

Proyecto Fin de Carrera

Ingeniería en Informática

Curso de programación de videojuegos en Unity 3D para iPad

Autor:

Cristian Escudero Cestero

Director:

Ramón Piedrafita Moreno

Área de Ingeniería de Sistemas y Automática

Departamento de Informática e Ingeniería de Sistemas

Escuela de Ingeniería y Arquitectura
Universidad de Zaragoza

Septiembre 2013

Curso de programación de videojuegos en Unity 3D para iPad

Resumen

El curso de programación de videojuegos en Unity 3D para iPad es un PFC que consiste en una aplicación iOS interactiva diseñada y creada para el dispositivo iPad de Apple. Con esta aplicación se pretende facilitar y estimular la enseñanza de la programación de videojuegos.

Unity es un motor gráfico multiplataforma que se distribuye junto a un IDE (Entorno de Desarrollo Integrado) que permite la creación de contenido 3D (en especial videojuegos) de forma más rápida y sencilla que la mayoría de entornos existentes.

La aplicación tiene el objetivo de introducir a los usuarios dentro del mundo del desarrollo de videojuegos mediante el uso de Unity 3D. Para ello se ha creado una amplia documentación que cubre desde los aspectos más básicos hasta la creación de un videojuego completo.

Esta aplicación está compuesta por las siguientes partes:

- ▶ **Visor:** es la parte de la aplicación que centraliza todo el curso. Desde el visor se accede al resto de elementos. Se han programado las funcionalidades típicas de los visores de documentos (zoom, desplazamiento táctil, acceso a los contenidos y a la ayuda), así como el lanzamiento del reproductor de vídeos, ejecución de las aplicaciones y del juego completo. También muestra la documentación del curso.
- ▶ **Documentación:** conjunto de temas escritos que explican la programación de videojuegos con Unity 3D.
- ▶ **Vídeos:** ejemplos visuales que resumen partes de las explicaciones escritas más relevantes. Los vídeos se reproducen en la misma aplicación.
- ▶ **Aplicaciones:** ejemplos interactivos creados con Unity 3D, que se ejecutan desde el visor y se manejan dentro de la aplicación principal.
- ▶ **Juego:** ejemplo completo de videojuego. Es una parte fundamental puesto que es una aplicación en sí misma ya que posee los elementos tradicionales de un videojuego. El juego se comunica con una aplicación web de gestión de máximas puntuaciones.

La aplicación está diseñada para que su manejo sea intuitivo y sencillo y que pueda ser usada por cualquier persona sin la necesidad de tener conocimientos avanzados en este tipo de dispositivos. Para el seguimiento del curso se recomienda tener mínimos conocimientos de programación.

Índice

Índice de contenido

1. Introducción	14
1.1. Contexto y motivación del proyecto	15
1.2. Objetivos y alcance	17
1.3. Estudios preliminares	19
1.4. Tecnología. Uso y justificación	19
1.5. Ámbito del proyecto	21
1.6. Metodología y planificación	22
1.7. Estructura del documento	23
2. Trabajo realizado	25
2.1. Visor	27
2.1.1. Introducción	27
2.1.2. Estudio preliminar	27
2.1.3. Determinación de requisitos	28
2.1.4. Casos de uso	29
2.1.5. Decisiones de diseño	31
2.1.6. Diseño de la interfaz	33
2.1.7. Implementación del visor	37
2.2. Videjuego “La venganza de Timan”	38
2.2.1. Introducción	38
2.2.2. Análisis y diseño	38
2.2.3. Modelado de objetos en 3D	50
2.2.4. Implementación	53
2.2.5. Interfaz	56
2.3. Aplicación web	58
2.3.1. Introducción	58
2.3.2. Decisiones de diseño	59
2.3.3. Web de puntuaciones	62
2.4. Resto de elementos de la aplicación	65
2.4.1. Textos del curso	65
2.4.2. Vídeos	65
2.4.3. Aplicaciones interactivas	66

2.5. Pruebas	68
3. Conclusiones	70
3.1. Cumplimiento de objetivos	70
3.2. Mejoras y ampliaciones	71
3.3. Perspectivas de futuro del proyecto	71
3.4. Valoración personal	72
4. Bibliografía	74
Anexos	
Anexo A. Estudios preliminares	77
A.1. Situación de la industria de los videojuegos	77
A.2. Estudio de mercado de los dispositivos iOS	85
A.3. Importancia de Unity 3D como motor gráfico	90
A.4. Conclusiones	90
Anexo B. Tecnologías empleadas	92
B.1. Pages	92
B.2. Keynote	93
B.3. Numbers	93
B.4. GIMP	94
B.5. InkScape	97
Anexo C. Modelado de objetos con Blender	100
Anexo D. Unity 3D	103
D.1. UnityScript	104
D.1.1. Generalidades	104
D.1.2. Sintaxis	106
D.2. C#	109
D.2.1. Características de C#	110
Anexo E. Tecnologías web empleadas	115
E.1. AJAX	115
E.2. PHP	117
E.3. MySQL	119
Anexo F. Documentación del proyecto	121
F.1. Estudio preliminar de visores	121

F.2. Especificación de requisitos	123
F.3. Planificación	128
F.3.1. Planificación inicial	128
F.3.2. Planificación real	130
F.4. Análisis y diseño del videojuego	134
F.4.1. Casos de uso	134
F.4.2. Diagrama de flujo del videojuego	135
F.4.3. Diagrama de clases	139
F.4.4. Modelado dinámico	147
F.4.5. Interfaz del videojuego	155
F.5. Pruebas	156
F.6. Vídeos del curso	158
F.7. Aplicaciones interactivas	160
Anexo G. Manual de usuario	163
G.1. Perspectiva general de la aplicación	163
G.2. Visor	164
G.3. Videojuego “La venganza de Timan”	174
G.3.1. Objetivos del juego	176
G.3.2. Control del personaje	177
G.3.3. Puntuaciones	178
G.3.4. Interfaz de la partida	179
Anexo H. Documento de Diseño del Juego (GDD)	180
H.1. Modelo de DGG	181
H.2. Modelo de TDD	183
Anexo I. Ejemplo de capítulo del curso	185

Índice de figuras

Figura 1. Ingresos a nivel mundial logrados por la industria de los videojuegos	15
Figura 2. Motores gráficos empleados por los desarrolladores de videojuegos para móviles	16
Figura 3. Diagrama del modelo en cascada modificado	22
Figura 4. Visión general del sistema creado	25
Figura 5. Caso de uso general del visor	30
Figura 6. Diagrama de actividades del visor	32
Figura 7. Jerarquía de ventanas del visor	33
Figura 8. Aspecto de la interfaz del curso con una página de texto cargada	34
Figura 9. Aspecto de la interfaz con el menú de opciones desplegado	35
Figura 10. Aspecto de la interfaz con el menú de vídeos cargado	36
Figura 11. Diagrama del patrón de diseño Modelo - Vista - Controlador	37
Figura 12. Diagrama de secuencia del caso de uso avanzar página	38
Figura 13. Diagrama de flujo de datos de nivel 0 del juego	41
Figura 14. Elementos básicos de un videojuego	42
Figura 15. Ejemplo de interacción con el personaje y enemigos en el videojuego	43
Figura 16. Ejemplo de los enemigos del personaje principal	44
Figura 17. Ejemplo de superficie mortal	44
Figura 18. Ejemplo de estrella y cofre del videojuego	45
Figura 19. Ejemplo de checkpoint	45
Figura 20. Ejemplo de Corazón que representan las vidas del personaje	46
Figura 21. Esquema del primer nivel del videojuego	47

