UNIVERSIDAD DE ALCALÁ

Escuela Politécnica Superior

Grado en Sistemas de Información

Trabajo Fin de Grado

"Desarrollo de videojuegos:
Accesible vs No Accesible"

Rubén Cumbreño Juan

Año 2016/17

Agradecimientos

Gracias a todos aquellos involucrados directa o indirectamente, que me han dado ideas, que han probado mi aplicación en distintos dispositivos a todas horas y que me han facilitado las cosas y me han dado apoyo para poder finalizar este proyecto.

Gracias gente de **Freepick** por las imágenes CC prestadas.

Gracias gente de **Freesound** por los sonidos CC prestados.

Gracias gente de **OpengameArt** por los Sprites CC prestados.

Gracias **Irene**. Por ser mi Tester oficial en cada fase, mi traductora y mi fuente de ideas.

Gracias **Juan**. Por sus críticas constructivas e ideas para mejorar el juego, así como su apoyo y su TFM para recopilar información.

Gracias **José María**. Por introducirme en este mundo y darme las nociones fundamentales para empezar.

UNIVERSIDAD DE ALCALÁ

Escuela Politécnica Superior

Grado en Sistemas de Información

Trabajo de Fin de Grado

"Desarrollo de videojuegos:
Accesible vs No Accesible"

Autor: Rubén Cumbreño Juan

Tutor: José María Gutiérrez Martínez

Cotutor: Juan Aguado Delgado

Tribunal:

Presidente: .....

Vocal 1º: ..... Vocal 2º: .....

Calificación: .....

Fecha: ..... de ..... de .....

Resumen

Trabajo de fin de grado realizado para comprobar el impacto tiempo-económico (esfuerzo) resultante de la creación de un videojuego y su adaptación accesible a diversas discapacidades. Se pretende analizar si existe una posible rentabilidad en transformar un juego para cubrir ciertas necesidades específicas, a saber, auditivas, visuales, motoras y lingüísticas. Tras su finalización se espera poseer los conocimientos necesarios para desarrollar un videojuego 2D en Unity 3D y datos concretos sobre los esfuerzos necesarios para hacerlo accesible.

Palabras clave

Las siguientes palabras son clave tanto en el documento como en el proyecto desarrollado. La búsqueda de ellas ayuda a aclarar, definir y completar toda la información referente a este trabajo:

Palabras Clave: accesibilidad, discapacidad, desarrollo, coste, Unity.

Abstract

End of Degree Project developed in order to check the time-economical impact (effort) resultant from creating a videogame and adapting it to diverse impairments. In this way, the aim is to analyze if there is a possible profitability on transforming a game to cover some specific necessities, named auditory, visual, motor and linguistic. By the time it has been finished, it is expected to provide the readers with the knowledge necessary to create a 2D videogame using Unity 3D, as well as concrete data about the effort needed to make it accessible.

Key words

The following words are key both in the document and in the project developed. The search for them helps to clarify, define and complete all information regarding this work:

Key Words: accessibility, disability, development, cost, Unity.

Resumen extendido

Se va a realizar un estudio y desarrollo de todas las fases de organización, elaboración y conclusión de un videojuego creado para dispositivos móviles.

Se pretende estudiar cuales son las mejores herramientas que nos faciliten la labor de desarrollo en el mercado a día de hoy, aprender el manejo básico de la herramienta seleccionada y adquirir la suficiente fluidez como para poder desarrollar un videojuego.

Una vez se haya realizado dicho trabajo, se modificará la mecánica del videojuego para añadirle distintas accesibilidades. Tras realizar los estudios pertinentes, se realizarán cambios para cubrir necesidades visuales, auditivas, motoras y lingüísticas.

Se estudiarán los casos con mayor dificultad de ejecución, así como las soluciones que se han llevado a cabo para su resolución. Aprenderemos los elementos del videojuego, el diseño de cada pantalla y realizaremos la comparativa con las versiones accesibles para comprobar los cambios realizados y las funcionalidades añadidas o eliminadas.

Una vez desarrollado el juego se desplegará en el repositorio correspondiente (Play Store o App Store), siguiendo el pliego de condiciones correspondiente y se dará acceso al público para obtener un mayor volumen de informes (favorables o desfavorables) con objeto de hacer evolucionar nuestra aplicación y someterla a la crítica social.

Se hablará sobre la industria del videojuego, trataremos los distintos perfiles técnicos que conforman un equipo de desarrolladores, la diferencia salarial entre ellos, cuales están mejor valorados y hablaremos sobre si hay o no discriminación de género.

Se llevará a cabo un análisis económico de los costes de todo el desarrollo de la aplicación y de los costes de añadir dichas accesibilidades, veremos formas de hacer dinero con una app y se realizará una comparativa para decidir si es o no rentable el esfuerzo realizado.

Para terminar, se definirán posibles futuras mejoras que se podrían añadir al videojuego y que lo harían más atractivo de cara al público.

Índice

1. Introducción
2. Objetivo
3. Estado del arte
   - 3.1 Descripción de las herramientas utilizadas
   - 3.2 Accesibilidad
4. Trabajo realizado
   - 4.1 Diseño y desarrollo de la aplicación
     - 4.1.1 Planteamiento del trabajo y objetivos a conseguir
     - 4.1.2 Características y arquitectura de la aplicación
     - 4.1.3 Principales retos tecnológicos
     - 4.1.4 Descripción de los elementos tecnológicos
       - 4.1.4.1 Interfaz
       - 4.1.4.2 Niveles
       - 4.1.4.3 Tipos de enemigos y su programación
       - 4.1.4.4 Armas
       - 4.1.4.5 Banda sonora y efectos sonoros
       - 4.1.4.6 Animaciones
       - 4.1.4.7 Power Ups
   - 4.2 Diseño y desarrollo de la accesibilidad
     - 4.2.1 Accesibilidad
     - 4.2.2 Principales retos tecnológicos
   - 4.3 Publicación del juego en Play Store
   - 4.4 Análisis de mercado
   - 4.5 Pliego de condiciones
   - 4.6 Análisis económico
     - 4.6.1 Coste económico realización de la aplicación
     - 4.6.2 Rentabilización
     - 4.6.3 Coste de la accesibilidad
   - 4.7 Manual de usuario
5. Resumen
6. Conclusión
7. Futuras mejoras
8. Bibliografía

1 Introducción

En un mercado en alza como es el de los videojuegos, que cada vez cuenta con más adeptos y cuyas ayudas, manuales e información desbordan por internet, sigue existiendo una clara discriminación por cuestiones de género o discapacidad. Este trabajo va a centrar su atención en esa parte de personas con características diferentes y se va a investigar por qué no tienen las mismas oportunidades que otros usuarios.

Según cifras del IMSERSO, el número de gente con alguna discapacidad es de 2,5 millones (en España) frente a los 46,56 millones de habitantes que hay (un 5,37% de la población). Debido a ser el grupo minoritario a lo mejor las empresas no ven factible partir del 100% de la población y prefieren centrarse en el grupo mayor. En este proyecto, se va a analizar si realmente es o no rentable la modificación de una aplicación para acceder a este conjunto de personas y se aclararán (en caso de haberlos) los límites de las funcionalidades a las que sería necesario renunciar al adaptar una app.

Se pretende, por tanto, establecer el esfuerzo necesario para llevar a cabo diferentes accesibilidades en una aplicación, y concluir si es o no rentable para las empresas. Se estudiará el porqué de esa situación y las posibles soluciones que se pueden llevar a cabo.

Se ha elegido dicho proyecto debido a la necesidad de buscar la igualdad entre todos los individuos de nuestra sociedad, así como para entender la posición de las empresas ante dicha situación y aprender métodos y formas de poder poner solución a este problema y poder verificar si es posible buscar unos métodos de adaptabilidad que sean rentables para las empresas.

Además se ha elegido dicho proyecto dado que otorga la posibilidad de introducirse en el mundo del desarrollo de videojuegos y da la oportunidad de aprender las diferentes herramientas, el estado del mercado y las distintas opciones profesionales que hay en dicho sector.

2 Objetivo

El objetivo principal del trabajo a desarrollar es la realización de un videojuego y una versión accesible del mismo (apto para ser usado por personas con diversidad funcional) y mediante una documentación exhaustiva del proceso concluir el esfuerzo y el tiempo necesarios que habría que invertir en la transición entre ambas versiones y descubrir así los mecanismos necesarios para completar la conversión.

Una vez se dé por concluido este proyecto, se espera alcanzar los siguientes objetivos secundarios:

- Aprendizaje del manejo de la herramienta seleccionada para el desarrollo.
- Aprendizaje de las distintas fases de creación, elaboración y despliegue de un videojuego para el entorno escogido.
- Obtener información sobre las distintas discapacidades existentes y posibles soluciones para realizar la adaptación de una aplicación.
- Aprendizaje de conocimientos más técnicos que permitan adaptar una aplicación a distintas discapacidades.
- Verificar si es necesario prescindir de funcionalidades básicas en una aplicación para su adaptación.
- Verificar si es posible hacer rentables la adaptación de las aplicaciones.

3 Estado del arte

En este apartado, se van a describir las diferentes herramientas investigadas para la elaboración del proyecto, así como la herramienta elegida. También se investigará y obtendrá información sobre las diferentes accesibilidades que se quieren tratar, para pensar formas de adaptación al juego.

3.1 Descripción de las herramientas utilizadas

Para la realización de dicho trabajo, se ha pensado en dos herramientas, "Android Studio" y "Unity 3D".

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android y se basa en IntelliJ IDEA. Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece aún más funciones que aumentan tu productividad durante la compilación de apps para Android, como las siguientes:

- Un sistema de compilación basado en Gradle flexible.
- Un emulador rápido con varias funciones.
- Un entorno unificado en el que puedes realizar desarrollos para todos los dispositivos Android.
- Instant Run para aplicar cambios mientras tu app se ejecuta sin la necesidad de compilar un nuevo APK.
- Integración de plantillas de código y GitHub para ayudarte a compilar funciones comunes de las apps e importar ejemplos de código.
- Gran cantidad de herramientas y frameworks de prueba.
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versión, etc.
- Compatibilidad con C++ y NDK.

Cada proyecto en Android Studio contiene uno o más módulos con archivos de código fuente y archivos de recursos. De manera predeterminada, Android Studio muestra los archivos de tu proyecto en la vista de proyectos de Android. Esta vista se organiza en módulos para proporcionar un rápido acceso a los archivos de origen clave de tu proyecto.

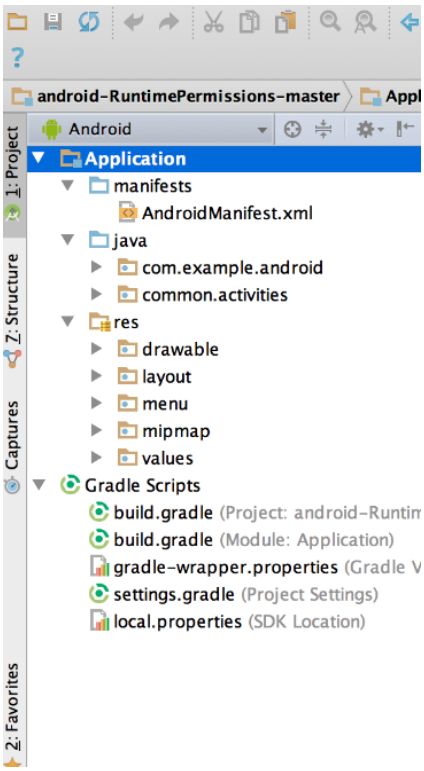


Figura 1: *Estructura proyecto Android Studio*

Todos los archivos de compilación son visibles en el nivel superior de Secuencias de comando de Gradle.

La estructura del proyecto para Android en el disco difiere de esta representación plana. Para ver la estructura de archivos real del proyecto, selecciona Project en la lista desplegable Project.

También puedes personalizar la vista de los archivos del proyecto para concentrarte en aspectos específicos del desarrollo de tu app. Por ejemplo, al seleccionar la vista Problems de tu proyecto, aparecerán enlaces a los archivos de origen que contengan errores conocidos de codificación y sintaxis, como una etiqueta de cierre faltante para un elemento XML en un archivo de diseño.

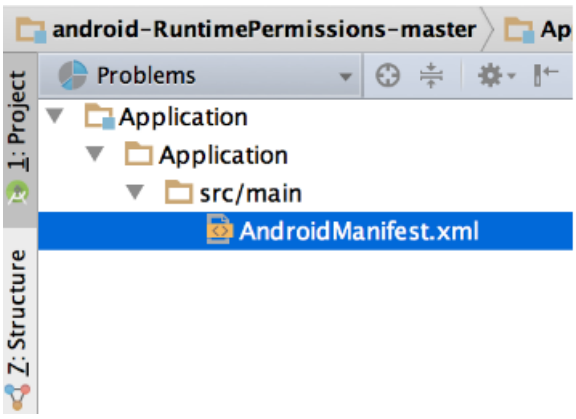


Figura 2: *Archivo de configuración básico de nuestra App*

Para la integración de las animaciones y funcionalidades más complejas se pensaba usar una librería externa como apoyo, LibGDX.

Libgdx es un framework multi-plataforma para el desarrollo de juegos. Posee un poderoso conjunto de APIs que le ayudan con las tareas de desarrollo de juegos común, como la prestación de los sprites y el texto, la creación de interfaces de usuario, la reproducción de efectos de sonido y corrientes musicales, álgebra lineal y cálculo de trigonometría, análisis sintáctico JSON y XML, y así sucesivamente.

Libgdx pretende ser un framework en lugar de un motor, reconociendo que no hay una talla única para toda la solución. En lugar de ello tiene abstracciones potentes que permiten elegir la forma en que desea escribir su juego o aplicación.

Pero la unión de estas dos plataformas (aunque ambas se basan en IntelliJ), resultó ser complicado y difícil de llevar a cabo, por lo que se optó por explorar otras vías.

Como alternativa se ha usado "Unity 3D". Es un motor gráfico utilizado para crear videojuegos y que ha sido desarrollado por Unity Technologies. Esta herramienta funciona tanto en Windows, Linux y OS X y permite la realización de estos juegos para dichos sistemas además de para para plataformas de sobremesa como: Playstation3, Playstation4, XBOX 360, XBOX ONE, WII, Android, WindowsPhone e incluso y más novedoso, para Smart TV y en el campo de la realidad virtual.

Unity cuenta con cuatro módulos que puedes adquirir para su desempeño: Personal, Plus, Pro, Empresa. Las diferencias entre ellos residen en los ingresos que produzcan los juegos o de las características que se deseen, aunque todas las versiones (incluida la gratuita) incluyen la capacidad de crear tu videojuego con el motor y la exportación a cualquier plataforma.

Una de las limitaciones más importantes es el límite de ingresos. Para la versión personal, el límite de ingresos se sitúa en los 100.000€, cifra a partir de la cual habrá que adquirir la siguiente versión, la Plus, que permite llegar a los 200.000€. Ninguna de las otras tarifas impone un límite de ingresos. Pueden parecernos cifras extremadamente altas para usuarios que son iniciados, pero no lo son tanto para empresas del sector especializadas en esto. Por ello, en este proyecto adquirimos la versión **Personal**.

Figura 3: *Módulos de Unity 3D* (Unity Technologies)

En cuanto a las características del editor, en Unity todos los objetos que interactúan en el juego son llamados **Game Objects** y se deben situar en alguna parte de la escena.

La **escena** es el contenedor de todos los objetos del nivel, menú o cualquier utilidad que se le quiera dar, son las distintas pantallas que se ven. Por norma general, al cambiar de escena todos

los objetos de la anterior son destruidos a no ser que se especifique lo contrario mediante un **script**. Una vez creado un Game Object con las propiedades deseadas, este se puede guardar en forma de **prefab**, que es como una plantilla para crear otro objeto idéntico en cualquier momento y lugar.

El editor de Unity se compone de varias ventanas que se pueden agrupar en forma de pestañas por comodidad. Entre estas ventanas destacan:

La **cámara de escena** donde se pueden ver todos los Game Objects que pertenecen a la escena y modificar sus atributos y posiciones.

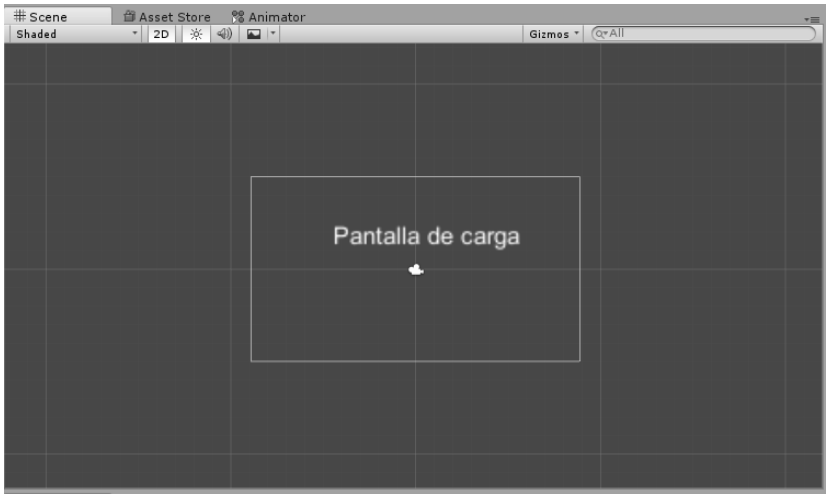


Figura 4: *Pestaña de escena*

La **cámara del juego**, es donde se reproduce lo que se vea en la cámara principal del editor y que será lo que vea el usuario final. Se puede interactuar con ella como jugador para realizar todas las pruebas pertinentes de que el juego funciona.

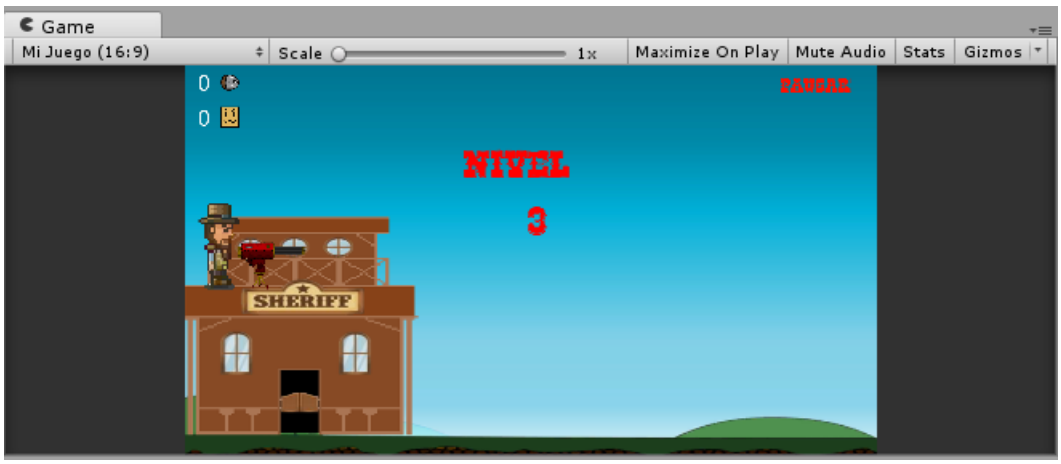


Figura 5: *Pestaña del Juego*

La ventana de **animación**, es la que nos permite crear y modificar las animaciones que luego se instanciarán en la escena.

