático a hablando. Éstos fallos son fácilmente modificables puesto que debemos modificar los *scripts* del escenario tienda pero por falta de tiempo no ha podido solventarse.

Uno de los fallos que más me duelen es el no haber conseguido que el personaje deje de moverse cuando su movimiento es 0, la animación sigue en marcha como si estuviera caminando lo que queda mejor que si se quedara parado pero no consigue el efecto deseado.

Un poco ligado al punto anterior, me hubiera encantado conseguir implementar que el personaje se acercara el máximo posible al objeto a observar antes de ejecutar la acción pertinente.

No se ha podido conseguir que los diálogos del personaje queden por encima de todas las capas pero que siga siguiendo al protagonista. El efecto que queda, sobretodo en el escenario tienda, del texto tapado por las cortinas superiores y el sombreado enturbia saber que comentarios está realizando el protagonista.

Una de los factores que tenía en mente y que no se ha podido implementar es poder guardar gracias a la función PlayerPrefs por donde debe aparecer el protagonista al pasar entre pantallas puesto que siempre aparece por la derecha.

Por desgracia éstos fallos o mejoras se habrían podido implementar con más tiempo, aún así son fallos que no impiden que se pueda apreciar la interacción tal y como se quería lograr al iniciar el Trabajo sino que son mejoras para pulir los detalles. Con todo y ello se puede apreciar la interacción típica de los juegos de aventuras gráficas clásicas de los años 90.

12. Conclusiones

Sin duda ha sido toda una experiencia la realización del Trabajo Final de Grado, he tenido que aprender a marchas forzadas una serie de conocimientos

de programario desconocido al igual que la satisfacción de lograr que el videojuego consiguiera actuar como yo deseaba.

Creo que justo ésta rama escogida, videojuegos, cumple con la finalidad de todo el Grado Multimedia ya que debes poner en práctica toda la base que se ha adquirido a lo largo para poder buscar, implementar y mejorar a lo largo de la elaboración. Sin duda una experiencia gratificante, aunque a la vez de estresante, que sirve de culminación de éste camino de aprendizaje.

Creo sin duda que cualquier Trabajo Final de Grado te prepara para solventar y organizar proyectos, trabajar con unos tiempos precisos y saber generar una buena autogestión del tiempo invertido, el poder vislumbrar y ser capaz de saber asumir las fortalezas y las debilidades de uno mismo y poder mejorarlas y organizarlas para poder llegar a cumplir un objetivo propuesto. Por otro lado también creo que sirve de toque de atención. Un toque necesario y que te prepara para el mundo real, ésto es el día a día de la elaboración de proyectos, unos objetivos, unos tiempos marcados y una meta a alcanzar por lo cual creo que sirve de baipás perfecto entre el mundo académico y profesional.

Por desgracia no se ha podido llegar a los objetivos que teníamos planteados inicialmente como meta, por ese mismo motivo se ha tenido que rebajar dicha meta para poder llegar a presentar un producto que mostrara la interacción y lógica del juego y que mostrara una visión de a lo que podría derivar dicho proyecto, llegándose a poder convertir en un videojuego completo si se dedicaran años. No se han podido llegar a todos los objetivos iniciales dadas las limitaciones tanto humanas como en tiempos. Al ser la primera vez que nos topábamos tanto con el programario de Unity como enfrascándonos en la programación lógica de los videojuegos se ha “perdido” un tiempo indispensable en llegar a conseguir algunos de los procesos llegando en ocasiones a no encontrar una solución a ellos, como alguno de los fallos comentados con anterioridad.