Figura 22. Captura de pantalla real del primer nivel del videojuego	47
Figura 23. Esquema del segundo nivel del videojuego	48
Figura 24. Captura de pantalla real del segundo nivel del videojuego	48
Figura 25. Esquema del tercer nivel del videojuego	49
Figura 26. Captura de pantalla real del tercer nivel del videojuego	50
Figura 27. Modelo del personaje principal para uno de los niveles del juego	51
Figura 28. Modelo de lava con textura animada	51
Figura 29. Modelo de foso con pinchos	51
Figura 30. Modelos de las plataformas empleadas en los tres niveles	52
Figura 31. Modelo de rayo creado para el nivel 3	52
Figura 32. Modelos de objetos empleados como decoración de los niveles	53
Figura 33. Diagrama de las clases empleadas en el personaje principal del videojuego	54
Figura 34. Diagrama de secuencia de la interacción entre enemigos y personaje principal	55
Figura 35. Diagrama de navegación del menú principal	56
Figura 36. Ejemplo de interfaz del menú principal	57
Figura 37. Interfaz del juego	58
Figura 38. Diagrama Entidad / Relación de la aplicación web	60
Figura 39. Diagrama de actividades del Juego - Aplicación web	62
Figura 40. Captura de pantalla de la página de resultados máximos	63
Figura 41. Captura de pantalla de resultados máximos ordenados por el total de estrellas	64
Figura 42. Hola Mundo 3D. Ejemplo de aplicación interactiva	67
Figura 43. Ejemplo de aplicación de fuerzas sobre objetos	68
Figura A.1. Ingresos globales en móviles (1010 dólares) y participación del total del segmento de videojuegos (%) de 2008 a 2017	78

Figura A.2. Ingresos por venta de consolas por zonas (1010 dólares) de 2011 a 2017	79
Figura A.3. Ingresos por venta de videojuegos en los tres mercados más importantes (1010 dólares) de 2008 a 2017	80
Figura A.4. Gasto global de los usuarios por tipo de dispositivo (1010 dólares) de 2008 a 2017	81
Figura A.5. Total de ingresos por juegos en línea (1010 dólares) y participación del total del segmento de videojuegos (%) de 2008 a 2017	82
Figura A.6. Ingresos totales (109 dólares) de la industria de los videojuegos en EE.UU. (2012)	82
Figura A.7. Videojuegos más vendidos por género en consolas en unidades (2012)	84
Figura A.8. Videojuegos más vendidos por género en computadoras en unidades (2012)	85
Figura A.9. Distribución de nuevos proyectos en iOS Vs Android	86
Figura A.10. Dispositivos Android activos como porcentaje de dispositivos iOS	86
Figura A.11. Total de tiempo pasado en aplicaciones por Android como porcentaje del total de tiempo en aplicaciones por iOS	87
Figura A.12. Países con el mayor número de dispositivos iOS y Android activos (millones) (2013)	88
Figura A.13. Millones de ventas de iPad (2010-2013)	89
Figura A.14. Tiempo empleado por categoría de aplicación	91
Figura B.1. Icono de la aplicación Pages	92
Figura B.2. Icono de la aplicación Keynote	93
Figura B.3. Icono de la aplicación Numbers	93
Figura B.4. Icono de la aplicación GIMP	94
Figura B.5. Modelo 3D del enemigo Cosa con textura aplicada	95
Figura B.6. Modelo 3D del enemigo Cosa sin textura	95
Figura B.7. Modelo 3D del objeto con las aristas marcadas en rojo	95
Figura B.8. Mapa UV en formato png importado con GIMP	96
Figura B.9. Textura creada a partir del mapa UV	97

Figura B.10. Logotipo de la aplicación InkScape	98
Figura B.11. Imagen del primer nivel del videojuego. Fondo y nubes creados con InkScape	99
Figura C.1. Logotipo de la aplicación Blender	100
Figura C.2. Proceso de creación del personaje principal y de algunos enemigos	101
Figura C.3. Modelo de plantilla para la creación de los modelos de un videojuego	102
Figura C.4. Ejemplo de boceto de Timan	102
Figura E.1. Logotipo de AJAX	115
Figura E.2. Tecnologías que forman parte de AJAX	116
Figura E.3. Logotipo de MySQL	117
Figura E.4. Logotipo de MySQL	119
Figura F.1. Gráfica con el tiempo estimado en porcentaje	129
Figura F.2. Gráfica con el tiempo real en porcentaje	130
Figura F.3a. Diagrama de Gantt del tiempo estimado y real del proyecto	132
Figura F.3b. Diagrama de Gantt del tiempo estimado y real del proyecto	133
Figura F.4. Casos de uso del videojuego	134
Figura F.5. Diagrama de flujo de datos de nivel 0 del juego	135
Figura F.6. Diagrama de flujo de datos de nivel 1 del juego	137
Figura F.7. Diagrama de flujo de datos de nivel 2. Especificación del proceso 1: Tratar usuario	138
Figura F.8. Diagrama de flujo de datos de nivel 2. Especificación del proceso 2: Jugar partida	139
Figura F.9. Diagrama de clases principales del videojuego	139
Figura F.10. Diagrama de clases del personaje principal	140
Figura F.11. Diagrama de clases de los enemigos	143
Figura F.12. Diagrama de clases de las plataformas	144
Figura F.13. Diagrama de clases de las superficies mortales	145

Figura F.14. Diagrama de estados del movimiento del personaje	148
Figura F.15. Diagrama de secuencia de la interacción entre enemigos y personaje principal que provoca la muerte de un enemigo	149
Figura F.16. Diagrama de secuencia de la interacción entre enemigos y personaje principal que provoca la pérdida de una vida del personaje	150
Figura F.17. Diagrama de secuencia que describe la situación provocada por la entrada en contacto de una superficie mortal en su desplazamiento con el personaje principal	151
Figura F.18. Diagrama de secuencia que describe la interacción entre el personaje y una estrella	151
Figura F.19. Diagrama de secuencia que describe la interacción entre el personaje y corazón que recupera una vida del personaje	152
Figura F.20. Diagrama de secuencia que describe la interacción entre el personaje y un punto de guardado	153
Figura F.21. Diagrama de actividades de FinNivel	154
Figura F.22. Jerarquía de ventanas del videojuego	155
Figura F.23. Índice de vídeos del curso	158
Figura F.24. Índice de aplicaciones del curso	161
Figura G.1. Avance y retroceso de página mediante pulsaciones en la pantalla	164
Figura G.2. Avance y retroceso de página por deslizamiento	165
Figura G.3. Activación y desactivación del zoom	166
Figura G.4. Desplazamiento de pantalla con zoom activado	166
Figura G.5. Ejemplo de pantalla con las opciones sin desplegar	167
Figura G.6. Ejemplo de pantalla con las opciones desplegadas	168
Figura G.7. Índice de aplicaciones interactivas	169
Figura G.8. Índice de vídeos del curso	170
Figura G.9. Índice de temas escritos del curso	170
Figura G.10. Ayuda del curso cargada desde el menú de opciones	171