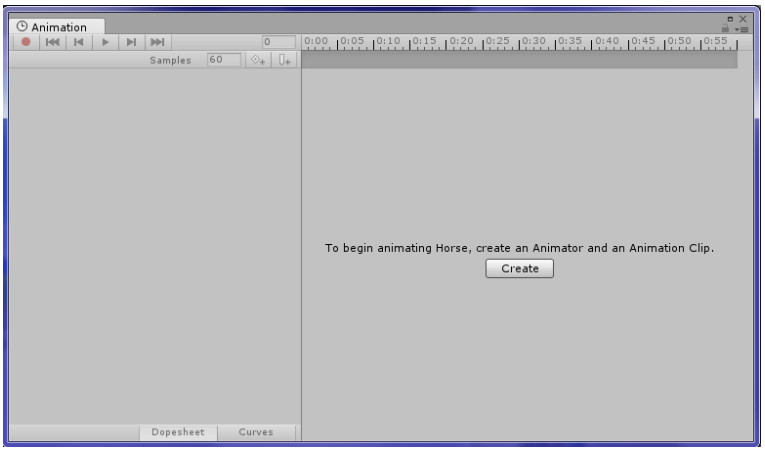


Figura 6: *Pestaña Animación*

La ventana de **jerarquía**, muestra los Game Objects pertenecientes a la escena que se está editando actualmente, ofreciendo un acceso fácil a estos y a sus propiedades.

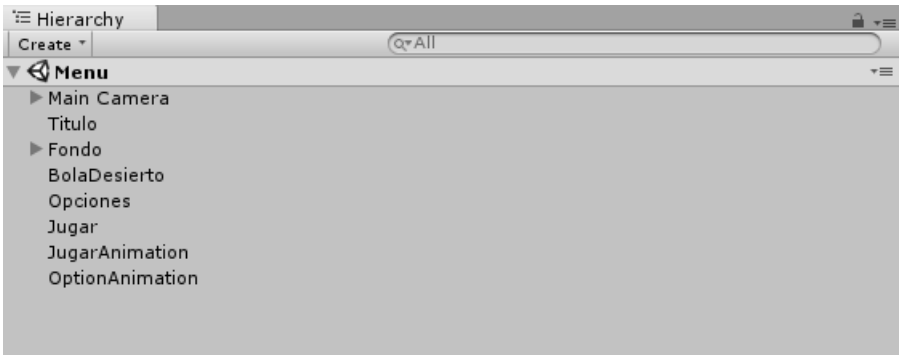


Figura 7: *Pestaña Jerarquía*

La ventana del **inspector**, es una de las más importantes, muestra las características y propiedades del objeto seleccionado, como la posición, las propiedades físicas, los scripts, etc.

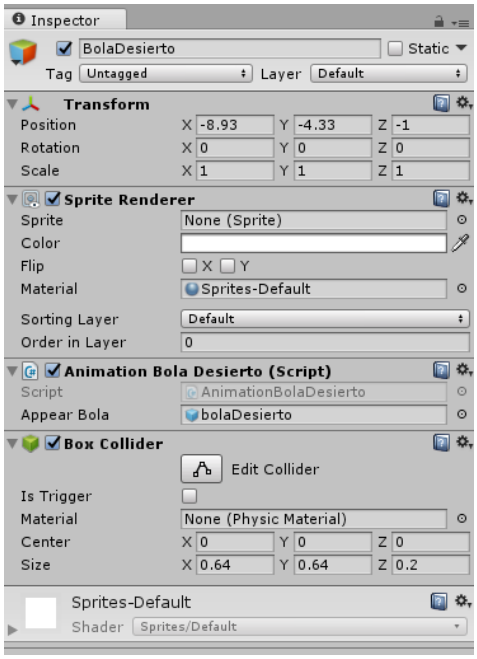


Figura 8: *Pestaña del Inspector*

La pestaña del **proyecto**, es aquella que muestra la disposición jerárquica de las carpetas y archivos que lo componen pudiendo acceder fácilmente a estos y modificarlos.

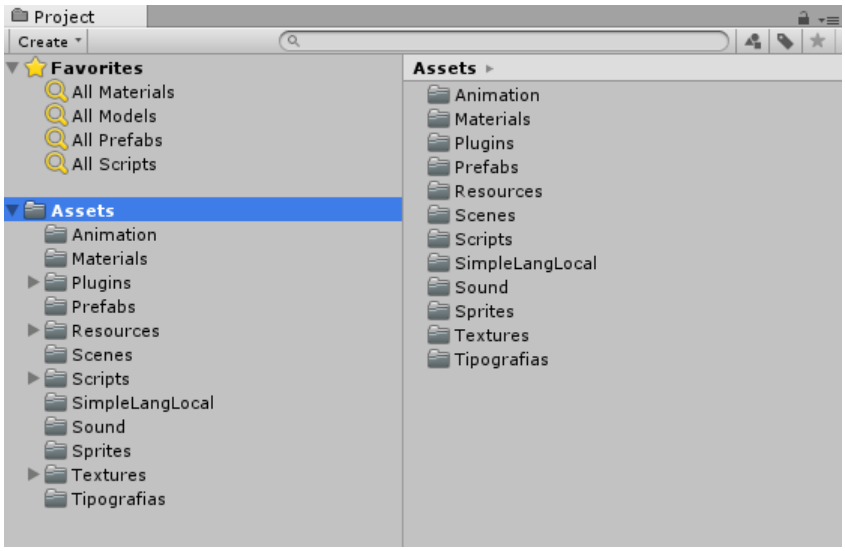


Figura 9: *Pestaña del Proyecto*

Por último la **consola**, es otra de las pestañas más interesantes debido a que es aquella que muestra los posibles errores y mensajes de debug que se activan por alguna condición en los scripts.

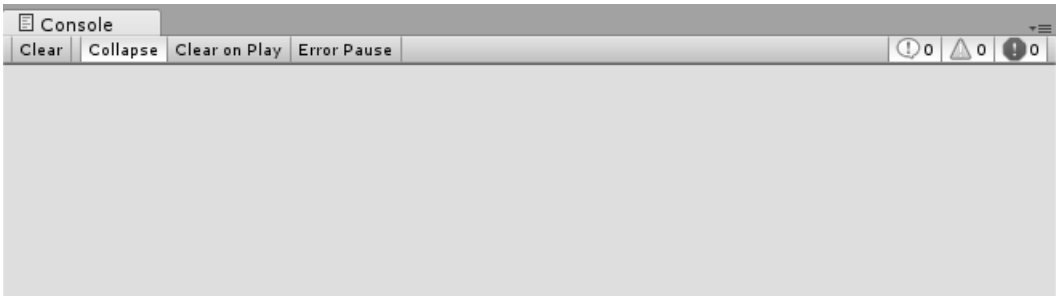


Figura 10: *Pestaña de la consola*

3.2 Accesibilidad

Según la ley de igualdad de oportunidades, no discriminación y accesibilidad universal de las personas con discapacidad (Ley orgánica 51/2003, del 2 de diciembre de 2003), se define la accesibilidad como la "Condición que deben cumplir los entornos, productos y servicios para que sean comprensibles, utilizables y practicables por todas las personas". (Aguado Delgado, 2014).

Las investigaciones destinadas a mejorar el entorno de las personas con diversidad funcional se ha centrado, principalmente, en la accesibilidad web y en tecnologías de apoyo como audífonos y/o dispositivos de ayuda visual. En los videojuegos no se ha producido un empeño tan grande o un impacto tan amplio, de ahí la naturaleza y necesidad de este proyecto.

Para centrar el foco en el que se basa este trabajo, se describen a continuación las discapacidades a tratar, basadas en el estudio previo realizado por (Aguado Delgado, 2014):

- **Discapacidad Visual**: es la consecuencia de un cierto grado de pérdida de visión. También se considera una deficiencia visual a la falta de sensibilidad a los colores.

    - **Ceguera:** se define como la pérdida de visión que no puede ser corregida con el uso de lentes. Los jugadores que son totalmente ciegos no pueden utilizar correctamente los videojuegos que se basan en señales visuales (la gran mayoría), teniendo para ello que depender de sonidos o Hardwares especiales. En este proyecto, se va a implementar una función de vibración calculando la velocidad de caída del objeto y la distancia y velocidad de la bala, de tal forma que el jugador solo tenga que tener los reflejos necesarios para, una vez vibre el dispositivo, pulsar la pantalla y destruir el objeto.

    - **Falta de sensibilidad a los colores:** se entiende como la incapacidad para detectar ciertos colores, lo que se traduce en una falta de respuesta a los mismos. Este proyecto va a trabajar con una "falta de sensibilidad severa", es decir, con el extremo máximo de deficiencia, pudiendo el jugador, únicamente, distinguir una tonalidad de grises. Para ello se va a instaurar una función que permitirá adaptar el videojuego de forma que se aprecie en la tonalidad de grises adecuados para su perfecto funcionamiento.

- **Discapacidad auditiva:** se refiere a una pérdida total o parcial de la habilidad para escuchar por uno o ambos oídos, con un nivel de deficiencia que puede ir de leve a completo.

    - **Sordera o pérdida de audición:** esto supone la pérdida total o parcial de la habilidad para escuchar, lo que implica una incapacidad para entender el habla o reconocer sonidos ambientales. En este proyecto se pretende habilitar una función que permita, o bien aumentar el volumen de los altavoces para personas con pérdida de audición media (los sonidos más silenciosos captados rondan entre los 40 y 75 dB) o bien implementar una función de vibración de tal manera que consiga añadir esos matices que el juego pierde con dicha discapacidad (como por ejemplo, vibrar el móvil al destruir un enemigo, o disparar).

- **Discapacidad motora**: también conocida como discapacidad física, alberga aquellas minusvalías que, por diferentes motivos, suponen una reducción de la movilidad o producen una pérdida o limitación de la función de control muscular o del movimiento.

    - **Parálisis parcial**: según la severidad del caso, se trata de una discapacidad que afecta a los nervios que controlan los músculos de tal forma que su habilidad se puede ver reducida. Para ello, el juego contará con facilidades táctiles adaptadas de tal manera que cualquier jugador con movilidad reducida pueda desempeñar su práctica sin problemas.

- **Accesibilidad lingüística:** por último, no se puede hablar de accesibilidad sin tratar la capacidad del lenguaje y del entendimiento. Se entiende pues, como la capacidad del jugador, independientemente de lengua y país, de desarrollar con total normalidad el desempeño de la actividad. Por ello, se tomarán las medidas necesarias para internacionalizar el lenguaje del videojuego, por el momento, a dos lenguas diferentes, dando un videojuego multilenguaje como resultado.

4 Trabajo realizado

A continuación se va a describir todo el desarrollo del trabajo, desde el diseño de la aplicación (desarrollo, elementos, etc.) y su accesibilidad (dificultades, retos tecnológicos, etc.), hasta el estudio de mercado y análisis económico.

4.1 Diseño y desarrollo de la aplicación

Una vez recopilada toda la información relevante e ideado que tipo de juego queremos, empieza el desarrollo del mismo. En esta fase se van a definir las metodologías de desarrollo, así como las características de nuestra aplicación, dificultades halladas y los elementos del juego.

4.1.1 Planteamiento del trabajo y objetivos a conseguir

Para la realización del desarrollo del juego se establecerá una metodología de **observación activa**. Esto es realizar un estudio cuidadosamente documentado para resolver un problema y/o cambiar una situación (Herbert, 1990); llevando a cabo una revisión bibliográfica profunda sobre la situación actual de los sistemas.

Para el desarrollo del prototipo también se seguirá una **adaptación** de la metodología **IDEAL**, la cual posee un ciclo de vida basado en prototipaje rápido, consistente en conseguir, desde etapas muy iniciales del desarrollo, unos prototipos que den idea de cómo debe funcionar el sistema definitivo. Se debe conseguir un proceso de mejora gradual, construyendo prototipos que se someten a pruebas de uso, las cuales mostrarán nuevas mejoras a incorporar a un nuevo prototipo.

Con esta metodología se irán desarrollando APKs progresivamente que se irán probando en distintos dispositivos por distintos usuarios con el fin de someterlo a pruebas e identificar las constantes mejoras a la vez que se van añadiendo.

El esquema a seguir a la hora de la realización ha de ser algo como:

Analizar la estructura y tema del videojuego, para una adecuada funcionalidad se ha de saber cómo se quiere la estructura del juego y que temática va a seguir, así se podrá hacer un estudio previo de las discapacidades a tratar y tener una visión más periférica de lo que queremos.

Análisis y creación de las escenas necesarias, una vez se sabe cómo se va a estructurar, lo siguiente es empezar con la creación de las distintas pantallas que va a contener nuestro videojuego.

Selección de imágenes e iconos, una de las partes más importantes a la hora de crear escenas es elegir correctamente los estilos que se van a añadir. No se pueden incluir imágenes con copyright si en un futuro se pretende desplegar la aplicación en Play Store.

Desarrollo del videojuego, en este momento ya estamos preparados para empezar a crear nuestros scripts que darán la funcionalidad a nuestras escenas. Es la parte más trabajosa y larga.

Pruebas, aunque las pruebas se realizan durante todo el proceso de forma constante, una vez se dé por finalizado el desarrollo, se instaura una última fase de prueba que bombardeen a la aplicación y la pongan a prueba.

Accesibilidad, con el juego inicial terminado solo queda hacerlo accesible. Estudiar las distintas discapacidades y decidir cuales implementar, desarrollar el código necesario para ello, comprobar nuevos fallos y estudiar su viabilidad.

Para terminar, una última batería de **pruebas** para contrastar que efectivamente todo está en orden.

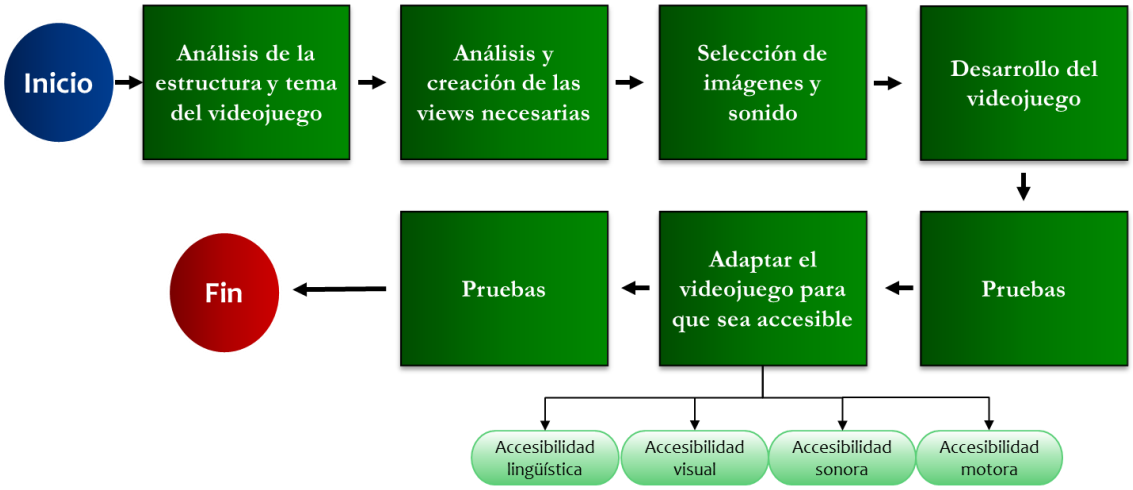


Figura 11: *Planteamiento del desarrollo de la aplicación*

En cuanto a los objetivos propuestos a la hora de desarrollar el juego:

Iniciarnos en un nuevo lenguaje, se ha escogido como lenguaje C#, el cual supone un reto y a la vez una oportunidad de aprendizaje.

Experimentar con un motor como Unity 3D, al emplear Unity como herramienta de desarrollo 2D, se va a profundizar en el aprendizaje de estos elementos y se conseguirá un mejor manejo de esta herramienta.

Accesibilidad, se analizará la dificultad de modificar un videojuego para hacerlo accesible a diversas discapacidades y se ampliará el concepto de accesibilidad y de la rentabilidad que esta puede aportar a un videojuego.

4.1.2 Características y arquitectura de la aplicación

El juego viene por defecto con una **pantalla de animación** de Unity debido a que se usa la versión Free. Después de ella se encuentra una **pantalla de carga** utilizada para precargar todas nuestras variables globales de la aplicación, así como aquellos datos serializados. Una vez realizada la carga se accede a un **menú principal**, desde el cual se puede acceder a una **pantalla de niveles** disponibles que da acceso a los distintos **niveles**. Desde allí se puede volver al menú principal o salir del juego.

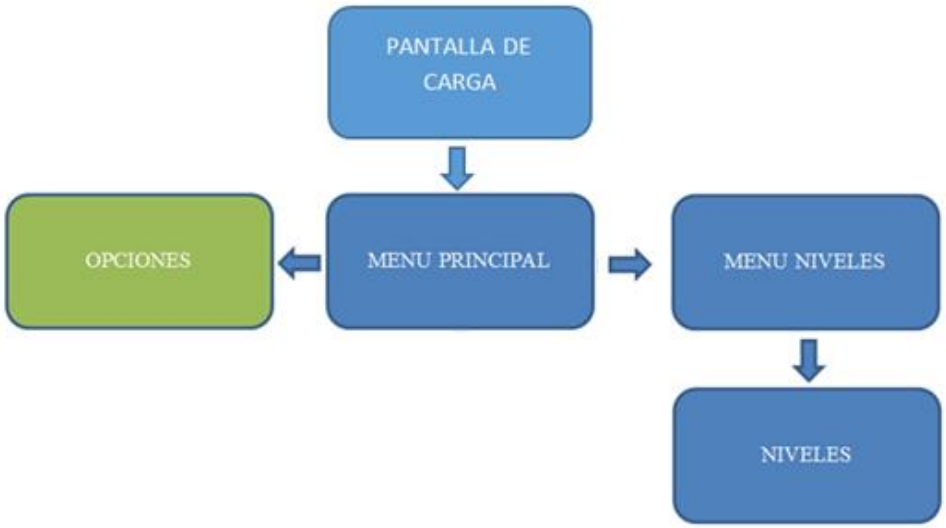


Figura 12: *Arquitectura de la aplicación*

**__Verde:__ se instanciará en el desarrollo accesible*

Todas las escenas tienen un Game Object que actúa como cámara con un script que me aporta el toque responsive sea el tamaño de pantalla que sea. Además cada pantalla posee un script que controla todos los textos de la misma, de tal forma que no se pueden encontrar textos dentro del inspector de la misma.