A pesar de ello la satisfacción personal está más que lograda, puesto que si echamos hacia atrás la vista a la última entrega (PAC3) que se entregó hace dos semanas la mejora del producto es exponencialmente enorme, pasando de no tener interacción alguna a tener dos escenarios con propia interacción con cada elemento, todos con animaciones propias y con relaciones causa efecto con nuestros movimientos. Ésto es debido a que el Trabajo Final de Grado lo he realizado junto tres asignaturas más con sus PACs y Prácticas unido a un trabajo de jornada completa y una casa que llevar a cuestas. Los trabajos finales de el resto de asignaturas se terminaron aproximadamente con la entrega de la PAC3 del TFG con lo que a partir de estas dos semanas he dedicado cada hora de mis días a traer el resultado final. Cierto es que podría explotarse más pero también puedo decir que me siento orgulloso del resultado obtenido puesto que todo el contenido es propio lo que hace que lo valore aún mucho más.
Una vez terminado el Trabajo podemos sacar algo en claro, gracias a ésta investigación de las herramientas a marcha forzada tengo los conocimientos básicos para crear pequeños proyectos propios, algo que tengo claro que voy a seguir explotando y mejorando cara al futuro. Como objetivo próximo es acabar

de depurar por cuenta propia todo lo que se ha quedado por el camino y no se ha podido implementar, al igual que ya he hablado con un amigo para poder iniciar la producción de algún proyecto conjunto en lo que él aporta un valor añadido a mis debilidades y yo aporto todas mis fortalezas.

Sin duda una experiencia que llena de energía y cierra un ciclo único de esfuerzo, perseverancia y alegrías que a su vez abre los caminos hacia el futuro inmediato.

13. Glosario

- **Anchor** Derivada de la palabra inglesa “ancla” hace referencia a un componente asignado a los objetos de la escena en Unity que se sitúa en los límites del objeto para hacer que este pueda ser adaptativo al tamaño de resolución

- **Collider** Derivada de la palabra “colisionador” en inglés, hace referencia al componente que se le aplica a los objetos de la escena para que éste pueda tener colisión con otros objetos que también lo contengan

- **Engines kit** Hace referencia a los packs de desarrollo para desarrollar videojuegos. Unity, por ejemplo, es uno de ellos.

- **Key** Derivada de la palabra inglesa “llave” hace referencia a las marcas que añadimos en las animaciones para definir que en un punto existe un cambio.

- **MIDI** Es la abreviatura de las siglas inglesas para Musical Instrument Digital Interface. És un estándar tecnológico que describe un protocolo.

- **PNJ** Las siglas de Personaje No Jugable, es decir, todos aquellos que no son el protagonista al que controlamos.

- **Script** Derivada de la palabra inglesa “guión”. Son todos aquellos documentos de programación que se encargan de dar la lógica al juego y definen su comportamiento.

- **Sprites** Son todas aquellas imágenes que se modifican para que puedan contener una animación en plano.

14. Bibliografía

- (Libro)Smith, Rob. Chronicle Books. 26 de Noviembre de 2008. Rogue Leaders: The story of LucasArts. ISBN 0-8118-6184-8.

- (Web)Wikipedia, Monkey Island:
  - https://es.wikipedia.org/wiki/Monkey_Island

- (Web)Wikipedia, Simon the sorcerer:
  - https://es.wikipedia.org/wiki/Simon_the_Sorcerer

- (Web)Wikipedia, Leisure Suit Larry:
  - https://es.wikipedia.org/wiki/Leisure_Suit_Larry_6:_Shape_Up_or_Slip_Out!

- (Web)VidaExtra, Si quieres hacer tus propios juegos, estos son los mejores motores que vas a encontrar, 28 de Diciembre de 2017:
  - https://www.vidaextra.com/listas/si-quieres-hacer-tus-propios-juegos-estos-son-los-mejores-motores-que-vas-a-encontrar

- (Web)VidaExtra, Cuatro motores gráficos para perder el miedo y lanzarse al desarrollo de videojuegos, 11 de Octubre de 2015:
  - https://www.vidaextra.com/listas/4-motores-graficos-para-perder-el-miedo-y-lanzarse-al-desarrollo-de-videojuegos

- (Web)Unity 3D, lugar web oficial:
  - https://unity3d.com/

- (Web)Unity Docs, Documentación explicativa de las funciones incluidas dentro de Unity:
  - https://docs.unity3d.com

- (Web)Shiva Engine, lugar web oficial:
  - http://www.shiva-engine.com/

- (Web)Unreal Engine, lugar web oficial:
  - https://www.unrealengine.com/en-US/what-is-unreal-engine-4

- (Web)Elaboración diagrama de Gantt, Ganttproject:
  - https://www.ganttproject.biz/