Figura G.11. Botones ‘Play Video’ y ‘Play Game’ que permiten reproducir desde la misma hoja el vídeo o la aplicación interactiva relacionada	172
Figura G.12. Ejemplo del reproductor de vídeos	173
Figura G.13. Ejemplo de aplicación interactiva con el botón de vuelta al curso	173
Figura G.14. Formulario de creación de nuevo usuario	174
Figura G.15. Menú principal del juego	175
Figura G.16. Checkpoint del juego	176
Figura G.17. Resumen de los movimientos del personaje principal	177
Figura G.18. Resumen de los saltos del personaje principal	177
Figura G.19. Elementos del juego	178
Figura G.20. Interfaz del juego	179

Índice de tablas

Tabla 1. Resumen de puntuaciones del videojuego	46
Tabla A.1. Millones de ventas de las tabletas de distintos fabricantes (2012-2013)	89
Tabla F.1. Requisitos funcionales de la aplicación	123
Tabla F.2. Requisitos no funcionales de la aplicación	127
Tabla F.3. Planificación inicial	128
Tabla F.4. Tiempo real empleado	130
Tabla G.1. Tabla de puntuaciones del juego	178

1. Introducción

Este Proyecto Fin de Carrera (PFC) titulado “Curso de programación de videojuegos en Unity 3D para iPad”, de la titulación de Ingeniería en Informática impartida en la Escuela de Ingeniería y Arquitectura de la Universidad de Zaragoza recoge el trabajo realizado por el alumno Cristian Escudero Cestero durante el periodo comprendido entre los meses de octubre de 2012 y julio de 2013.

El proyecto ha sido dirigido y supervisado por Ramón Piedrafita Moreno del Departamento de Informática e Ingeniería de Sistemas de la Universidad de Zaragoza.

El trabajo consiste en la creación de una aplicación para dispositivos iPad de Apple que permita la enseñanza del proceso de creación y desarrollo de un videojuego haciendo uso de un motor gráfico. Para su creación ha sido necesario evaluar las características y posibilidades del dispositivo y del propio motor gráfico así como ajustar la dificultad del curso de tal forma que una persona que nunca haya programado un videojuego sea capaz de desarrollar uno. También ha sido necesario crear una Interfaz Gráfica de Usuario (GUI) desde la que se acceda a todos los contenidos del curso, la documentación del mismo, las aplicaciones de ejemplo, los vídeos de ejemplo y el videojuego completo. Por último se ha necesitado crear una base de datos y una aplicación web para el intercambio de datos durante la creación y modificación de un usuario y para la gestión de las puntuaciones máximas de las partidas del videojuego completo.

El proyecto es novedoso puesto que mediante una aplicación intenta formar al alumno en la creación de un videojuego y mostrar el proceso y la complejidad que supone crear uno desde cero. Al crear un videojuego desde cero se pasa por todas las fases de creación de un proyecto informático (determinación de requisitos, análisis, diseño, implementación y pruebas) pero se añaden aspectos que involucran a otras disciplinas y que suponen un reto para el creador de un videojuego, como el diseño y modelado de los personajes, niveles y ambientes, la creación de texturas, elección de la música y programación de la lógica del juego entre otros. También resulta novedoso debido a la escasa información organizada existente sobre Unity 3D en especial en castellano.

1.1. Contexto y motivación del proyecto

La industria de los videojuegos ha sufrido una gran transformación en los últimos años. Han quedado atrás los tiempos en los que la tecnología usada en los videojuegos era baja, las consolas eran de una calidad inferior y en el que los usuarios jugaban de forma individual. En los últimos años se ha convertido en la tercera industria del entretenimiento con 10.500 millones de dólares de ingresos, muy cerca del cine con 10.600 millones y a poca distancia de la música con 11.600 millones (según datos de ventas en EE.UU. en 2012).

En la figura 1 se muestra la evolución en ventas experimentada por la industria de los videojuegos. En ella se aprecia como los ingresos se incrementan año tras año (a excepción del año 2003 que se produjo un descenso respecto al año anterior) hasta alcanzar los 78.000 millones de dólares en 2012. Consultoras internacionales como *Pricewaterhouse Coopers* estiman que el crecimiento continuará en los años sucesivos alcanzando los 82.000 millones de dólares en 2017.