4.1.3 Principales retos tecnológicos

Se han tenido diversas dificultades en el desarrollo del proyecto, las más resaltables han sido:

Creación de animaciones. Las animaciones son una de las partes más importantes a la hora de crear un videojuego, dado que es por la vista por donde se obtiene la mayor parte de la información y el primer sentido que juzga la calidad de un juego. En este trabajo se ha hecho uso de los conocidos "**Sprites**". Los Sprites son mapas de bits 2D que se dibujan directamente en la pantalla del ordenador por hardware gráfico especializado. Con la llegada de los videojuegos en 3D, el uso de Sprites se ha comenzado a dejar de lado, al ocupar personajes poligonales. Unity facilita de cierta manera la instanciación de estos Sprites mediante la ventana Animación mencionada anteriormente, pero su creación, búsqueda y modificación resulta ser complicada para gente sin los conocimientos y el estudio previo adecuado.

Figura 13: *Sprite Bola del Oeste (Barrilla)*

Creación de Script responsive. Uno de los mayores problemas que tiene la creación de un videojuego es que este se adapte de igual manera a todos los diferentes dispositivos que existen en el mercado, a todas las resoluciones de pantalla, sistemas operativos y tamaños de pantalla. Para ello se ha creado un script que se encarga de adaptar a cualquier terminal el tamaño que queremos que tenga nuestro videojuego. Funciona de la siguiente forma: se piensa como se quiere el videojuego (en este caso horizontal con una medida de 16:9) y se modifica la cámara de nuestra pantalla a esa resolución para poder verlo correctamente. Se asigna el script a la cámara de la pantalla, y se le asigna el "Aspecto" que hemos decidido que va a tener. Esto se calcula dividiendo nuestra anchura entre nuestra altura (Aspect = W/H). Al asignarle este valor el script se encarga de comprobar el tamaño de pantalla y asignar esa proporción al videojuego. Ejemplo:

Si el tamaño fuese de anchura = 400px y altura = 1000px al aplicar nuestro ratio se adaptaría así; 400/1000 = 0,4 (ratio de aspecto) con lo cual se vería de la siguiente forma, debido al salto que hay entre el ratio que posee la pantalla y la que nosotros deseamos y tenemos asignada al script:

Figura 14: *Prueba responsive screen*

Siempre ajustado a la resolución de pantalla. (En este caso se ha mostrado para una pantalla vertical, en el proyecto se trabaja con una pantalla horizontal y el ratio está asignado para ello)

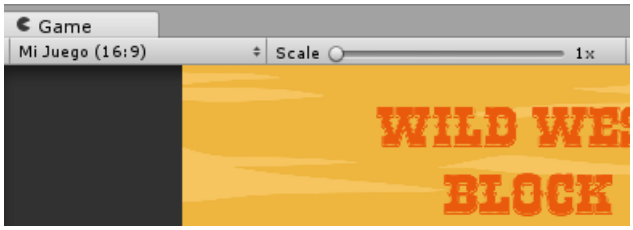


Figura 15: *Dimensión de la cámara del juego*

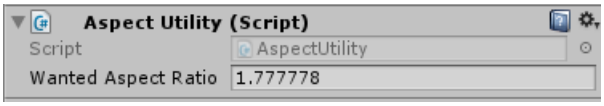


Figura 16: *radio asignado a nuestro script*

Creación de enemigos. Otra de los retos que ha presentado el videojuego ha sido la creación de enemigos y su instanciación. La creación masiva de enemigos mediante un controlador de enemigos y la aparición de múltiples enemigos a la vez en escena ha exigido la utilización de múltiples variables para poder tener un mejor manejo de ellos.

Manejo de las funciones Update, Start, Awake. Estas funciones son imprescindibles en Unity.

- **Awake**: Esta función siempre se llama antes de cualquier función Start y también justo después de que un prefab es instanciado. Se usa sobre todo para inicializar los valores de la pantalla.
- **Start**: es llamado antes de la primera actualización de frame solo si la instancia del script está activada. Solo se ejecuta una vez, por lo que es muy útil para acciones que queramos ejecutar al iniciar la pantalla.
- **Update**: se llama una vez por frame. Es la función principal para las actualizaciones de frames. Lo que quiere decir que si nuestro juego está funcionando a 30 fps, la función Update se ejecutará 30 veces cada segundo. Pero como la frecuencia de llamada de la función no será siempre la misma ya que ejecución de nuestro juego no será siempre homogénea, dependerá de la carga de objetos, efectos, etc. Por todo esto la función Update se suele usar para la el movimiento de objetos que no dependan de la física.

Manejo de funciones globales. Al crear un videojuego hay una serie de variables que vamos a incluir siempre en todos los niveles. Estas variables se suelen considerar como globales, ya que afectan a todo el juego. En este proyecto se ha definido un script de variables globales para manejar todos estos datos. La problemática vino en el cuándo cargar estas variables, dado que si se hacía a la vez que el menú en muchos casos daba problemas de compatibilidad. Por ello se creó un "Splash Screen" o pantalla de carga inicial, donde se cargan todas estas variables antes de iniciar el juego.

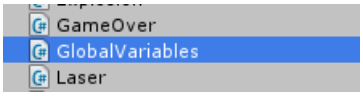


Figura 17: *Script de variables globales*

Figura 18: *Pantalla de carga inicial*

Serializar datos. La serialización es el proceso de convertir un objeto en una secuencia de bytes para conservarlo en memoria, una base de datos o un archivo. Su propósito principal es guardar el estado de un objeto para poder crearlo de nuevo cuando se necesita. En este TFG se realiza una serialización del "nivel pasado", de tal forma que se vayan desbloqueando los niveles a la vez que vamos superándolos, quedando el resto bloqueados por el momento. Unity por defecto te aporta un módulo para cargar y guardar datos llamado "**PlayerPrefs**", pero este módulo da diversos problemas además de que deja más vulnerable la aplicación a un posible hackeo de datos. Por ello se ha optado por crearse unos métodos propios de guardado y cargado que se lanzan al acabar un nivel y al iniciar la aplicación. Estos métodos consisten en crearse un archivo de forma binaria con los datos a guardar, de tal forma que están protegidos y podemos acceder a ellos cuando queramos para cargarlos.

```
    public static void Guardar(){
        BinaryFormatter bf = new BinaryFormatter();
        FileStream file = File.Create(rutaArchivo);

        DatosAGuardar datos = new DatosAGuardar();
        datos.nivelPasado = nivelPasado;
        bf.Serialize(file, datos);
        file.Close();

    }

    void Cargar(){
        if (File.Exists(rutaArchivo)){

            BinaryFormatter bf = new BinaryFormatter();
            FileStream file = File.Open(rutaArchivo, FileMode.Open);
            DatosAGuardar datos = (DatosAGuardar) bf.Deserialize(file);
            nivelPasado = datos.nivelPasado;
            file.Close();

        }else{
            nivelPasado = 1;
        }
    }
}

[Serializable]
class DatosAGuardar{
    public int nivelPasado;
}
```

Figura 19: *Métodos de carga y guardado*

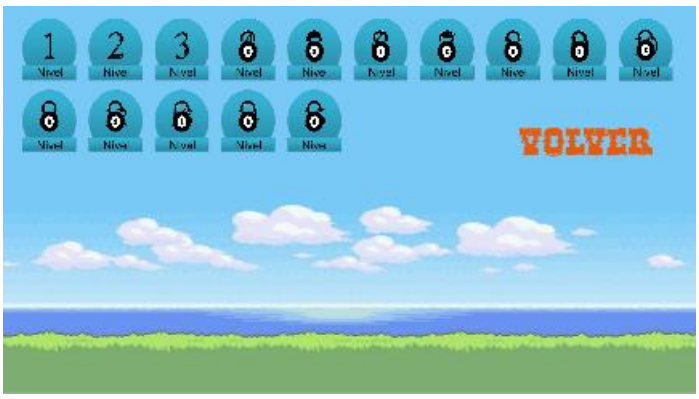


Figura 20: *Serialización y bloqueo de niveles*

Creación de diseños gráficos. Sin lugar a dudas, uno de los mayores retos y de los más importantes en un videojuego, ha sido la creación o búsqueda de imágenes, efectos, animaciones y sonidos que hacen falta para la creación de la APP. Sin tener experiencia en el campo, he tenido que recurrir a herramientas online tales como "Piskel" o "freesound" que me han ayudado a conseguir el resultado final esperado.

4.1.4 Descripción de los elementos tecnológicos

A continuación se describen y explican los elementos más característicos que contiene el juego, tales como interfaces, niveles, enemigos, power ups y demás.

4.1.4.1 Interfaz

En Unity la interfaz se coloca mediante un objeto canvas, al cual se le pueden asignar botones, textos, imágenes, etc. Cada objeto se modifica de acuerdo a como queda mejor visualmente.

En cada **nivel** se muestra, en la parte superior izquierda, el número de balas que posee el pistolero, el número de bloques que quedan por salir y las vidas que tiene el PJ antes de morir. Justo debajo aparece nuestro pistolero, el cual realizará la acción de disparar si se pulsa sobre él. En la parte superior derecha tenemos el botón de "Pause", el cual detiene el juego al ser pulsado y despliega un menú pause oscureciendo el resto de la escena. Al iniciar el nivel empieza con una cuenta atrás desde tres. Una vez finalizada empiezan a caer bloques enemigos. A partir de cierto nivel, aparece para ayudarnos un segundo pistolero en la parte derecha de la pantalla, con la misma mecánica que el primer pistolero.

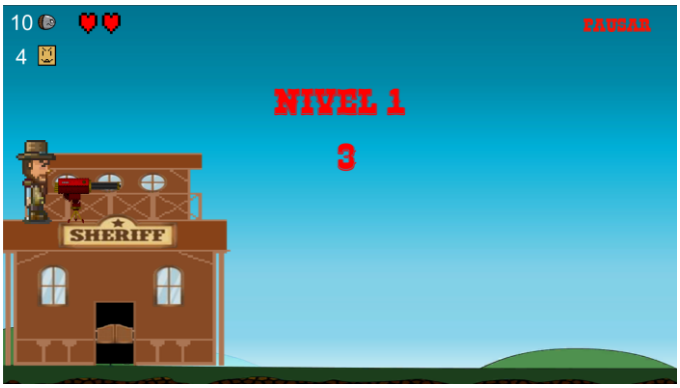


Figura 21: *Interfaz*

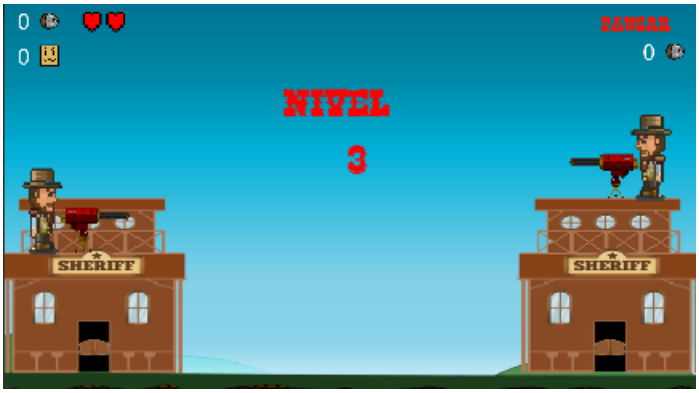


Figura 22: *Interfaz dos pistoleros*

El **menú pause** cuenta con tres botones: **Continuar** (continua con la ejecución del juego), **Reiniciar** (reinicia el nivel en el que te encuentras), **Salir** (sale al menú principal).

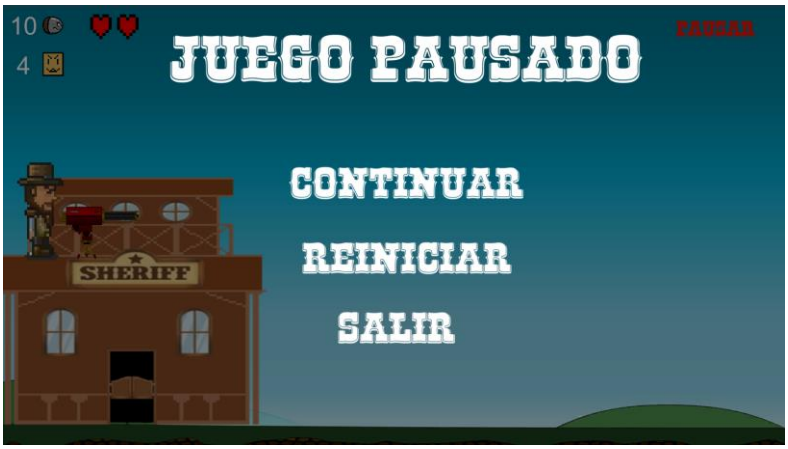


Figura 23: *Interfaz Pause*

La **interfaz del menú selector de niveles** se compone de varios botones, cada uno envía a un nivel distinto. Están dispuestos en líneas de 10 botones, con un botón de **Volver** (te hace regresar al menú principal). Corren un par de animaciones por la pantalla con las que no se puede interactuar.

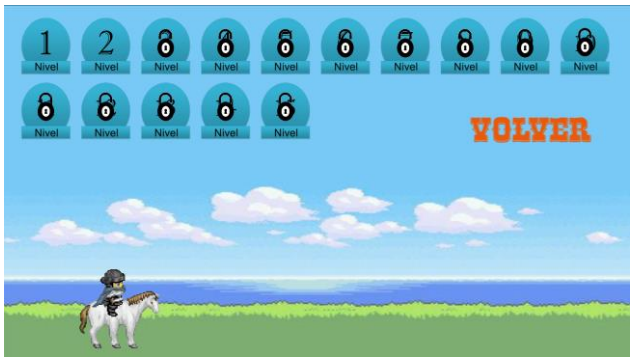


Figura 24: *Interfaz Menú Selector de Niveles*

La **interfaz del menú de Game Over** se compone de un botón **Jugar** (reinicia el nivel) y un botón **Menú** (retorna al menú) y ciertos datos de interés como cuantos enemigos has destruido y cuantos han sobrevivido.

Figura 25: *Interfaz Menú Game Over*

La **interfaz del menú de Winner** se compone de un botón **Next** (pasa al siguiente nivel) y un botón **Menú** (retorna al menú) y ciertos datos de interés como cuantos enemigos has destruido y cuantos han sobrevivido.

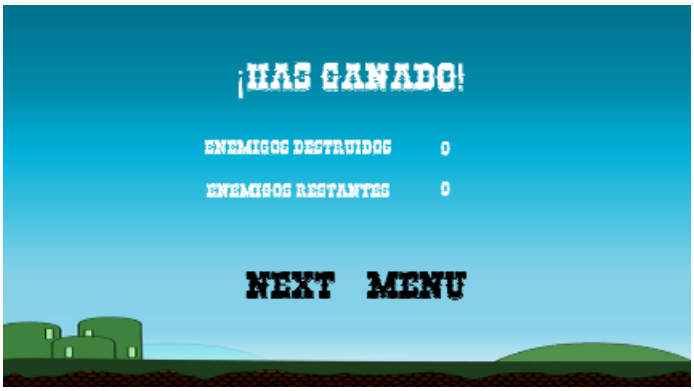


Figura 26: *Interfaz Menú Winner*

4.1.4.2 Niveles

El conjunto de niveles se divide en varias fases, cada una compuesta por cinco niveles basados en destruir varias oleadas de enemigos. Cada nivel puede durar desde los 20 segundos hasta los 120 segundos. Los niveles se crean totalmente aleatorios, la instanciación de enemigos se produce aleatoriamente dentro de un rango preseleccionado, lo que implica que un mismo nivel nunca va a ser igual dos veces. Los niveles entre sí van aumentando la dificultad y varía la combinación de enemigos.

- **Nivel 1-5**: Son niveles cortos y fáciles, con un solo pistolero, donde caen bloques por una sola zona de la pantalla a una velocidad lenta y con un tiempo considerable entre cada uno de ellos.

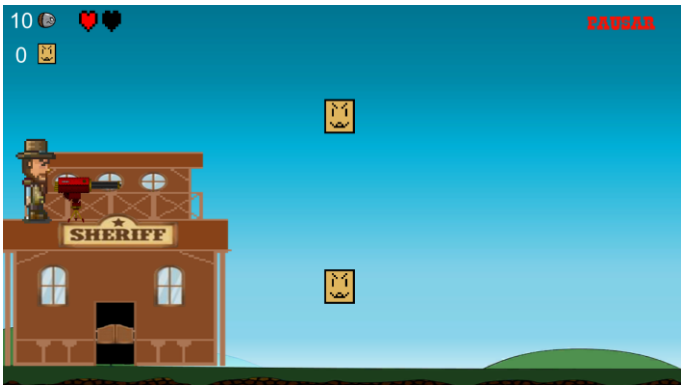


Figura 27: *Nivel 1-5*

- **Nivel 6-10**: Son niveles más largos y más complicados, con un solo pistolero, donde caen bloques por dos zonas de la pantalla a una velocidad media y con un tiempo más corto entre cada uno de ellos.

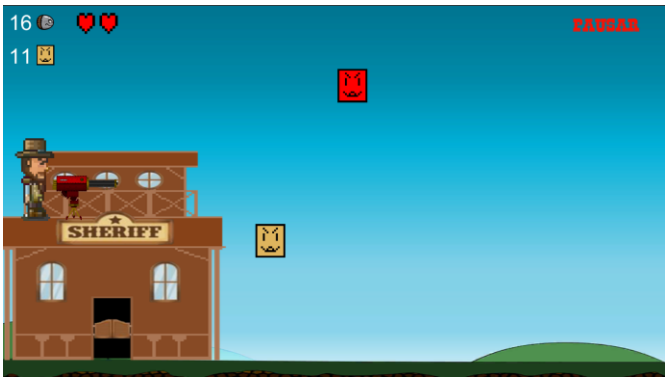


Figura 28: *Nivel 6-10*

- **Nivel 11-15**: Son niveles largos y complicados, con dos pistoleros, donde caen bloques por dos zonas de la pantalla a una velocidad media/rápida y con un tiempo más corto entre cada uno de ellos.

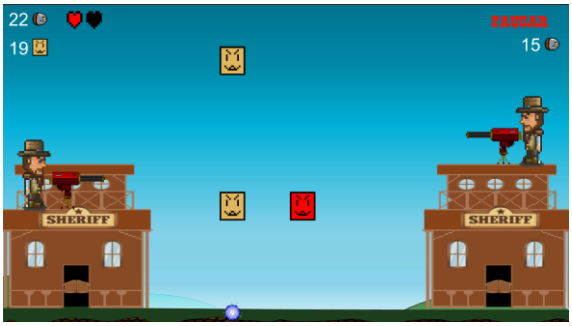


Figura 29: *Nivel 11-15*

- **Nivel 16-20**: Son niveles extremos y complicados, con dos pistoleros, donde caen bloques por tres zonas de la pantalla a una velocidad rápida y con un tiempo más corto entre cada uno de ellos poniendo a prueba los reflejos.

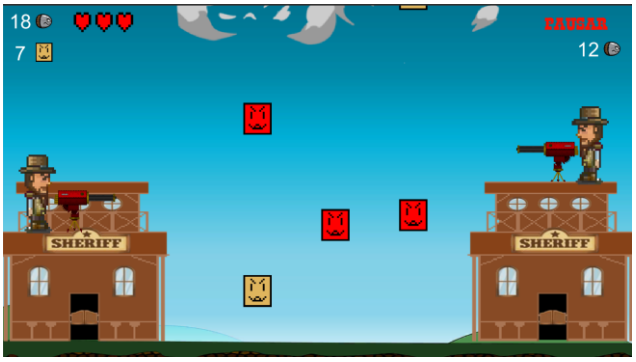


Figura 30: *Nivel 16-20*

4.1.4.3 Tipos de enemigos y su programación

Se pueden diferenciar dos tipos de enemigos, los **bloques normales** (aquellos que hay que destruir para ganar) y los **bloques bomba** (aquellos que te eliminan si les disparas).

Los **bloques normales** son enemigos básicos cuya única función es caer desde la parte superior de la pantalla hasta la inferior y se destruyen de un disparo.

Los **bloques bomba** son similares a los bloques normales pero con la diferencia de que estos explotan y te matan si les disparas, con lo que saltas automáticamente a la ventana de Game Over.

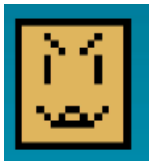


Figura 31: *Enemigo Bloque Normal*

Figura 32: *Enemigo Bloque Bomba*

Figura 33: *Bomba modificada*

4.1.4.4 Armas

Actualmente existe un tipo de arma. Se trata de una ametralladora que dispara proyectiles a una velocidad elevada y de uno en uno.

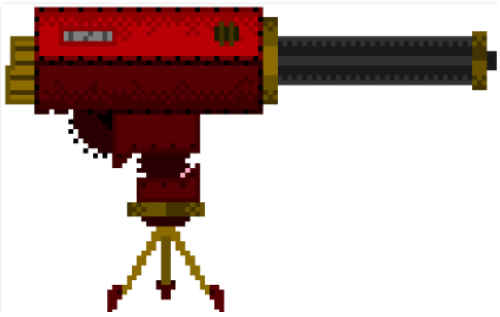


Figura 34: *Arma 1 (ametralladora)*

4.1.4.5 Banda sonora y efectos sonoros

Otro de los aspectos fundamentales a la hora de realizar un videojuego es la banda sonora y los efectos sonoros que posea. Tener una música acorde a la temática del juego y que trasmita una sensación de inmersión es importante para el desempeño del juego. Todas las piezas utilizadas se pueden encontrar de manera gratuita en la página de "**freesound.org**".

- **Country Sound:** se puede encontrar en los menús y submenús del juego.
- **Winner Sound:** se encuentra en la pantalla Winner.
- **Game Over Sound:** se encuentra en la pantalla Game Over.
- **Shoot:** se reproduce cuando realizamos un disparo.
- **Empty Gun:** se reproduce cuando disparamos sin tener balas.
- **Explosion:** se reproduce cuando se destruye un enemigo.
- **Apper Enemy:** se reproduce cuando aparece un enemigo.

4.1.4.6 Animaciones

Para que el juego y las acciones que se realicen den una sensación de credibilidad, son muy importantes las animaciones. Estas, pueden usarse tanto para decorar o ilustrar una pantalla, como para dar un efecto perfecto a una acción iniciada por el usuario. En este TFG se han tratado las siguientes animaciones:

- **Loading**: se activa cuando inicia el juego.
- **Bird**: se activa en el menú de niveles.
- **Horse**: se activa en el menú de niveles.
- **Explosion**: se activa cuando se destruye a un enemigo.
- **Fire**: se activa cuando se dispara el arma.
- **Smoke**: se activa cuando se crea un enemigo.

4.1.4.7 Power Ups

Además, se han añadido una serie de potenciadores o Power Ups, que hacen que el juego resulte más entretenido y vistoso:

- **Cargador de munición.** Iniciar el nivel con un número bajo de balas y hacer aparecer cargadores necesarios para la superación del nivel, haría el juego más difícil, imprevisible y entretenido.

Figura 35: *Power Up munición*

- **Vidas.** Dar la posibilidad de regenerar las vidas, sobre todo en niveles avanzados, puede ayudar al jugador a sobrevivir en más de un nivel. Además incrementaría la audacia del jugador y pondría a prueba la velocidad de reacción y toma de decisiones al tener que decidir cuándo es buen momento para buscar este Power Up.

Figura 36: *Power Up vida*

4.2 Diseño y desarrollo de la accesibilidad

En este apartado se van a tratar y explicar las diferentes formas de accesibilidad adoptadas en el proyecto, así como las diferentes dificultades que han aparecido en su desarrollo.

4.2.1 Accesibilidad

Para tratar las distintas discapacidades, se ha creado un menú opciones con todas ellas lo que permite al usuario ir eligiendo cual quiere activar o desactivar.

Figura 37: *Menú Opciones con todas las accesibilidades*

Si selecciona **daltonismo**, todas las escenas cambiarán automáticamente a una escala de grises, además se han modificado las formas de los enemigos, de tal forma que no dependa del color saber a cual nos enfrentamos. De esta manera las personas que tenga problemas con los colores (sobre todo los casos más severos) no deben de tener ningún problema para desempeñar el juego con normalidad.

Figura 38: *Bomba cambiada por enemigo rojo*

Figura 39: *Fondo en B/N*

Si se selecciona **Discapacidad Auditiva**, para llamar igualmente la atención del usuario, y que pueda tener los mismos matices el juego, se ha añadido una función de vibración al destruir enemigos y una serie de etiqueta al disparar.

Figura 40: *Onomatopeya al disparar*

Figura 41: *Onomatopeya al disparar sin balas*

Con la **Discapacidad Motora** activada se amplía la zona de contacto de la pantalla para acciones como disparar, pausar, salir, etc.

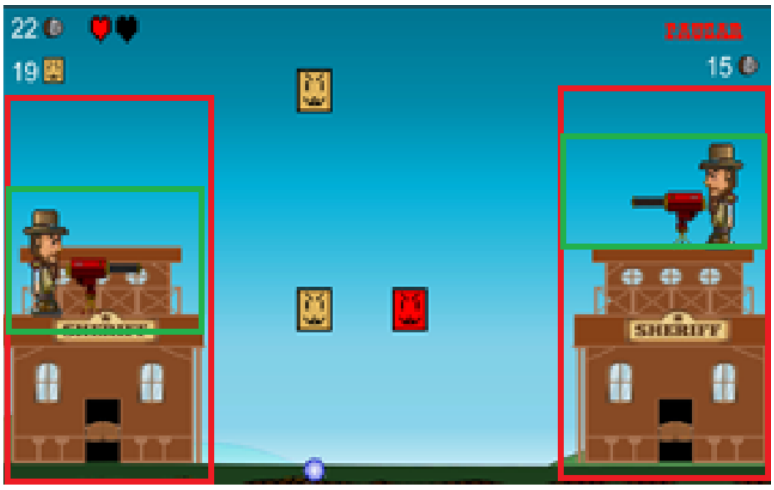


Figura 42: *Zona de contacto ampliada (Rojo: zona ampliada, Verde: zona sin ampliar)*

Si escogen **Discapacidad Visual**, al igual que en la discapacidad anterior, se amplía la zona de contacto y además se calcula la trayectoria y velocidad a la que cae de tal forma que vibra cuando el usuario pueda disparar y destruir el bloque. De esta forma, vibra y suena una voz "left" si ha de pulsar la parte izquierda de la pantalla, y vibra y suena otra voz "right" si ha de tocar la parte derecha. En caso de caer una bomba vibrará y sonara un sonido de alarma para que no pulsemos.

Por último, existen dos **lenguajes** a elegir (Inglés y Español), los cuales cambian todos los textos del juego a un idioma u otro.

Figura 43: *Imagen lenguajes en menú opciones*

4.2.2 Principales retos tecnológicos

Se han tenido diversas dificultades en el desarrollo de la adaptación del videojuego a las características y mecánicas de los distintos mecanismos de accesibilidad, las más resaltables han sido:

Añadir la vibración al destruir los objetos. Añadir una vibración en el momento justo, o jugar con la duración de la vibración, fue uno de los primeros problemas. Para facilitar esta labor, Unity posee unas funciones de vibración (**Vibrate**) que nos permiten instanciar esta utilidad. Esto conlleva a que en el teléfono, será necesario dar permisos a la aplicación para poder manejar la vibración del mismo.

```
using UnityEngine;
using System.Collections;

public class ExampleClass : MonoBehaviour
{
    void OnGUI()
    {
        if (GUI.Button(new Rect(0, 10, 100, 32), "Vibrate!"))
            Handheld.Vibrate();
    }
}
```

Figura 44: *Instanciación de método para vibrar*

Jugar con la escala de grises. Otro de los problemas fue la manera en la que se debía cambiar el juego a una escala de grises en blanco y negro. Inicialmente se planteó, por desconocimiento de otra forma mejor, la posibilidad de intercambiar todos los Sprites del juego de tal forma que cambiasen a otro exactamente igual, pero con la escala de grises deseada. Esto conllevaba duplicar la cantidad de imágenes, Sprites y por lo tanto muchísimo tiempo y casi se duplicaba el tamaño final de nuestra aplicación. Tras investigar, se descubrió una librería estándar, que permitía insertar una escala de colores prediseñada a la cámara de la pantalla. De esta manera, al activar dicho script, toda la escena cambiaba a la escala predefinida.

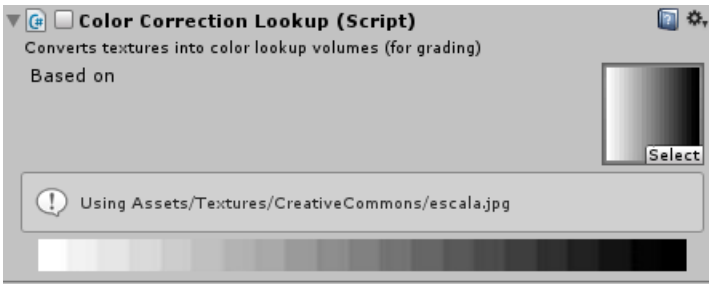


Figura 45: *Escala de grises*

Ampliar la zona de contacto. Para cambiar la zona de contacto, se tuvo que tener en cuenta la posición de los objetos que se creaban y de los ya instaurados en la pantalla, debido a que estos entraban en conflicto con esa zona de contacto. Manejar el cambio de posiciones en los distintos ejes solucionó el problema.

Calcular la distancia a la que destruir el objeto. Para esto, se ha tenido que tener en cuenta diversos factores en cada una de las escenas, tales como gravedad, velocidad del proyectil, velocidad de aparición del enemigo, de tal forma que los cálculos se basan siempre en la construcción de triángulos.

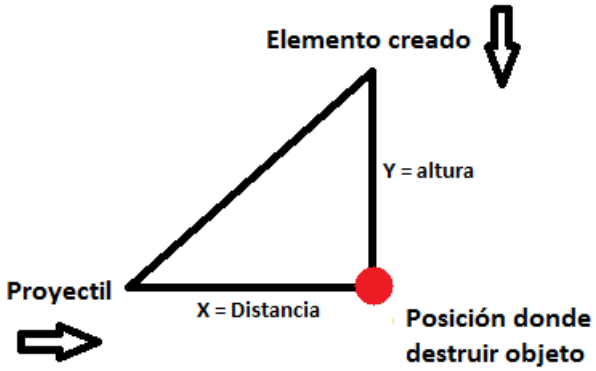


Figura 46: *Cálculo de momento de vibración*

Añadir reconocimiento de voz. Para las discapacidades visuales y motoras, se pretendía introducir un reconocer de voz de tal manera que en los menús, pudieses avanzar con la voz. La integración de esta parte a la aplicación ha resultado imposible con los conocimientos actuales. Se ha creado un sistema paralelo de reconocimiento de voz para comprobar que funciona, pero el problema viene al introducirlo en nuestro videojuego con el resto de funcionalidades. Dado que no se aíslan los ruidos ambiente, al intentar reconocer la voz, con un ruido elevado de fondo o con la propia música del juego entra en conflicto y no hace caso a las instrucciones que mandamos. Por ello se adjunta al trabajo el código correspondiente al reconocedor de voz, pero no está integrado en la aplicación.

4.3 Publicación del juego en Play Store

Una vez completado el juego, se ha procedido a publicarlo en Play Store, esta es la tienda de aplicaciones de los dispositivos Android, donde se pueden adquirir todo tipo de aplicaciones, ya sean de pago o gratuitas.

Para poder subir contenido a la tienda hay que tener una cuenta de desarrollador de Google Play, para ello es necesaria una cuenta de Google, y el pago de una cuota anual de 25€. Una vez creada la cuenta se puede proceder a publicar la aplicación.

Para publicar la aplicación se debe rellenar una ficha con datos de la misma (como el nombre, descripción breve, iconos, capturas de pantalla, etc.). Hay que responder un cuestionario que nos aplicará la clasificación por edades y seleccionar si va a ser de pago o gratuita.

Después se pasará un breve examen interno al desplegar la aplicación, para ver si se infringen derechos de Copyright, de ser así se nos informará que ha de ser cambiado en un breve lapso de tiempo si se quiere desplegar finalmente el juego.

Una última cosa a tener en cuenta es que para publicar una aplicación en Google Play hay que tenerla firmada mediante una "**Keystore**". Esto se puede hacer desde el apartado Settings de Unity.

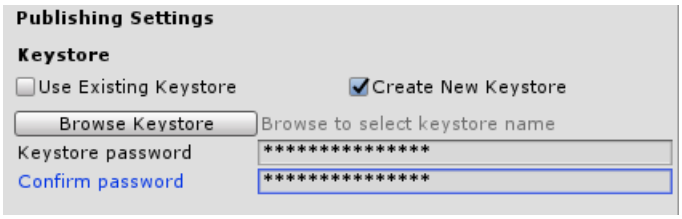


Figura 47: *creación de KeyStore*

Unity proporciona la opción de usar una clave ya existente o de crearte tú una nueva. En este caso nos vamos a crear una nueva. Para ello, seleccionamos donde queremos que sea guardada, introducimos una contraseña y pulsamos sobre crear nueva clave. Se nos abrirá una pantalla donde nos pedirá información referente a nosotros o nuestra empresa, así como una contraseña.

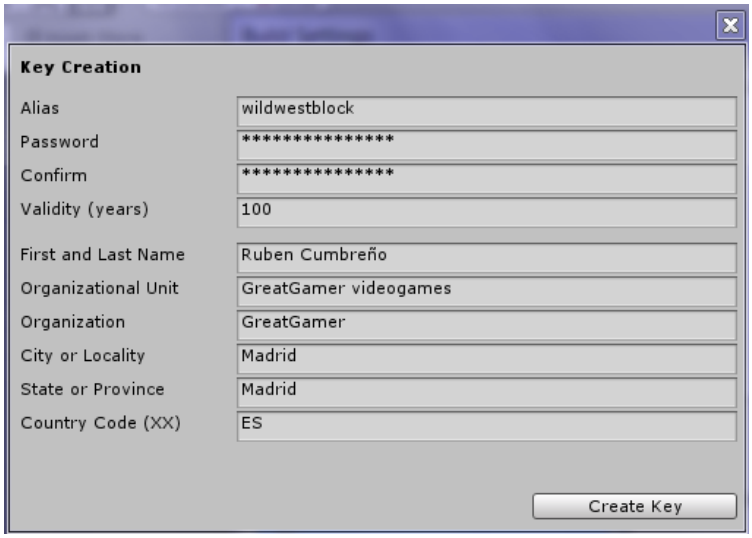


Figura 48: *Creación de Key*

Una vez creada hay que guardar bien la Key, porque será necesaria siempre que queramos actualizar la APP.

Una vez hecho esto, vamos a proceder a subir nuestra APK, creada ya con una firma, al Play Store. Lo primero que necesitamos es tener una cuenta como desarrollador. Una vez creada, accedemos a un menú con todas nuestras aplicaciones subidas y donde podemos ver nuestros datos, informes sobre las aplicaciones, etc. Si pulsamos sobre "crear nueva aplicación", se nos abre una pantalla con una serie de pasos que debemos ir completando.

Primero debemos escoger que tipo de aplicación vamos a crear y nos dan tres opciones para ello. Una aplicación **Alfa**, **Beta** o en **Producción**.

Figura 49: *elección de tipo de App*

Después se nos pide rellenar una serie de campos respectivos a nuestro videojuego, como descripciones, imágenes que debemos insertar, a quien va dirigida, países en los que queremos publicarla, si contiene anuncios, etc.