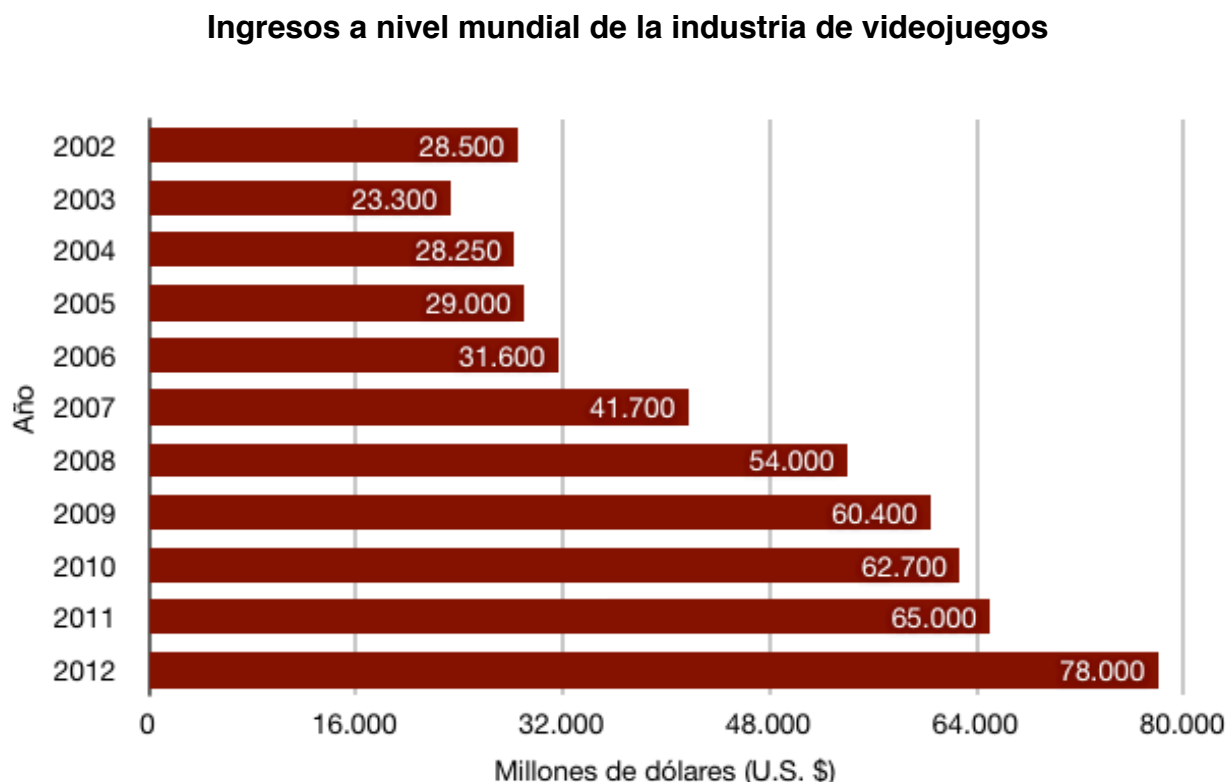


Figura 1. Ingresos a nivel mundial logrados por la industria de los videojuegos

En términos económicos la creación de videojuegos es una actividad que genera volumen de negocio y a grandes rasgos queda justificada su importancia. La importancia económica se explica con más detalle en el estudio de mercado incluido en el anexo A.1.

Los videojuegos no son solo para jugar, también proporcionan puestos de trabajo. Por ejemplo en 2009 se crearon 32.000 puestos de trabajo como desarrollador de videojuegos en los Estados Unidos. El salario medio anual se encuentra entre los 52.00 y 60.000 dólares y el crecimiento en los últimos diez años ha sido del 32,4% en creación de puestos de trabajo. La opción laboral puede ser otra motivación para especializarse en el desarrollo de videojuegos.

Desde el punto de vista del motor gráfico, Unity se encuentra entre el top 10 de los mejores motores usados por la industria de videojuegos. Es usado por importantes compañías como Coca-Cola, Electronic Arts, LEGO, Cartoon Network, Disney, Microsoft, Nasa, U.S. Army.

Motor gráfico empleado por desarrolladores de videojuegos para móviles

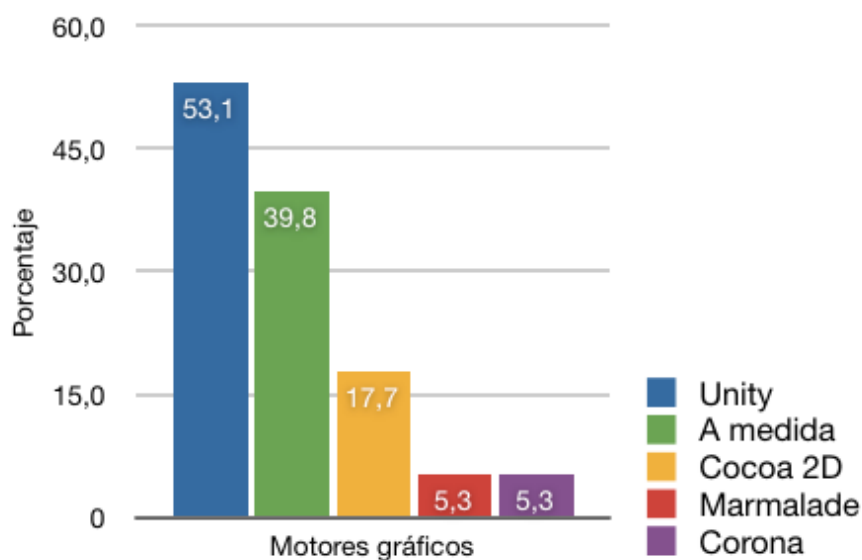


Figura 2. Motores gráficos empleados por los desarrolladores de videojuegos para móviles

En la figura 2 se puede observar la importancia de Unity como motor gráfico para el desarrollo de videojuegos dentro del sector de móviles ocupando la primera posición con el 53,1%¹.

En el anexo A.3 se justifica la importancia de Unity como motor gráfico.

El desarrollo de este proyecto tiene como propósito general desarrollar una aplicación que introduzca al alumno en el mundo de la creación de videojuegos mediante el uso de un motor gráfico puntero cubriendo los aspectos fundamentales del mismo.

Actualmente no existe ninguna aplicación de este tipo, y se pretende que gracias al impacto que tiene en la sociedad el mercado de los videojuegos sea el origen para la creación de aplicaciones interactivas de enseñanza en general y de videojuegos en particular.

Por último añadir que el proyecto aparte de ser novedoso por sus características y por el potencial que posee es interesante desde el punto de vista de trabajo final de una carrera de Ingeniería en Informática. Supone un trabajo que implica la ingeniería del software, estructura de datos y algoritmos, diseño de aplicaciones móviles, análisis de interacción hombre-máquina, servicios web, diseño de bases de datos, informática gráfica, expresión gráfica y redes móviles.

1.2. Objetivos y alcance

El objetivo principal del proyecto es el de crear una aplicación para el sistema operativo iOS (iPad) que haciendo uso del motor gráfico Unity 3D permita impartir un curso completo de programación de videojuegos.

El proyecto presenta otros objetivos secundarios:

- ▶ Formar al alumno en los aspectos fundamentales del manejo de Unity 3D.
- ▶ Mostrar el manejo de un motor gráfico. Los motores gráficos al igual que ocurre con otras aplicaciones presentan similitudes en su manejo y características.

¹ Nota: el porcentaje total es superior al 100% eso es debido a que hay desarrolladores que emplean varios motores gráficos.

- ▶ Guiar al alumno en el proceso de creación de un videojuego.
- ▶ Crear un Documento de Diseño del Juego (GDD) que sirva como ejemplo de documentación generada durante la creación de un videojuego.
- ▶ Motivar a los alumnos del curso a crear aplicaciones interactivas.
- ▶ Mostrar otras especialidades implicadas en el proceso de creación de videojuegos abriendo nuevas posibilidades de desarrollo para el alumno.

Para el desarrollo de la aplicación se utilizará el propio entorno de Unity 3D, el IDE (Entorno de Desarrollo Integrado) MonoDevelop y como lenguajes de programación se emplearán UnityScript (variante de JavaScript) y C#.

El desarrollo del proyecto se ha realizado en varias partes que se indican a continuación:

La primera parte del proyecto consiste en la creación de la documentación escrita que cubre los aspectos fundamentales del curso. Junto a esta documentación escrita se han creado los vídeos de ejemplo que tienen relación con ella y muestran el resultado final de las explicaciones realizadas. Para complementar esta primera parte, se han creado las aplicaciones interactivas realizadas con Unity 3D.

La segunda comprende la creación del videojuego de ejemplo completo y la documentación escrita que se corresponde con este. Esta documentación estaría formada por el GDD (Documento de Diseño del Juego) y por una descripción detallada del proceso de creación del videojuego.