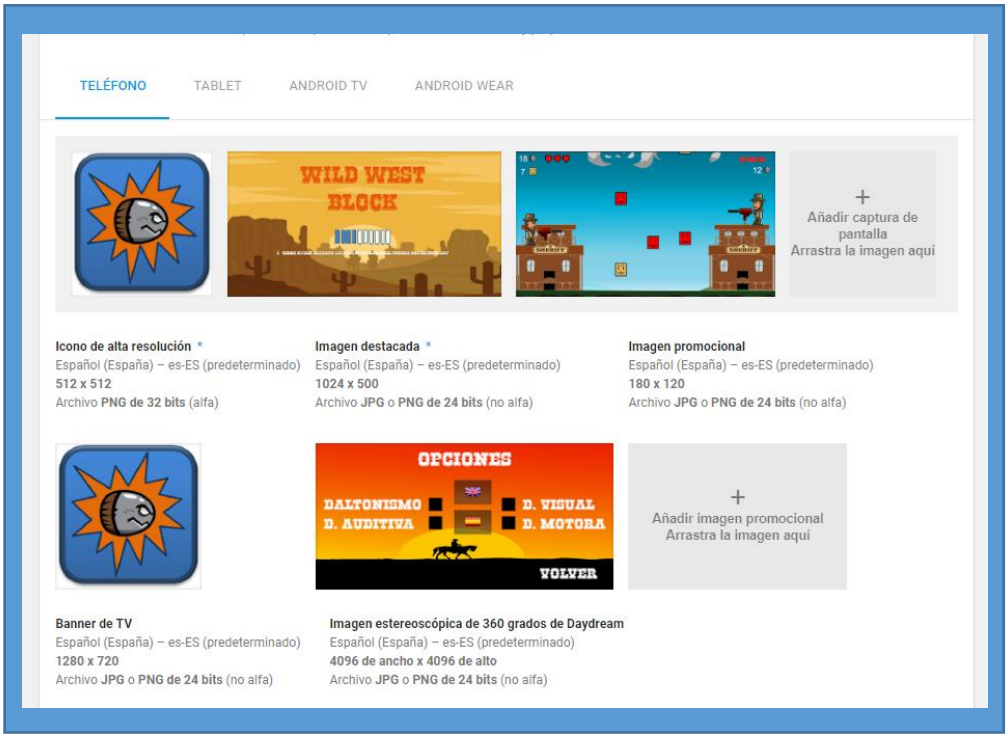


Figura 50: *imágenes obligatorias para subir apk*

Figura 51: *comprobar si aplicación de pago*

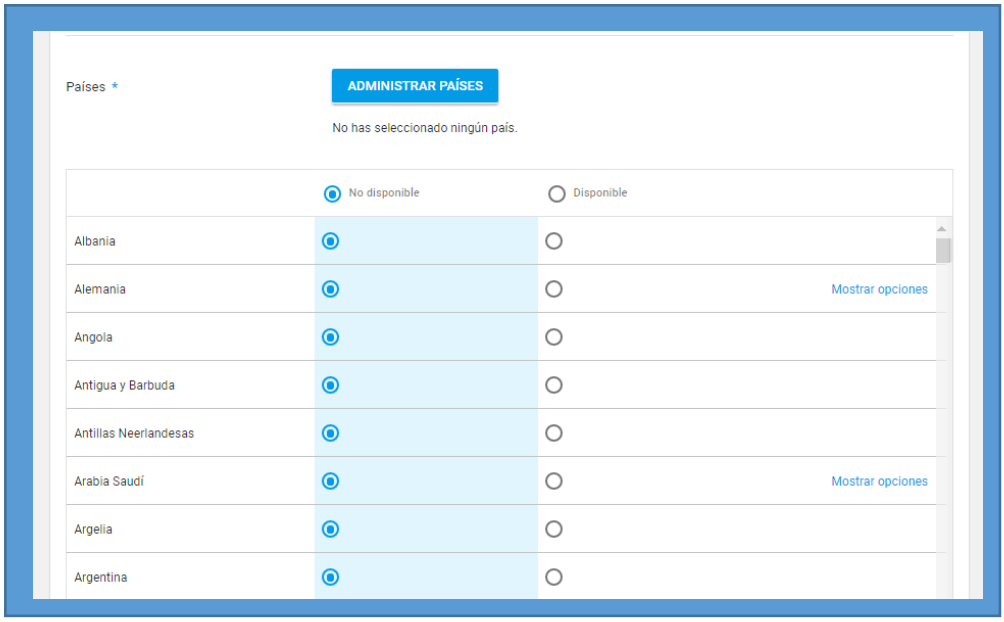


Figura 52: *selección de países donde será publicado*

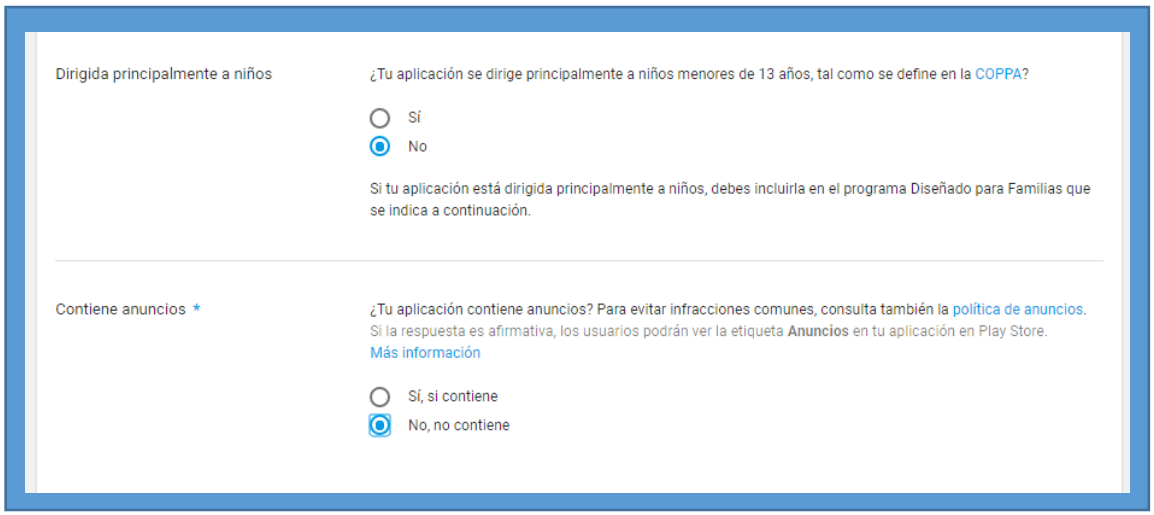


Figura 53: *selección público objetivo*

Pasada esta fase accedemos a otra donde debemos subir nuestra APK y realizar un cuestionario para que nos categoricen según contenido de drogas, violencia, etc.

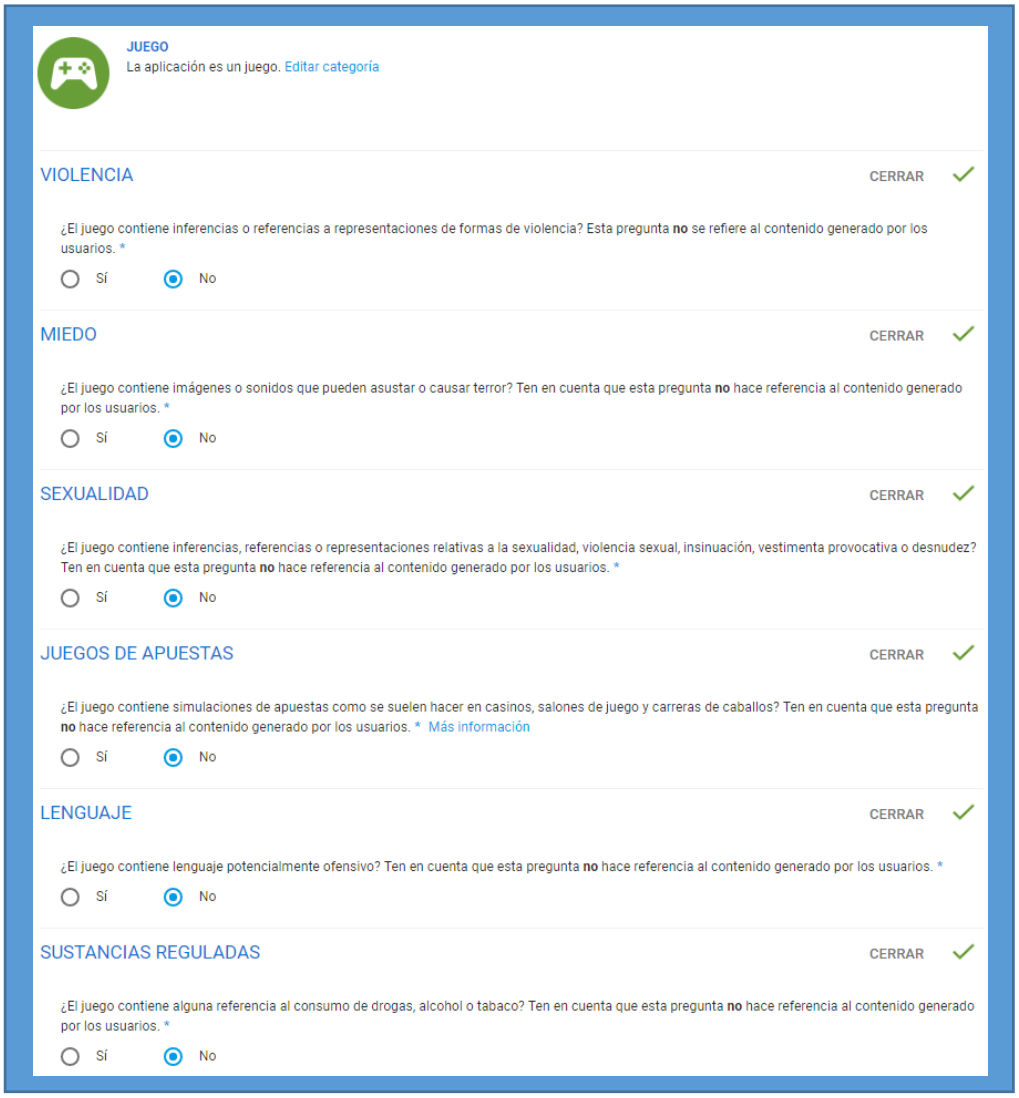


Figura 54: *cuestionario categorías*

Figura 55: *información autogenerada*

Por último subimos nuestra aplicación, y tras unos minutos, en los cuales Google comprueba si infringes algún derecho de autor o tu APK es algún tipo de virus, la aplicación aparece en la Play Store.

https://play.google.com/store/apps/details?id=com.GreatGamer.games.WildWestBlock

Desde la Web ahora podemos contemplar datos sobre la cantidad de gente que descarga nuestra aplicación u opiniones.

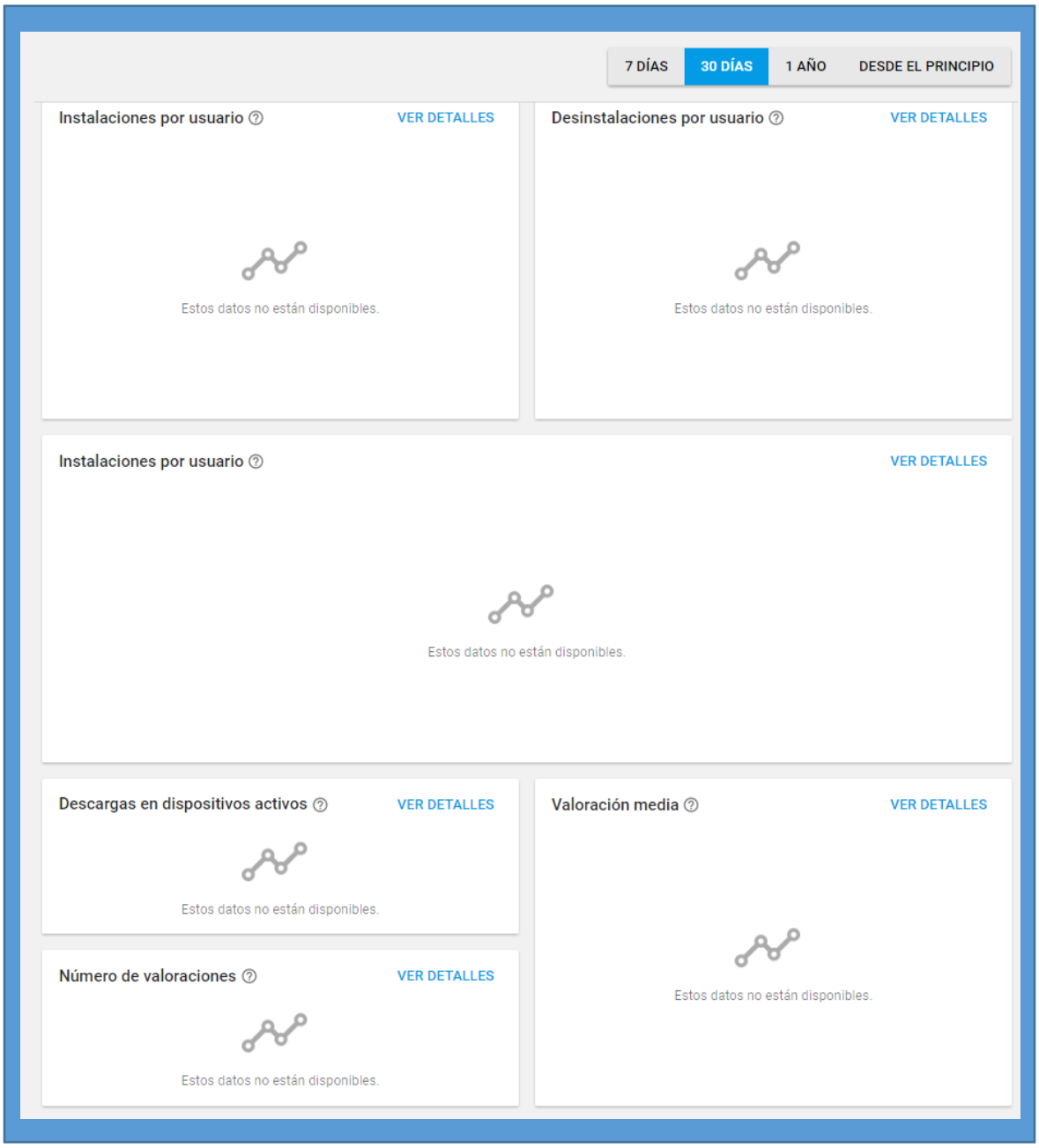


Figura 56: *datos sobre nuestra aplicación subida*

4.4 Análisis de mercado

La industria del videojuego ha evolucionado muchísimo en los últimos años. Según el informe oficial producido por la (E.S.A) sobre los videojuegos este último año, las ventas totales de software para videojuegos superaron los 24.500 millones de dólares y ya hay más de 220.000 empleados solo en EE.UU dedicados al sector. Solamente mirando la cantidad de empresas del sector en los últimos 2 años, vemos que ha aumentado un 14% y los sueldos medios de los trabajadores dedicados al desarrollo ronda entorno a los 97.000€ en EE.UU (en partes del mundo como Europa esta cifra disminuye hasta los 50.000€).

Con esto se observa que el sector del videojuego sigue en auge y no parece que vaya a parar de crecer, generando millones de euros anualmente y creando miles de puestos de trabajo.

Podemos ver en la siguiente tabla, los datos recogidos por la (E.S.A) referentes a los beneficios en las ventas de los últimos 8 años en este sector.

Tabla 1: *Tabla de beneficios de ventas de ordenadores y videojuegos en los últimos 8 años.*

Año	Ventas(billones $)	Cambio anual (%)
2009	10.1 $	...
2010	17.1 $	69,3 %
2011	16.7 $	-2.3 %
2012	15.2 $	-9.0 %
2013	15.4 $	1.3 %
2014	15.4 $	0.0 %
2015	16.5 $	7.1 %
2016	24.5 $	48.5 %

Esto hace ver que el mercado constantemente está en auge, y los usuarios cada vez demandan más y más productos que poder tener. Esa gran demanda concluye en un aumento de los distintos puestos de trabajo que integran un equipo dentro del desarrollo de un videojuego. A continuación se van a ver los datos económicos de los distintos trabajos y una comparativa y conclusión de estos datos.

Programadores

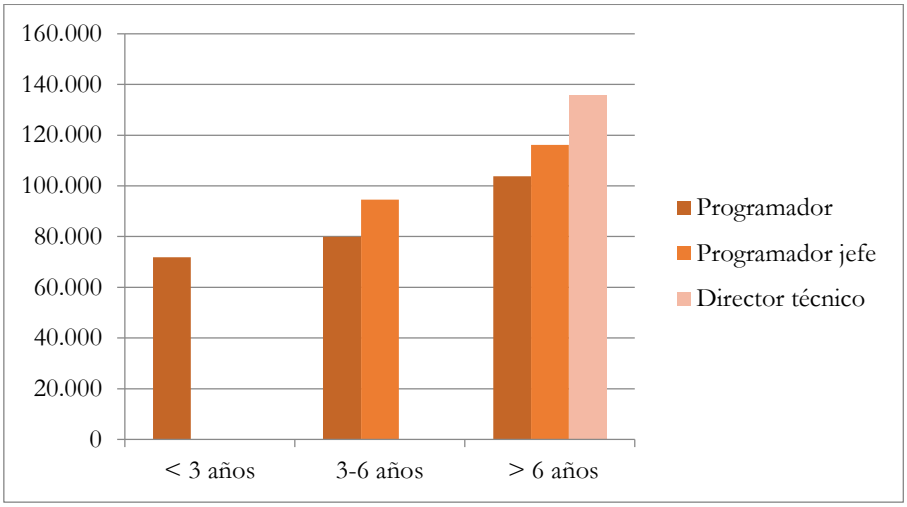


Figura 57: *Sueldos de los programados según experiencia y cargo*

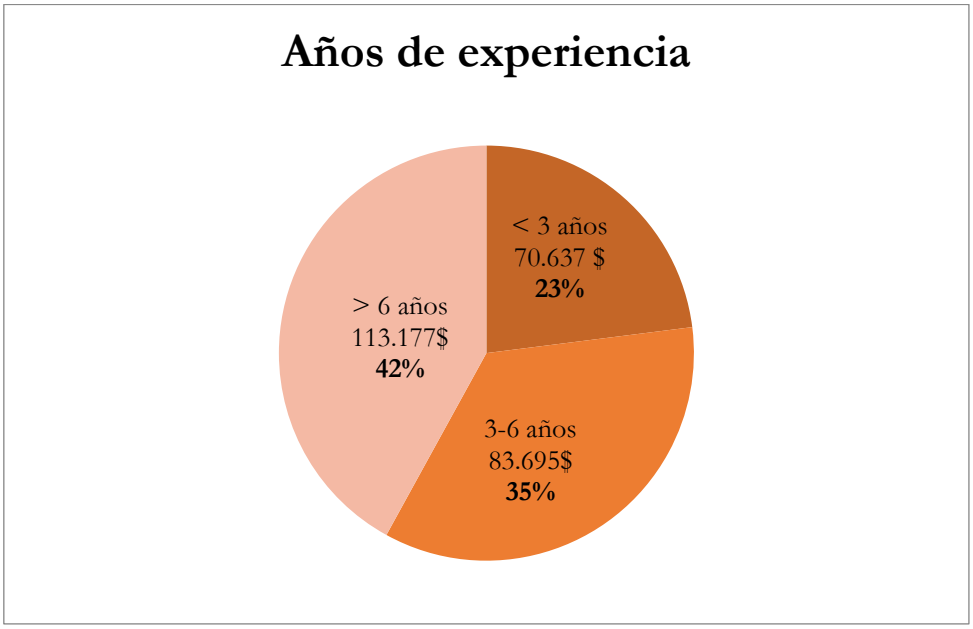


Figura 58: *% de trabajadores y años en la industria*

Tabla 2: *Tabla comparativa entre hombres y mujeres*

Sexo	Representación	Salario medio
Hombre	95%	$93,977
Mujer	5%	$79,318

Artista y animador de videojuegos

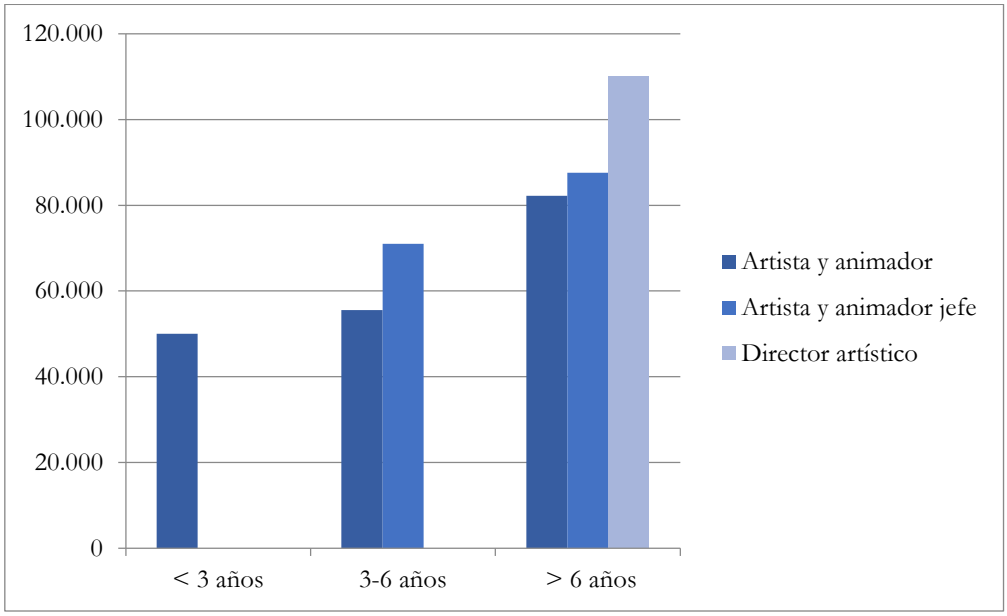


Figura 59: *Sueldos de los Artista y animador de videojuegos según experiencia y cargo*

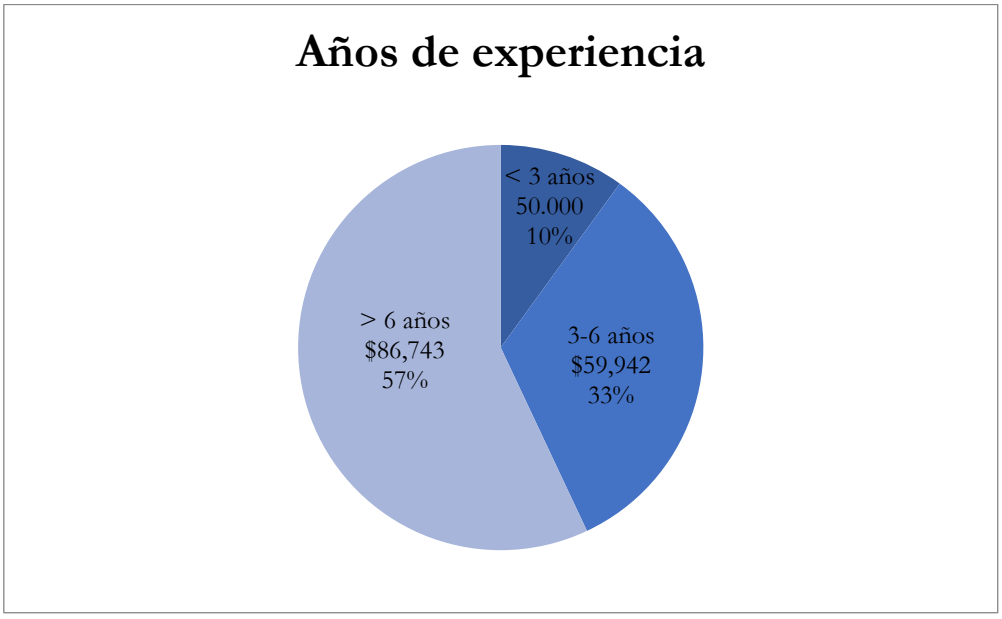


Figura 60: *% de trabajadores y años en la industria*

Tabla 3: *Tabla comparativa entre hombres y mujeres*

Sexo	Representación	Salario medio
Hombre	91%	$76,054
Mujer	9%	$55,909

Diseñador de videojuegos

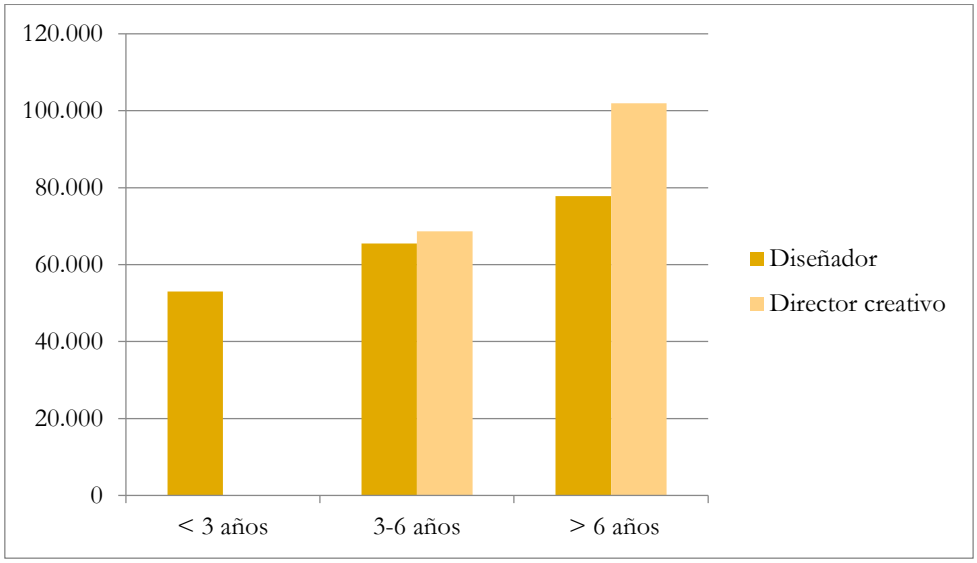


Figura 61: *Sueldos de los Diseñadores de videojuegos según experiencia y cargo*

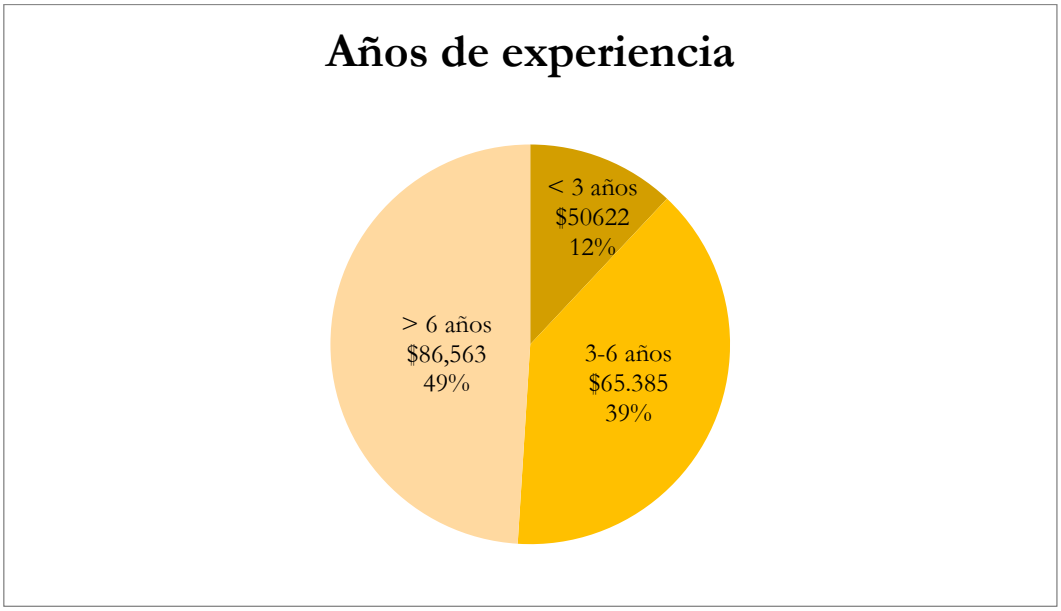


Figura 62: *% de trabajadores y años en la industria*

Tabla 4: *Tabla comparativa entre hombres y mujeres*

Sexo	Representación	Salario medio
Hombre	87%	$74,448
Mujer	13%	$70,000

Productores de Videojuegos

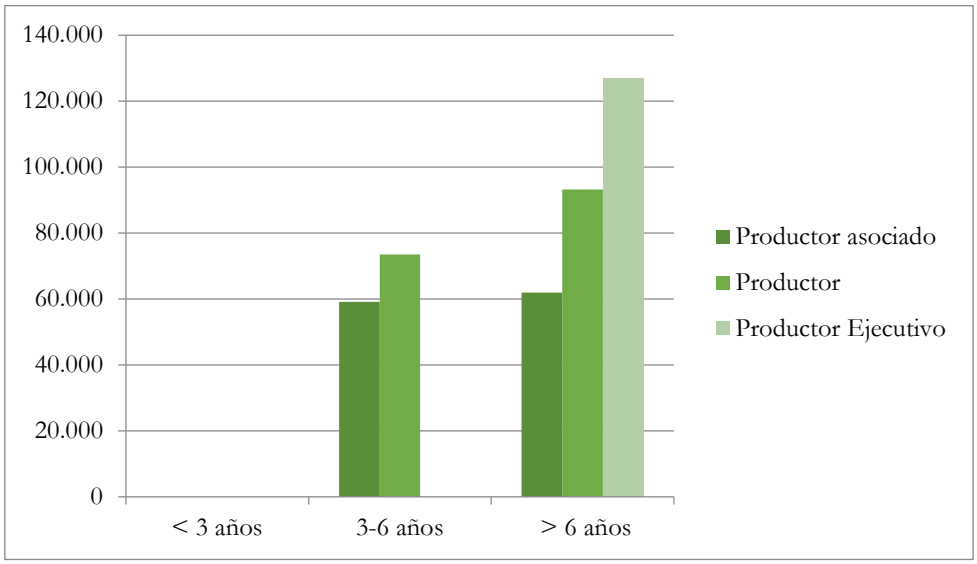


Figura 63: *Sueldos de los Productores de videojuegos según experiencia y cargo*

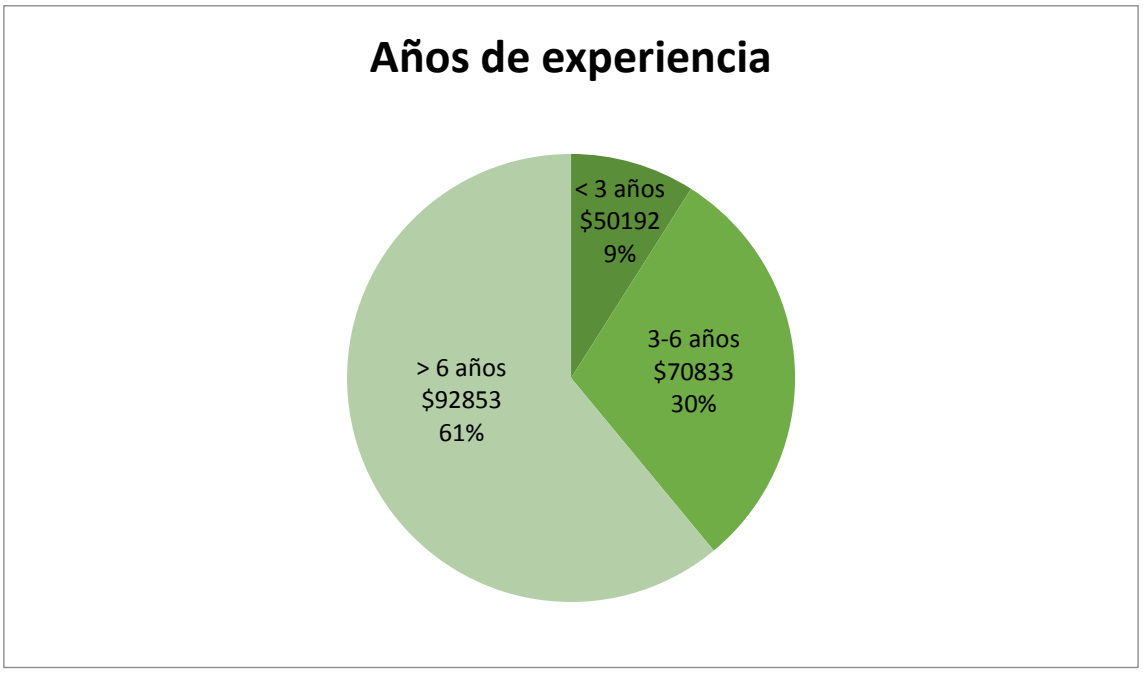


Figura 64: *% de trabajadores y años en la industria*

Tabla 5: *Tabla comparativa entre hombres y mujeres*

Sexo	Representación	Salario medio
Hombre	78%	$84,151
Mujer	22%	$75,726

Profesional de audio

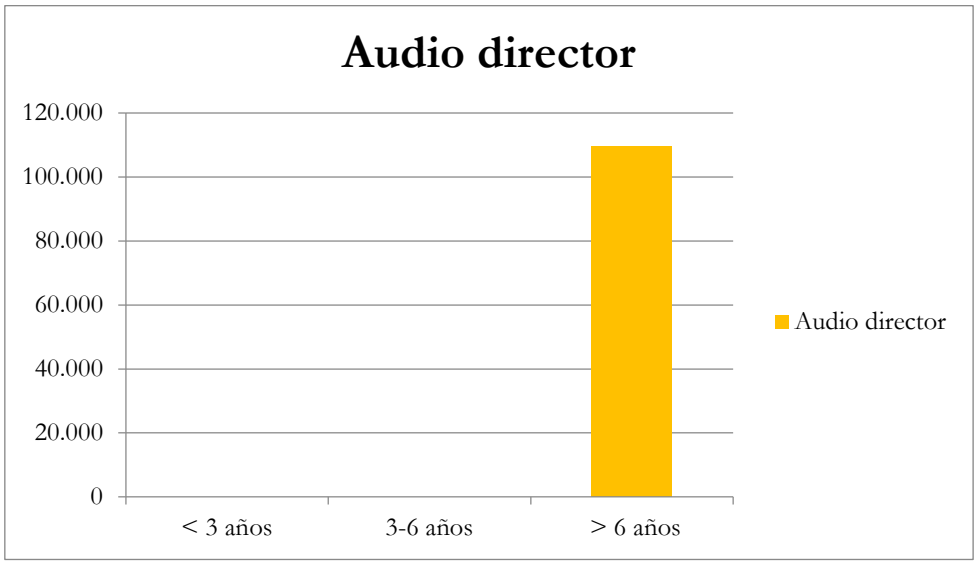


Figura 65: *Sueldos de los Profesionales de audio de videojuegos según experiencia y cargo*