La tercera parte consiste en la creación de la aplicación web que interacciona con el juego completo. El videojuego realiza peticiones a la base de datos sobre los usuarios del juego para poder crearlos y posteriormente, si se desea, modificarlos. Una vez creado un usuario si se consigue una puntuación máxima (dentro del top 100) se registra en la base de datos y se muestra en la web, enviando posteriormente un mensaje al juego indicando la posición lograda en ese top 100.

Por último la cuarta parte del proyecto es la creación del visor de documentos que permite también la reproducción de los vídeos y la ejecución de las aplicaciones interactivas y del videojuego completo. Es decir

aglutina todas las partes del proyecto a excepción de la aplicación web y permite reproducirlas o ejecutarlas sin salir de la aplicación.

1.3. Estudios preliminares

Pese a realizar un PFC en el que se predefinen las características tecnológicas a nivel hardware (iPad) y software (Unity 3D y sus lenguajes asociados junto a Xcode y Objective-C), en el anexo A se ha realizado un estudio que pone en relevancia la importancia de la industria de los videojuegos, el uso de dispositivo móviles y las características y razones por las que se ha elegido un iPad como dispositivo.

Debido a su extensión y a que se han justificado parcialmente las razones anteriormente mencionadas en el apartado “Contexto y motivación” del presente documento, se remite al lector al anexo A redactado al final de la memoria.

1.4. Tecnología. Uso y justificación

Aunque el proyecto verse sobre un curso de videojuegos con un motor y dispositivo determinados, para su realización se han utilizado diversos entornos y herramientas de trabajo.

En este punto se detallan y justifican las tecnologías que han sido empleadas aunque habrá momentos en los que se remita al lector a los anexos en los que existen versiones más detalladas de estas o que han sido utilizadas pero resultan de menor entidad en la realización del proyecto y que se ha decidido no mencionar en este apartado.

A continuación se va a ir indicando la tecnología empleada para cada una de las partes en las que se ha realizado el proyecto.

Para la **primera parte** se han utilizado diversas herramientas, por ejemplo, para la creación de los textos del curso se han empleado: editores de texto como Pages, de presentaciones como Keynote y de imágenes como Gimp e Inkscape y hojas de cálculo como Numbers (ver anexo B).

Los vídeos del curso se han creado utilizando ScreenFlow, aplicación para la grabación de la pantalla y edición de video.

Por último para la creación de las aplicaciones interactivas se ha hecho uso de Unity 3D (ver anexo D) como entorno de desarrollo, ya que permite integrar el proyecto en una única aplicación, realizar pruebas mientras se realiza el desarrollo y servir como ejemplo de creación de aplicación para el curso. Con Unity se ha modelado el entorno y creado los objetos necesarios para poblar las escenas de las aplicaciones. Para la programación del comportamiento de los objetos creados con Unity se ha hecho uso de los lenguajes de programación C# y UnityScript (variante de JavaScript) (ver anexo D).

Para la **segunda parte** del proyecto se ha hecho uso de varias herramientas que pertenecen a distintas disciplinas, que forman parte de la creación de un videojuego y que han sido usadas en la creación del videojuego completo de ejemplo del curso. Para el modelado y animación de los personajes del juego y objetos del juego se ha empleado Blender (anexo C), por ser una herramienta de software libre que permite crear elementos del juego para ser usados directamente con Unity 3D. Otros motivos por los que se ha elegido Blender para el modelado de los objetos del juego es que existe gran cantidad de documentación en la red sobre modelado de objetos en 3D, por su facilidad de uso para crear objetos simples a partir de formas geométricas primitivas y por su integración con otras aplicaciones usadas en el proceso de creación de los elementos de un videojuego (como por ejemplo creación de mapas UV para ser utilizados con Gimp).

La siguiente herramienta utilizada es Gimp, aplicación para la manipulación de imágenes, que como se comentó, fue usada en la primera parte. Con Gimp se han creado las texturas de los objetos en 3D a partir de los mapas UV generados con Blender.

Algunos elementos usados en el juego han sido directamente creados con Unity 3D. Una vez creados se les ha aplicado una textura importada creada con Gimp y al igual que se hizo con los objetos en la primera parte del proyecto se programó con C# y UnityScript su comportamiento. Unity y UnityScript también se han empleado para crear todos los niveles (escenas) del juego, menús, control y lógica del juego y la comunicación exterior a través de la aplicación web de la tercera parte.

Para las pruebas del videojuego se han empleado distintas herramientas, se ha usado Xcode y un iPad que ha permitido probar el juego en un dispositivo real. Otra herramienta muy útil ha sido Unity Remote, esta herramienta permite realizar pruebas de aplicaciones con el propio entorno Unity mientras se usa el dispositivo iOS como unidad controladora. Unity remote

desempeña su cometido haciendo una transmisión (streaming) del juego a la unidad iOS a través de una conexión Wifi, capturando la entrada de usuario desde la unidad iOS pasándola al entorno de Unity donde es tratada.

Por último para la documentación de esta parte se han empleado las herramientas ya citadas de la primera parte.

Para la **tercera parte** del proyecto se ha utilizado MySQL como gestor de la base de datos, ya que no se necesita una base de datos de gran tamaño, no está prevista una alta concurrencia en la modificación y en caso de darse un acceso masivo sería en lectura de datos (mostrar las puntuaciones), todo esto hace que MySQL sea el gestor idóneo.

Para la aplicación web con contenido dinámico se han utilizado los lenguajes de programación PHP y JavaScript junto a la tecnología AJAX (anexo E). En las primeras fases de desarrollo se ha empleado el software MAMP (Mac Apache MySQL PHP) para la realizaciones de pruebas en un entorno local, ya que permite la gestión y control del servidor web y de la base de datos. También se ha empleado CSS para dar formato a la página de resultados, separando así, el contenido del diseño de las páginas.

En la **cuarta** y última **parte** se ha creado el visor que permite unir entre sí las dos primeras partes, es decir, permitirá ver el contenido del curso (texto, vídeos y aplicaciones interactivas) y lanzar el videojuego. Para ello se ha empleado Unity 3D junto a UnityScript y Objective-C.

Por último para la creación de la aplicación a ser ejecutada en un iPad, se ha utilizado el entorno de programación Xcode, ya que es la única forma a través de la cual se puede realizar la instalación de la aplicación y poder realizar pruebas reales en un dispositivo iOS.

1.5. **Ámbito del proyecto**

El ámbito de trabajo del proyecto reúne las siguientes características:

- ▶ El trabajo se ha realizado de forma remota utilizando un portátil Mac Book Pro con Mac OS X versión 10.8.3 como sistema operativo.
- ▶ Para las pruebas de la aplicación se ha empleado un iPad de tercera generación con sistema operativo OS X versión 6.1.3.

- ▶ La mayor parte de las pruebas han sido de forma local, realizadas con el portátil indicado a excepción de las pruebas de comunicación que se han realizado con un servidor externo.
- ▶ Para los intercambios de ficheros, las revisiones de los textos, vídeos y aplicaciones se ha creado un repositorio de acceso común con el director del proyecto.
- ▶ Se han mantenido contactos constantes con el director del proyecto vía correo electrónico y mediante reuniones personales.

1.6. Metodología y planificación

Se ha elegido como metodología de trabajo el modelo en cascada modificado (Figura 3) ya que es el que mejor se adapta a las necesidades del proyecto por su mayor flexibilidad respecto al modelo en cascada puro y a que permite desarrollar partes sencillas del proyecto sin esperar a terminar otras más complicadas.

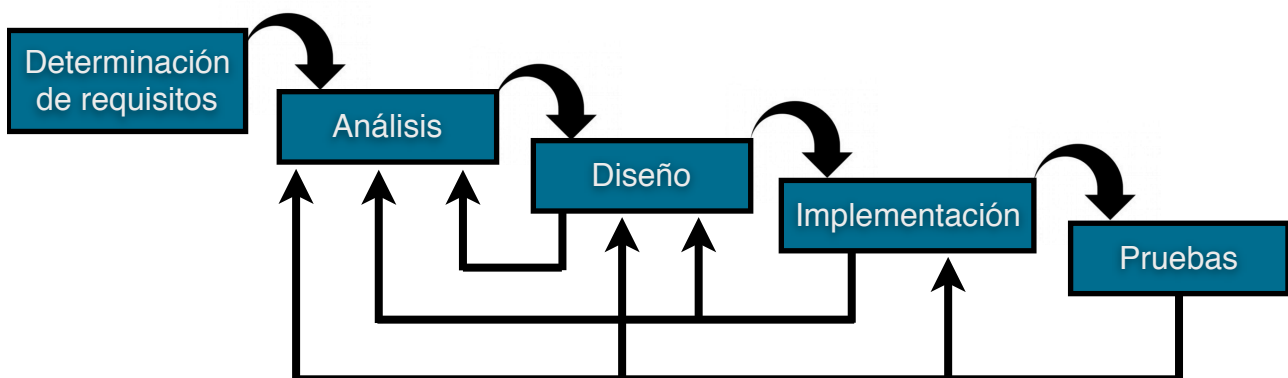


Figura 3. Diagrama del modelo en cascada modificado

En la figura 3 se muestra el diagrama del modelo en cascada modificado con sus cuatro fases de desarrollo. Como se observa en el diagrama desde cada una de las fases se puede volver a cualquiera de las anteriores.

Antes de estas fases se ha realizado una captura de requisitos que debería tener el sistema. Estos requisitos pueden ser funcionales o no funcionales como se verán en el anexo F en el que se detalla esta fase.

Previo a comenzar con el desarrollo de las cuatro fases principales comentadas ha sido necesario realizar un aprendizaje sobre las herramientas, algunas tecnologías y algunos lenguajes de programación que han sido empleados en la realización del proyecto; se mencionarán en la bibliografía de la presente memoria. Este proceso de aprendizaje ha sido continuo debido al carácter multidisciplinar del proyecto.

En cuanto a la planificación en el anexo F.3 se muestra la planificación inicial realizada antes de comenzar con el proyecto y la planificación real.

1.7. Estructura del documento

El documento se divide en dos partes o bloques. El primer bloque es la memoria que consta de cuatro capítulos y el segundo bloque son los anexos que contienen nueve capítulos.

► **Memoria.** Parte principal del documento, está formado por los siguientes capítulos:

- Capítulo 1 - Introducción:

Breve introducción al proyecto, contexto y razones por las que se ha realizado, objetivos del mismo, estudios preliminares realizados, tecnología empleada en su realización, ámbito, metodología empleada y estructura del documento.

- Capítulo 2 - Trabajo realizado:

Descripción del trabajo que se ha realizado durante todo el proyecto, visión en detalle de cada una de las partes del mismo y su proceso de desarrollo: visor, videojuego, aplicación web, vídeos y aplicaciones interactivas.

- Capítulo 3 - Conclusiones:

En este capítulo se expone el estado de consecución de objetivos, cómo podría orientarse el desarrollo del mismo para futuras ampliaciones y mejoras de la aplicación, perspectivas de futuro de la misma y una valoración personal del proyecto.

► **Anexos.** Información extendida de alguno de los capítulos de la memoria, consta de los siguientes anexos:

- Anexo A - Estudios preliminares:

Estudios preliminares realizados sobre la situación de la industria de los videojuegos, importancia de los dispositivos móviles en especial los de Apple y la importancia del motor gráfico Unity 3D en el desarrollo de videojuegos.

- Anexo B - Tecnologías empleadas:

Resumen de las tecnologías empleadas para la elaboración del proyecto.

- Anexo C - Modelado de objetos con Blender:

Proceso de creación de los modelos con la herramienta Blender y fichas empleadas para el modelado de los objetos 3D.

- Anexo D - Unity 3D:

Descripción de los lenguajes empleados en el curso, aplicaciones interactivas y creación del videojuego.

- Anexo E - Tecnologías web empleadas:

Descripción de las tecnologías web usadas para la creación de la aplicación web soporte del videojuego.

- Anexo F - Documentación del proyecto:

Ampliación de la documentación surgida en el desarrollo del proyecto.

- Anexo G - Manual de usuario:

Documento que contiene el manual de usuario para el uso del visor, videojuego completo, vídeos y aplicaciones interactivas.

- Anexo H - Documento de Diseño del Juego:

Ejemplo de documentos empleados durante el diseño del juego.

- Anexo I - Ejemplo de capítulo del curso:

Capítulo de ejemplo del curso.

2. Trabajo realizado

Este capítulo está dedicado a detallar todo el trabajo realizado a lo largo del Proyecto Fin de Carrera. Como se ha comentado anteriormente el proyecto parte de una idea innovadora del director, no hay aplicaciones en el mercado que sean un “todo en uno” (visor - documentación - vídeos - aplicaciones interactivas) en cuanto a formación en creación de videojuegos. Esta situación de partida permite un grado de libertad mayor frente a otro tipos de trabajos que forman parte de algún sistema ya realizado pero generan una carga de trabajo importante siendo necesario centrarse en los puntos fundamentales y desechar funcionalidades del sistema que no son primordiales. En el anexo F se encuentra la documentación de planificación del proyecto.