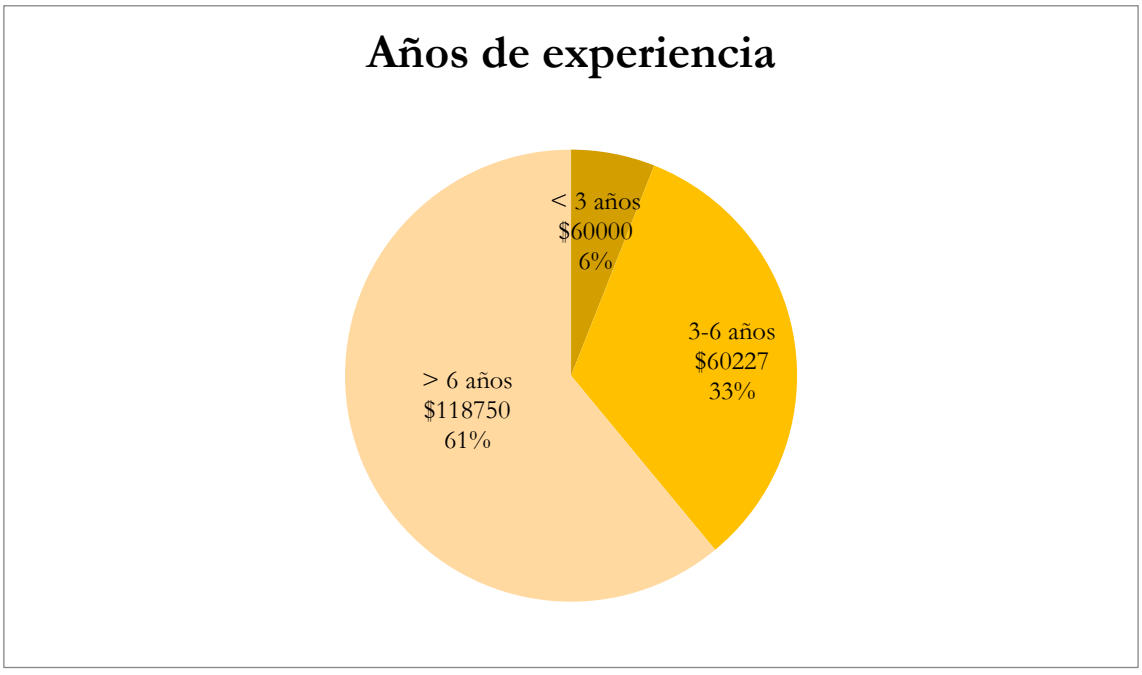


Figura 66: *% de trabajadores y años en la industria*

Tabla 6: *Tabla comparativa entre hombres y mujeres*

Sexo	Representación	Salario medio
Hombre	91%	$98,500
Mujer	9%	$67,500

Tester de Videojuegos

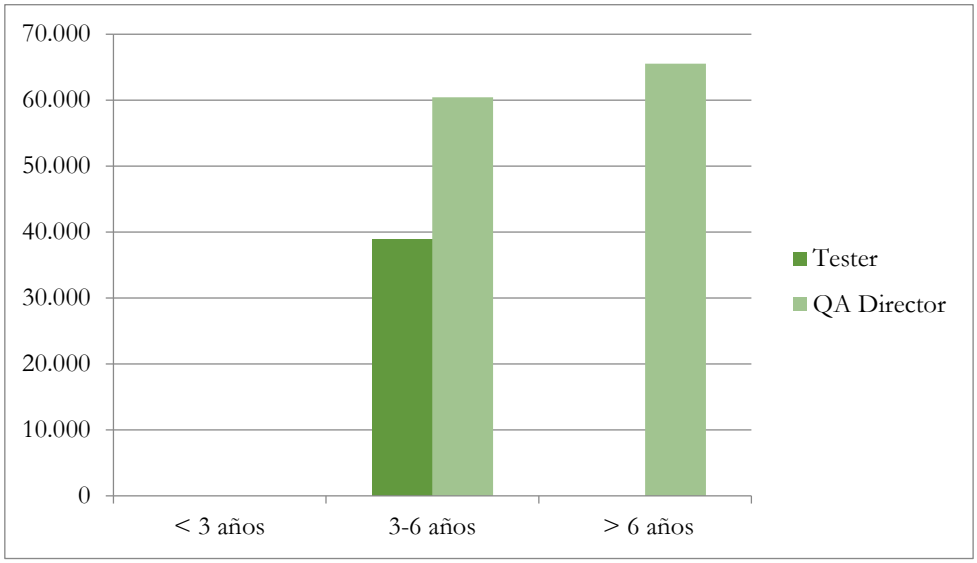


Figura 67: *Sueldos de los Tester de videojuegos según experiencia y cargo*

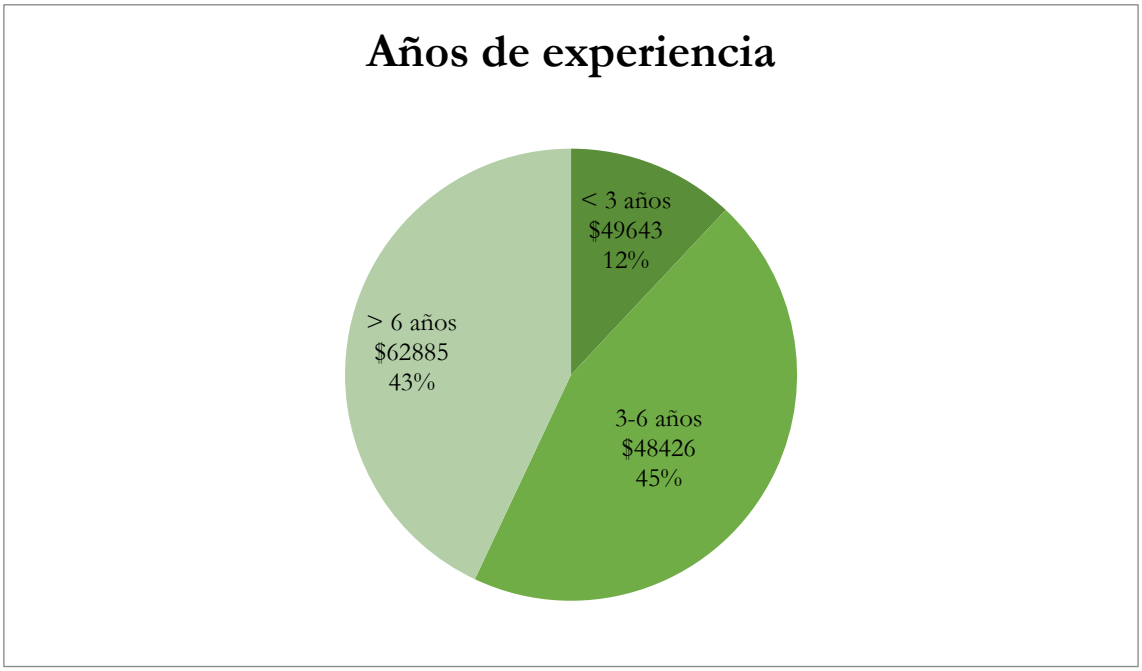


Figura 68: *% de trabajadores y años en la industria*

Tabla 7: *Tabla comparativa entre hombres y mujeres*

Sexo	Representación	Salario medio
Hombre	88%	$54,576
Mujer	12%	$56,786

Manager / gestores del Videojuego

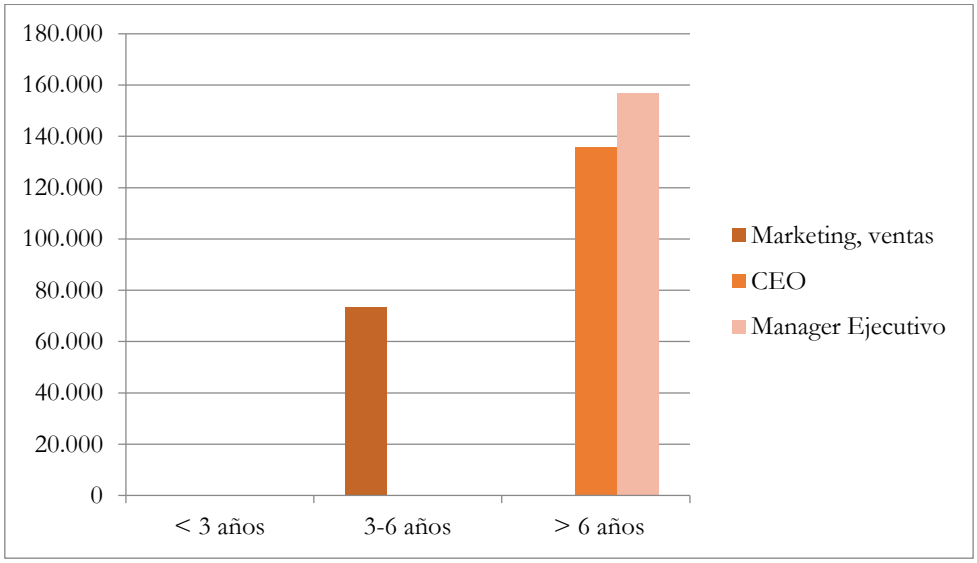


Figura 69: *Sueldos de los Managers de videojuegos según experiencia y cargo*

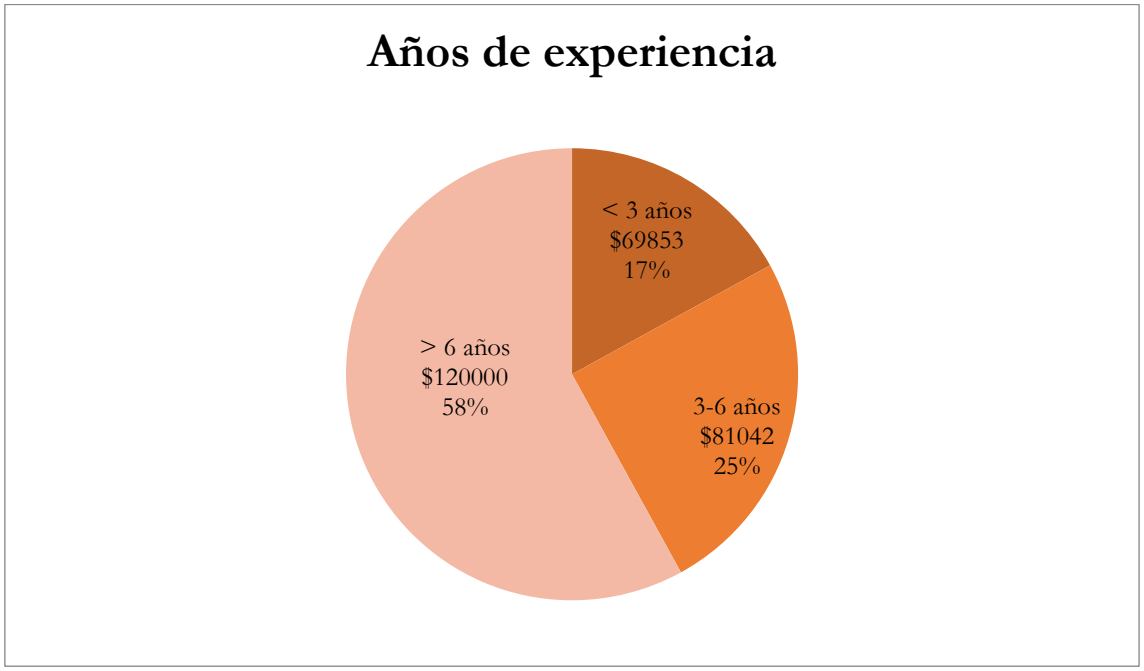


Figura 70: *% de trabajadores y años en la industria*

Tabla 8: *Tabla comparativa entre hombres y mujeres*

Sexo	Representación	Salario medio
Hombre	79%	$104,513
Mujer	21%	$90,250

Como se puede observar, la creación de un videojuego requiere de un equipo con un gran número de personas e implicados. Todos ellos tienen su especialidad y son necesarios a lo largo del desarrollo o una vez terminado.

Los datos nos dicen que en ciertas profesiones es difícil encontrar gente con poca experiencia, mientras que los que llevan más años suelen ser la mayoría de personas, con lo que se ve que este mercado lleva mucho tiempo en activo y sigue despegando.

Por otro lado, vemos que la gran mayoría de trabajadores del sector son hombres, parece ser que las mujeres no se interesan tanto aún por este sector, y en el caso de las pocas mujeres que hay, son discriminadas en cuanto a sueldo.

Parece ser que las posiciones de directivos a largo plazo son las que más dinero ganan, mientras que las de los programadores es la mejor pagada al iniciar.

A continuación se van a mostrar unos datos relevantes sobre la industria del videojuego en EE.UU y las disciplinas que más ganan y la gran diferencia que existe con respecto a los sueldos Europeos.

{AVERAGE SALARY} BY U.S. REGION

(across all levels of experience and disciplines)

West	$89,559
Midwest	$70,857
South	$74,838
East	$76,677

{TOP 10 STATES} WITH HIGHEST AVERAGE SALARIES

(across all levels of experience, excluding states with low sample size)

		AVERAGE SALARY	PERCENT WHO OWN HOMES	AVG. SALARY OF HOMEOWNERS
1	**California**	$91,602	29%	$112,301
2	**Washington**	$90,217	41%	$105,669
3	**North Carolina**	$88,214	48%	$98,000
4	**New York**	$82,738	17%	$108,214
5	**Nevada**	$79,286	36%	$106,000
6	**Texas**	$79,098	55%	$90,858
7	**Maryland**	$78,851	32%	$84,583
8	**Illinois**	$77,500	46%	$88,409
9	**Massachusetts**	$73,500	38%	$88,289
10	**Georgia**	$73,354	34%	$99,642

Figura 71: *Top 10 de salarios por región de EE.UU*

{AVERAGE SALARY}
BY U.S. REGION BY DISCIPLINE

	EAST	MIDWEST	SOUTH	WEST
Programming	$84,460	$76,538	$81,882	$99,851
Art and Animating	$72,236	$68,928	$64,705	$79,948
Game Design	$72,717	$64,318	$64,939	$77,906
Production	$78,571	$69,500	$84,464	$83,478
Audio	$58,500	$75,000	$67,500	$112,500
QA	$52,500	$60,833	$41,388	$58,310
Business/Mgmt.	$96,000	$66,785	$95,700	$109,681

Figura 72: *Comparación de los salarios por región y disciplina*

{AVERAGE SALARIES} U.S., CANADA, AND EUROPE

(across all levels of experience, by discipline, given in USD)

	U.S.	CANADA*	EUROPE**
OVERALL	**$83,060**	**$71,445**	**$46,232**
Programmer	$93,251	$81,660	$47,312
Art and Animation	$74,349	$62,663	$41,280
Game Design	$73,864	$60,434	$39,626
Production	$82,286	$68,000	$56,273
Audio	$95,682	$83,928	$66,666
QA	$54,833	$36,500	$27,500
Business	$101,572	$81,250	$57,500

Figura 73: *Comparación con los salarios Europeos*

Hay una gran diferencia salarial no solo a nivel europeo, sino entre los propios estados de un país.

Gracias a estos datos y los demás recopilados por la (E.S.A) se han podido estimar las contribuciones económicas del sector del videojuego en este último año, llegando a conseguir unos ingresos de más de 30.400 millones de dólares. Además en tan solo dos años se ha pasado de tener 1.800 compañías de videojuegos a más de 2.200, resultando esto en una tasa de crecimiento anual del 14% y en un aumento de empleados del 2,9%. Pronto esta industria será una de las más rentables del mundo, con un mercado muy provechoso, un mundo muy competitivo y unos trabajos muy demandados.

En cuanto al **sector en que se enfoca nuestra APP**, al ser un tipo de juego por niveles de corta duración que pone a prueba los reflejos del jugador, se ha destinado a personas de cualquier edad que tengan ratos libres y quieran un divertimiento corto. El juego se publicará de manera gratuita.

4.5 Pliego de condiciones

Para jugar con desenvoltura el videojuego se recomiendan los siguientes requisitos técnicos:

- Memoria Interna de 50 Mb.
- Memoria Ram de 1Gb.
- Estar en posesión de un dispositivo Android con Android 4.0 o superior.

Su descarga se podrá realizar mediante una cuenta de Play Store desde la siguiente dirección:

https://play.google.com/store/apps/details?id=com.GreatGamer.games.WildWestBlock

O buscando "**WildWestBlock**" en Play Store.

4.6 Análisis económico

Para comprobar si la realización de este proyecto puede convertirse en un negocio rentable, se ha elaborado un análisis de los costes de realizar dicha aplicación y formas de hacerla rentable.

4.6.1 Coste económico realización de la aplicación

Para calcular el costo económico nos basamos en el sueldo promedio de un desarrollador en España. Según informes del Presidente de la DEV (Desarrollo Español de Videojuegos), el salario medio de un desarrollador de videojuegos ronda los 30.000€ anuales. Con esta cifra sabemos que un desarrollador medio gana aproximadamente 13,4€/h.

Según la planificación inicial realizada el desarrollo del videojuego iba a costar un total de 105 horas. Este tiempo se ha visto incrementado hasta un total de 170 horas debido a:

- **Desconocimiento y aprendizaje de las herramientas a utilizar.** Aumento de 35 horas.
- **Problemas surgidos a la hora de realizar el desarrollo.** Aumento de 15 horas.
- **Ampliaciones del desarrollo inicial pensado**. Aumento de 15 horas.

Por lo tanto, si calculamos el costo de haber tenido un programador todo ese tiempo:

Coste de mano de obra: 170 Horas x 13,4€/h = **2.278€**

Ha esto hay que añadirle los gastos de Play Store que son de:

- Suscripción a Play Store = **25€ anuales**.

Y los gastos de luz consumidos en este tiempo por nuestro equipo (este cálculo usa datos oficiales del Ministerio de la Comunidad de Madrid como "precio medio de la electricidad", etc.):

- Precio Luz: 0,145€ / kWh
- Consumo de un equipo de gama Media: 180W/h
- Consumo monitor: 40W/h
- Consumo luz: 220W/h x 170h = 37.400 W = 37,40 kWh
- Coste Luz: 37,40 kWh * 0,145€/h = **5€ aproximadamente**

Vamos solo a reflejar el coste de los equipos necesarios, dado que ya los teníamos previamente:

- Ordenador sobremesa ASUS gama Media/alta: **800€**
- Móvil Android superior a 4.2 gama media: **150€**

Por tanto, el coste de realizar dicho proyecto ha sido de:

- Coste de mano de obra: = **2.278€**
- Suscripción a Play Store = **25€ anuales**.
- Coste Luz = **5€ aproximadamente**
- Ordenador sobremesa ASUS gama Media/alta: **800€**
- Móvil Android superior a 4.2 gama media: **150€**

COSTE TOTAL (sin equipos) = 2.308€

COSTE TOTAL (con equipos) = 3.258€

4.6.2 Rentabilización

Ahora que sabemos cuánto nos cuesta realizar este proyecto, vamos a ver las posibles formas de sacarle rentabilidad:

Aplicación de pago. La primera opción que tenemos para ganar dinero con nuestra aplicación es subirla al Play Store con un precio. En el mundo Android las aplicaciones de pago, por lo general, tienen muy mala fama. El cliente se ha acostumbrado a lo "free" y hacerles pagar por un videojuego o servicio no es visto con buenos ojos. Aun así, hay quien sigue sacando sus aplicaciones al mercado con un precio fijado.

Vamos a suponer que pusiésemos nuestra aplicación un coste inicial de 5€ y nuestra aplicación llegase a unas 1.000 personas que se la descargan. Nuestro beneficio inmediato sería de 5.000€

Publicidad. Por otro lado podemos establecer publicidad de terceros, tales como videos, o anuncios, de tal manera que recibamos ingresos por ello. Esta medida no es nada popular entre los usuarios, a nadie le gusta estar en pleno juego y que te salte un anuncio o video, suele crear más comentarios negativos que otra cosa, pero lo cierto es que la publicidad es el gran amigo de los desarrolladores indies (desarrolladores independientes). Y es que sin beneficio no hay apps. La gran verdad es que mientras los usuarios no estén dispuestos a aceptar que el desarrollo de apps requiere de muchos conocimientos técnicos, y además muy complejos, y que también requiere de un tiempo destinado a idear, diseñar, desarrollar y probar la app, tendrán que seguir existiendo apps con publicidad.

Vamos a situarnos en el mismo escenario. Unas 1.000 personas se descargan nuestra app gratuita y juegan diariamente viendo unos 10 videos al día cada uno y que 100 personas hacen clic en 1 anuncio al día. Sabiendo que el CPM (coste por miles) es de 2€ y que el CPC (coste por clic) es de 0,22€.

CPM = 10 videos x 1000 personas = 10.000 videos → 2€ *10 = 20€ diarios.

CPC = 100 clics *0,22€ = 22€ al día.

Si mantenemos la cifra en 1 año habremos obtenido: 42 * 365 = **15.330€**

Hay que aclarar que estas cifras son enormes, y mantener una comunidad tan activa durante tanto tiempo no es tarea fácil, y requiere de nuevos contenidos lo que supondría más gastos, etc.

Micro pagos. Por último, y la opción más de moda en la actualidad, incluir micro pagos dentro de la aplicación (no obligatorios) para conseguir mejoras, o avanzar más rápido en el desempeño del juego.

Suponiendo que 1.000 personas se descarguen la App, y 100 hagan un micro pago diario de 3€ para avanzar más rápido.

Dinero diario = 100 personas * 3€ = 300€ diarios

Si mantenemos la cifra en 1 año habremos obtenido: 300 * 365 = **109.500€**

Al igual que el anterior, son cifras extravagantes, no muchas compañías consiguen estos beneficios ni saben cómo mantenerlos en el tiempo.

4.6.3 Coste de la accesibilidad

Sabiendo que de las 170 horas realizadas de desarrollo, 80 son correspondientes a desarrollar todas las formas de accesibilidad, haciendo una regla de tres y calculando que las adaptaciones realizadas que este juego exigía no son en su mayoría costosas:

Coste accesibilidad Visual:

- Coste accesibilidad Daltonismo: 15 horas.
    - **COSTE TOTAL (sin equipos) = 203€**
    - **COSTE TOTAL (con equipos) = 287€**

- Coste accesibilidad Ceguera: 25 horas.
    - **COSTE TOTAL (sin equipos) = 339€**
    - **COSTE TOTAL (con equipos) = 479€**

Coste accesibilidad Auditiva: 15 horas.

- **COSTE TOTAL (sin equipos) = 203€**
- **COSTE TOTAL (con equipos) = 287€**

Coste accesibilidad Motora: 10 horas.

- **COSTE TOTAL (sin equipos) = 135€**
- **COSTE TOTAL (con equipos) = 191€**

Coste Multilenguaje: 15 horas.

- **COSTE TOTAL (sin equipos) = 203€**
- **COSTE TOTAL (con equipos) = 287€**

4.7 Manual de usuario

Para explicar el funcionamiento del juego y que se adapte a todos nuestros mecanismos de accesibilidad, se va a realizar en los diferentes lenguajes y teniendo en cuenta todas las capacidades.

Español

Este videojuego consta tal y como hemos visto de 6 pantallas diferentes. Una vez iniciado el juego y realizada la precarga de los datos necesarios, nos encontraremos en el menú principal. En este menú podemos elegir la opción de "**salir**" del juego o de "**silenciar**" todos los sonidos. En el mismo tenemos dos botones, "**Jugar**" y "**Opciones**".