El sistema creado se muestra en la figura siguiente:

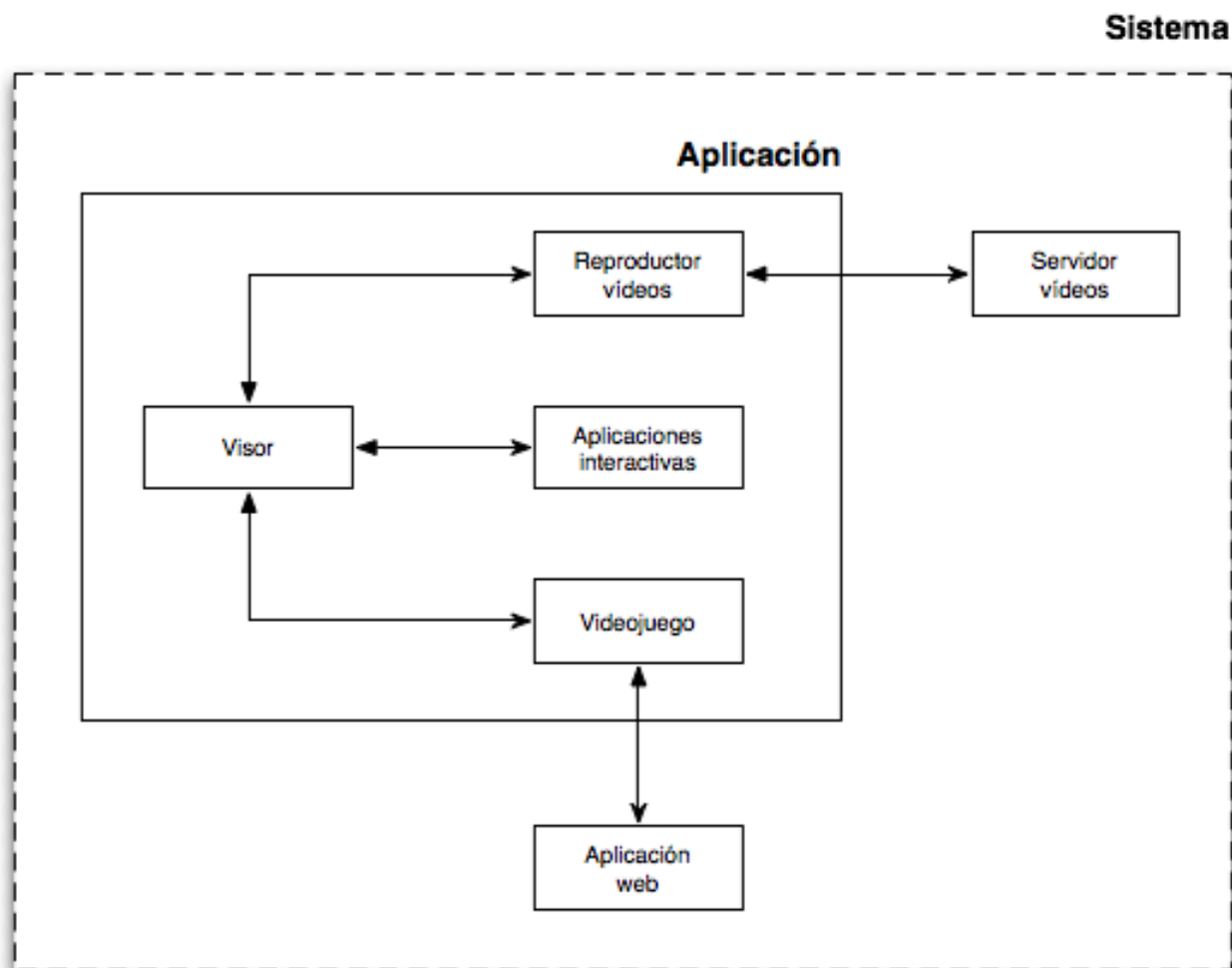


Figura 4. Visión general del sistema creado

En la figura 4 se observan las partes fundamentales del sistema, el visor, el videojuego, las aplicaciones interactivas, los vídeos y la aplicación web. En el diagrama también se observan las comunicaciones existentes entre cada una de ellas. La aplicación que se instala en el dispositivo iOS está formada por:

- ▶ El visor, que contiene los temas escritos y es el responsable de interactuar con las otras partes de la aplicación.
- ▶ El reproductor de vídeos que permite la reproducción de los vídeos sin salir de la aplicación. Los vídeos como se observa están situados fuera de la aplicación consiguiendo de esta forma reducir el tamaño de la aplicación creada.
- ▶ Las aplicaciones interactivas, que son pequeñas aplicaciones por sí solas y que permiten un grado de interacción con el usuario. Son aplicaciones que resumen y sirven de ejemplo de los conceptos explicados en el tema.
- ▶ El videojuego, que es una aplicación por sí sola, es una parte del trabajo en la que se ha invertido gran cantidad del tiempo de desarrollo. Por un lado en su análisis, qué se debería hacer, qué elementos debería tener, etc. Por otro lado se ha tenido que diseñar el personaje principal, los enemigos, los niveles, la interacción con los personajes y con el entorno, etc. También se han tenido que crear los elementos que pueblan el videojuego y crear la lógica necesaria para que se pueda crear el juego. Se han tenido que realizar las pruebas que confirmasen que los elementos con los que hay interacción en el juego funcionasen de forma correcta. Por último y no menos importante se han tenido en cuenta todos los elementos que forman parte de un videojuego, ambiente, efectos de sonido, música, historia, etc.

Para alguna de estas partes se han realizado los mismos pasos que para la aplicación en su conjunto, es decir, se ha evaluado su necesidad, el objetivo dentro del sistema, se ha realizado una determinación de las necesidades (requisitos) que planteaban, se han analizado, se han tenido en cuenta cuestiones de diseño, se ha realizado su implementación y por ser partes del sistema pruebas para cada uno de estos componentes y pruebas de integración.

Las otras partes que forman parte del sistema pero no de la aplicación son:

► Aplicación web: esta aplicación se ha creado por dos motivos, uno el que la aplicación tuviese una comunicación con el exterior y que no fuese totalmente cerrada. Segundo motivo, dar soporte al videojuego de ejemplo realizado. Este soporte consiste en permitir la creación de una tabla de puntuaciones que pueda ser consultada por internet.

2.1. Visor

2.1.1. Introducción

El objetivo principal de este proyecto es la creación de un curso de programación de videojuegos en Unity 3D empleando un iPad que permita reproducir ejemplos.

El primer elemento que se va a describir es el visor; el visor es un elemento fundamental del proyecto, como se ha comentado es la herramienta que permite la visualización de los documentos creados y lanzar el resto de aplicaciones (reproductor de vídeos, aplicaciones interactivas y ejemplos). Una vez que se se lanza un vídeo, una aplicación interactiva o el videojuego existe la posibilidad de volver otra vez al visor.

Para determinar el tipo de visor que se iba a desarrollar, se ha realizado un estudio (que se mostrará en el siguiente apartado) de las aplicaciones que son visores de documentos o de aplicaciones en las que esta funcionalidad tenga importancia dentro de la aplicación.

2.1.2. Estudio preliminar

Se ha realizado un estudio de visores de documentos con una valoración alta en el mercado de aplicaciones. Este estudio pretende determinar las características fundamentales que disponen estas aplicaciones y que sirvan para modelar el visor de la aplicación.

Este estudio preliminar se encuentra en el anexo F.1 junto a aquellas características que se han determinado como interesantes para implementar en el visor.