Si elegimos "**Opciones**", se nos abrirá la ventana de las perfiles de accesibilidad donde podremos escoger, pulsando encima de una de ellas, cual queremos. Al pulsar sobre una de ellas un cuadrado con la forma de un enemigo se situará indicando la seleccionada. En caso de elegir "**Daltonismo**", la aplicación cambiará inmediatamente a una escala de grises. En caso de elegir "**D. Auditiva**", el juego tendrá opciones de vibración y onomatopeyas que lo harán más interesante. En caso de elegir "**D. Motora**", las zonas de contacto de los botones se ampliarán de tal forma que resulte mucho más sencillo pulsarlos. En caso de escoger "**D. Visual**" se ampliarán igual las zonas de contacto y se activará un mecanismo de vibración y voz que nos dirá cuando y donde debemos pulsar para destruir a nuestros enemigos. El **lenguaje** es lo único compatible con cualquier otra accesibilidad. Podemos escoger si queremos el texto en inglés o en español.

Si elegimos "**Jugar**", pasará a otra pantalla donde seleccionaremos el nivel a jugar. Se recomienda empezar por el tutorial para familiarizarse con el juego y sus elementos. Una vez seleccionado comienza el nivel. Dependiendo de en cual se esté y de las opciones accesibles seleccionadas, tendrá unas características u otras. El objetivo es destruir todos los enemigos antes de que caigan al suelo. No debes destruir las bombas, ni quedarte sin vidas o balas o perderás. A veces caerán Power Ups que nos ayudarán a conseguir más balas o vidas.

Por último, en caso de **superar el nivel** nos encontraremos con una escena que nos felicita, nos muestra nuestras estadísticas del nivel y nos da a elegir entre volver al menú o continuar al siguiente nivel.

En caso de **no superar el nivel**, aparece otra pantalla de características similares a la anterior, pero que nos dice que hemos perdido y nos da la opción de reintentarlo.

Los niveles se irán desbloqueando según vayamos superando los anteriores y el objetivo actual es conseguir superar los 20 niveles que existen.

English

As we have already seen, this videogame consists of 6 different screens. Once the game is started and the necessary data is preloaded, we will find ourselves in the main menu. In this menu, we can choose "**exit**" to quit the App, or "**silence**" the game. It shows two buttons, "**Play**" and "**Options"**.

On the one hand, if we choose "**Options**", it takes us to a new window where we can see to which disabilities is adapted the videogame. We can choose any of them by clicking on its name. When we select one, the picture of an enemy appears next to the name, confirming that it has been picked. In case we choose "**Daltonism**", the application will immediately change to a grayscale. If we select "**Deafness**", the game will activate vibration and onomatopoeias that will make it more interesting. In case of picking "**Paralysis**", the contact zones of the buttons will be enlarged in such a way that it is easier to press them. If we choose "**Blindness**", the contact zones will be also expanded, and a mechanism of vibration and voice will be activated. It will tell us when and where we must press in order to destroy our enemies. **Language** is the only thing compatible with any other accessibility. We can choose to have the text in English or Spanish.

On the other hand, if we choose "**Play**", it will lead us to another screen where we will select the level to play. It is recommended to start with the tutorial in order to become familiar with the game and its elements. Once a level has been selected, the game begins. According to the level and accessibilities picked, the game will have some characteristics or others. The main objective is to destroy all the enemies before they touch the ground. If you destroy a bomb or you run out of lives or bullets, you will lose. Every now and then, Power Ups will fall in order to provide us with more bullets or lives.

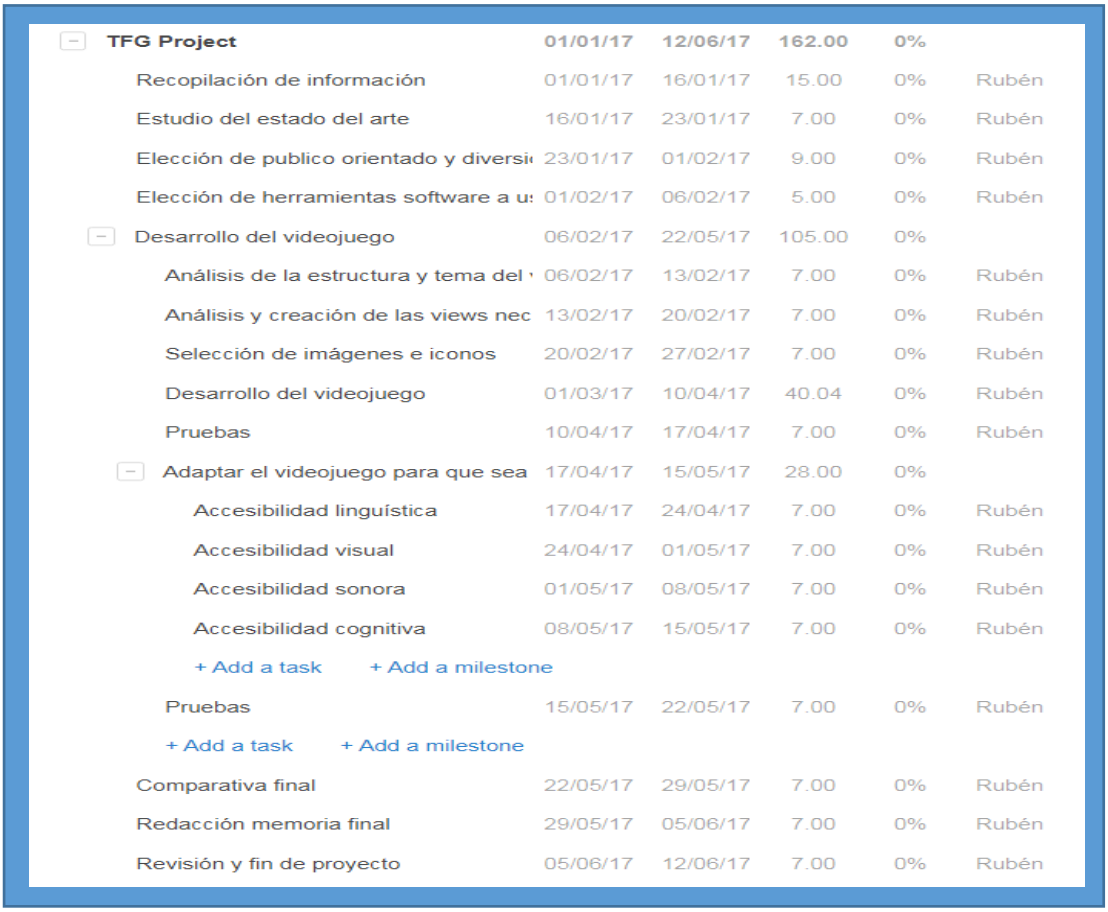
Finally, in case of **exceeding the level**, a congratulation screen will appear, show us our statistics of the level, and make us choose between returning to the menu and continuing to the next level.

In case of **not exceeding the level**, a different screen appears, with similar characteristics to the previous one, but it says that we have lost and it gives us the opportunity to retry it.

Each level will be automatically unlocked by the time we overcome the previous one. The current goal is to exceed the 20 levels that exist.

5 Resumen

Para la realización de dicho proyecto, se va a definir una metodología de trabajo y una planificación inicial que consta de las siguientes partes:



TFG Project	01/01/17	12/06/17	162.00	0%	
Recopilación de información	01/01/17	16/01/17	15.00	0%	Rubén
Estudio del estado del arte	16/01/17	23/01/17	7.00	0%	Rubén
Elección de publico orientado y diversi	23/01/17	01/02/17	9.00	0%	Rubén
Elección de herramientas software a u	01/02/17	06/02/17	5.00	0%	Rubén
Desarrollo del videojuego	06/02/17	22/05/17	105.00	0%	
Análisis de la estructura y tema del	06/02/17	13/02/17	7.00	0%	Rubén
Análisis y creación de las views nec	13/02/17	20/02/17	7.00	0%	Rubén
Selección de imágenes e iconos	20/02/17	27/02/17	7.00	0%	Rubén
Desarrollo del videojuego	01/03/17	10/04/17	40.04	0%	Rubén
Pruebas	10/04/17	17/04/17	7.00	0%	Rubén
Adaptar el videojuego para que sea	17/04/17	15/05/17	28.00	0%	
Accesibilidad linguística	17/04/17	24/04/17	7.00	0%	Rubén
Accesibilidad visual	24/04/17	01/05/17	7.00	0%	Rubén
Accesibilidad sonora	01/05/17	08/05/17	7.00	0%	Rubén
Accesibilidad cognitiva	08/05/17	15/05/17	7.00	0%	Rubén
+ Add a task + Add a milestone					
Pruebas	15/05/17	22/05/17	7.00	0%	Rubén
+ Add a task + Add a milestone					
Comparativa final	22/05/17	29/05/17	7.00	0%	Rubén
Redacción memoria final	29/05/17	05/06/17	7.00	0%	Rubén
Revisión y fin de proyecto	05/06/17	12/06/17	7.00	0%	Rubén

Figura 74: *Planificación Gantt*

Debido a la falta de experiencia en este campo, primero se hizo una recopilación de información general sobre tipos de accesibilidad, qué queremos y qué se puede lograr, cuánto esfuerzo nos conlleva dicho trabajo, etc. Se ha podido comprobar que hay una enorme cantidad de discapacidades distintas y cada una puede tener una solución diferente a la hora de ser adaptada a un juego. Por ello se procedió a realizar una selección amplia de las discapacidades que se querían tratar, intentando abarcar un gran abanico.

Una vez hecha esa primera toma de contacto, se realizó un estudio sobre el estado actual del mercado. Dicho estudio describía las distintas formas de rentabilizar una aplicación, los tipos de juegos más rentables y los más jugados, con el fin de definir qué tipo de juego se pretendía realizar. La información sobre los juegos más jugados unido a los distintos mecanismos de accesibilidad que se iban a adaptar, dio como resultado la creación de un tipo de juego de bloques y niveles, donde era posible tener en cuenta este gran abanico que se quería implementar.

Definido nuestro tipo de juego, cabía definir hacia quien lo íbamos a orientar. Al ser un juego de bloques por niveles de corta duración era ideal para aquellas personas con pequeños ratos libres en los que aprovechar para divertirse un poco.

El siguiente paso era ver que herramientas existían en el mercado y comparando ventajas y desventajas poder decidir, cuál era la que nos daba la capacidad de realizar todo lo que se pretendía. Unity 3D fue la herramienta elegida que cumplió las expectativas.

Se tuvo que aprender a usar la herramienta y su lenguaje para poder desarrollar el juego. Resultó difícil en el inicio, pero se fueron definiendo las estructuras que se iban a crear, las distintas pantallas que iba a tener el videojuego, los tipos de enemigos, con sus mecánicas, y la funcionalidad de cada uno de ellos, se fueron creando las imágenes y diseños y se aprendió a insertar animaciones vía Sprites.

Una vez se dio por terminado la creación del videojuego y con los perfiles de discapacidad ya pensadas, se creó una nueva pantalla con dichas opciones y se empezó a ver cómo adaptar el juego a las mismas. Aquí surgió uno de los grandes inconvenientes, dado que adaptar el juego no era lo complicado en sí, lo complicado era ver a qué estabas dispuesto a renunciar si lo adaptabas. De esta forma igual al adaptarlo para gente invidente perdíamos gran cantidad de matices y funcionalidades, lo que convertía al juego en algo aburrido. ¿Merecía la pena adaptar el juego si sacrificábamos su jugabilidad? Ver y encontrar esa línea, fue la siguiente tarea a realizar.

Una vez establecidos los límites de lo que queríamos se procedió a su desarrollo, con ciertos retrasos en los tiempos debido al desconocimiento de los nuevos métodos que se debían utilizar.

Periódicamente durante la evolución del videojuego se fue prestando la APK para la realización de pruebas a diversas personas, pero cada vez que se finalizaba una etapa más grande, se realizaba una batería de pruebas más exhaustivas donde salían a relucir posibles fallos.

Por último, una vez corregidos todos los fallos encontrados, se procedió a subir la aplicación al Play Store, desde donde se puede descargar.

6 Conclusión

Si nos fijamos en los datos económicos establecidos en los apartados anteriores, parece obvio que la aplicación es rentable hagamos lo que hagamos. Pero en realidad, dado que los datos están hechos a la alza, y que lo que nos interesa realmente no es ver si hacer una APP es rentable, sino si sale rentable su desarrollo accesible, y basándonos en los datos proporcionados por el IMSERSO y el Gobierno de España, que dicen que el número de gente con alguna discapacidad es de 2,5 millones (en España) frente a los 46,56 millones de habitantes que hay (tan solo un 5,37% de la población), podemos deducir, y siguiendo el ejemplo propuesto anteriormente que:

- Si 1000 personas iban a descargar nuestra aplicación (a nivel nacional), tan solo 53 personas con discapacidad iban a bajársela. Lo que iba a significar unos ingresos, en cualquier caso, menores que los costes de realizar dicha accesibilidad.

  1000 ---- 46.560.000 | x= 53 personas con discapacidad descargarían la aplicación

  X --------- 2.500.000 |

- Las empresas no invierten en estas tecnologías debido a que el sector es demasiado pequeño como para sacar un provecho económico que les interese y que sea sostenible. Hace un par de años, cosa que está animando a las empresas a investigar y desarrollar más este sector, se vienen dando unas subvenciones a las empresas que trabajan con estas tecnologías ya que se categorizan como "labor social".

- La complejidad del desarrollo de estas aplicaciones, y la necesidad obligada de renunciar a funcionalidades de una aplicación para volverla accesible, hacen que muchos desarrolladores no crean conveniente adaptar sus aplicaciones y perder la esencia inicial de un juego.

- La falta de conocimiento, manuales y herramientas del desarrollo de estos mecanismos de accesibilidad hace que resulte más costoso la realización de dichas adaptaciones.

Personalmente creo que es necesario e interesante que se desarrollen aplicaciones con estas características, que lleguen a todo el mundo y permitan, faciliten o entretengan a quien desee usarla. Pero en lo que a negocio se refiere, mientras los gobiernos no impongan ayudas o leyes obligando a las empresas a realizar dichos desarrollos, o la tecnología avance lo suficiente como para abaratar tanto los costes de estos desarrollos que merezca la pena realizarlos, las empresas y desarrolladores en su gran mayoría optarán por un desarrollo destinado al grupo mayoritario de este mercado.

7 Futuras mejoras

En un futuro, se podrían realizar las siguientes mejoras:

Ampliación de niveles. Actualmente se cuenta con 20 niveles diferentes que dejan probar y disfrutar del juego y hacen conocer la mecánica del mismo. Estos niveles son pocos si queremos competir en un mundo como el de los videojuegos, además, deberían seguir actualizándose y creándose más constantemente para no cansar a jugadores y atraer más su atención.

Ampliación de enemigos. Con solo dos mecánicas de enemigos, un juego puede resultar repetitivo y falto de sorpresa. Para ello se ha pensado aumentar la cantidad de tipos de enemigos así como sus mecánicas en el juego. Por ejemplo, cada 5 niveles se metería un "**Bloque jefe**" el cual tiene varias vidas y ha de ser golpeado varias veces antes de morir.

Ampliación de personajes. Como ya se ha hablado el aspecto visual es muy importante, por lo que dar la opción al usuario de personalizar esta le da al juego un toque más atractivo. De esta forma, el usuario puede ir desbloqueando ciertos personajes e ir usándolos en combate en vez del pistolero.

Figura 75: *Personaje 1*

Figura 76: *Personaje 2*

Figura 77: *Personaje 3*

Ampliación de idiomas. Como uno de los requisitos accesibles del juego, el idioma es fundamental para llegar a un público aún mayor. Por eso expandir el número de idiomas a tantos como sea posible siempre es garantía de éxito.

Ampliación de texturas. Al igual que pasa con los personajes, las texturas del videojuego (fondos, pantallas, etc.) actualmente son muy estáticas, es por esto que ampliar ese catálogo o permitir al usuario modificarlo a su antojo va a repercutir de forma visualmente positiva.

Ampliación de efectos visuales y sonoros. Los efectos de sonido y visuales son lo que hacen de un juego lo que es, por eso se ha pensado en ampliar estos campos de tal manera que el videojuego entre instantáneamente por los sentidos. Ampliar el abanico de efectos visuales y modificarlos acorde los nuevos personajes y enemigos es una forma fácil de ganar margen en este aspecto.

Ampliación objetos (Power Ups). Para amenizar la mecánica del juego y evitar un juego repetitivo y falto de emoción, en los niveles se ha pensado insertar una serie de "Power Up" tales como:

- **Congelador.** Otro de los posibles Power Up, es un objeto que haga disminuir drásticamente la velocidad de caída de los bloques durante un periodo de tiempo, facilitando así su destrucción.

Figura 78: *Power Up congelante*

Método competición y redes sociales. Introducir un sistema de puntuación online, permitir que varios jugadores compitan, o que puedan compartir sus logros por las redes sociales hace que un juego enganche más por ese plus de competitividad que tiene. Por ello, permitir e incorporar un método de clasificación es otra de las mejoras que se podrían realizar.

Añadir sistema de vidas. Actualmente puedes jugar una y otra vez sin cesar los niveles. Introducir un sistema de vidas con un contador que vaya regenerando dichas vidas para poder jugar ayudaría a introducir métodos de pago y a hacer que el juego durase más y enganchara.

Ampliación de armas. Por último, como posible mejora, se ha pensado en ampliar el catálogo de armas (actualmente solo una) y permitir al jugador ir desbloqueando y eligiendo que arma prefiere usar, así como ir actualizando la munición y animaciones que posea.

Añadir el reconocimiento de voz. Se quiere añadir un evento touch en la pantalla que active el reconocimiento de voz para las discapacidades de "D Visual" y "D. Motora". De esta forma, en el menú principal y el de elegir nivel, al pulsar la pantalla y hablar nos reconocerá la voz y en caso de decir "Play" o "Level X" según el menú en el que estemos, realizará dicha acción y cambiará de pantalla sin necesidad de pulsar nada.

8 Bibliografía

Aguado Delgado, J. (Septiembre de 2014). APORTES METODOLÓGICOS PARA LA CREACIÓN DE VIDEOJUEGOS ACCESIBLES.

Android developer. (s.f.). *Desarrolladores android*. Obtenido de https://developer.android.com/studio/intro/index.html

DEV. (s.f.). *desarrolladores españoles de videojuegos*. Obtenido de http://www.dev.org.es/

E.S.A. (s.f.). *Entertainment Software Association*. Obtenido de Entertainment Software Association: http://www.theesa.com

Herbert. (1990). *Observación Activa.*

Microsoft. (s.f.). *Microsoft*. Obtenido de https://msdn.microsoft.com/es-es/library/ms233836(v=vs.90).aspx

Unity Technologies. (s.f.). *Unity 3D*. Obtenido de Unity 3D: https://unity3d.com/es/unity

Wikipedia LibGDX. (s.f.). Obtenido de https://es.wikipedia.org/wiki/LibGDX