2.1.3. Determinación de requisitos

Para la determinación de requisitos ha sido fundamental el estudio preliminar realizado en el apartado anterior. En el anexo F.2 se especifica con mayor detalle todos los requisitos del sistema, a continuación se resumen los más importantes que afectan al visor y a parte del entorno:

Requisitos funcionales:

- ▶ El visor debe realizar zoom al documento cuando el usuario pulse con dos dedos la pantalla.
- ▶ El visor debe mostrar las opciones de la interfaz al pulsar en una determinada zona.
- ▶ El visor debe mostrar un índice de temas.
- ▶ El visor deberá cargar el índice de temas desde el menú que aparece en la interfaz.
- ▶ El visor debe cargar un índice de vídeos.
- ▶ El visor deberá cargar el índice de vídeos desde el menú que aparece en la interfaz.
- ▶ El visor debe cargar un índice de aplicaciones.
- ▶ El visor deberá cargar el índice de aplicaciones desde el menú que aparece en la interfaz.
- ▶ El visor debe cargar un tema del curso determinado al ser seleccionado desde el índice de temas.
- ▶ El visor debe reproducir un vídeo determinado al ser seleccionado desde el índice de vídeos.
- ▶ El visor debe ejecutar las aplicaciones interactivas al ser seleccionadas desde el índice.
- ▶ El visor debe permitir avanzar y retroceder las páginas del curso.

- ▶ Al avanzar una página si el tema termina el visor deberá cargar el siguiente tema. Si es el último tema no hará nada.
- ▶ Al retroceder una página si es la primera página del tema el visor cargará el tema anterior. Si es el primero, se cargará el índice de temas.
- ▶ El visor ocultará la interfaz al avanzar o retroceder página si está desplegada.
- ▶ El visor ocultará la interfaz en el inicio y contenidos de cada tema.

Requisitos no funcionales:

- ▶ Los vídeos deberán almacenarse en un servidor web externo (por motivos de espacio, para que no ocupe demasiado la aplicación).
- ▶ El visor se debe ejecutar en un iPad.
- ▶ La interfaz debe estar adaptada a las características de navegación y usabilidad de dispositivos móviles.
- ▶ El usuario interactúa desde un entorno sencillo e intuitivo.
- ▶ La aplicación debe funcionar en dispositivos iOS 5.0 o superior.

2.1.4. Casos de uso

En la figura 5 se recogen los casos de uso estándar que describen el funcionamiento del visor. El usuario de la aplicación puede ser el profesor que realiza la enseñanza o un alumno que está inscrito en el curso.

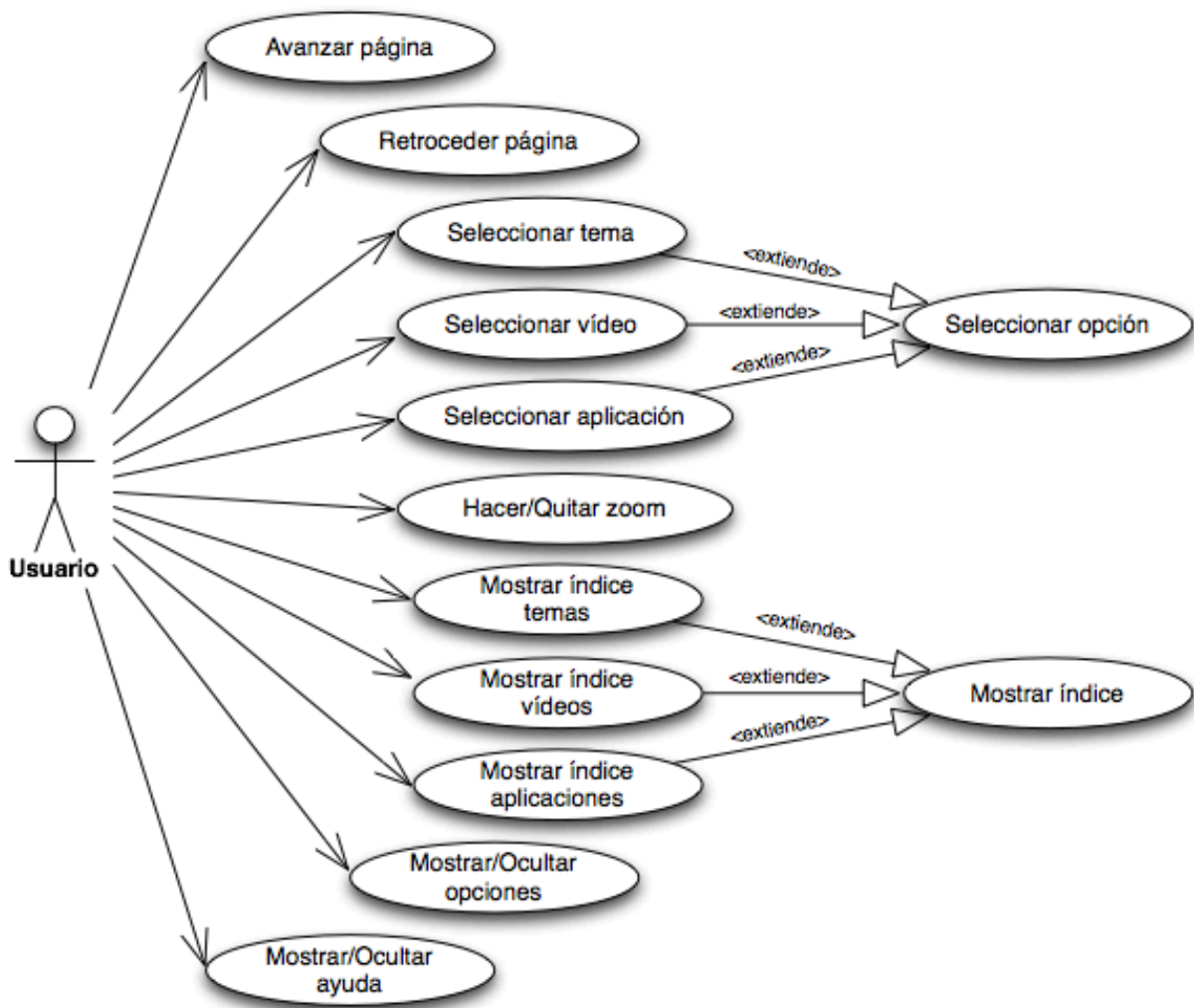


Figura 5. Caso de uso general del visor

A continuación se muestra en detalle el caso de uso de la figura 5:

- ▶ Los dos primeros casos de uso que se describen son los de navegación del visor, para la parte escrita es fundamental tener mecanismos que permitan avanzar y retroceder las páginas del documento.
- ▶ Los tres casos siguientes afectan a la selección de un tema, vídeo y aplicación, una vez cargado el índice apropiado.
- ▶ Para tener acceso a las características del punto anterior, se hace necesario poder cargar los índices de temas, vídeos y aplicaciones.
- ▶ Puede haber momentos en la explicación de un tema que interese mostrar alguna parte del documento, principalmente porque el tamaño

sea algo pequeño, con la opción de zoom, se pueden mostrar los contenidos que se deseen con más detalle.

- ▶ La carga de los índices se hace desplegando el menú de opciones, una vez desplegado se selecciona el índice deseado.
- ▶ Por último para aquellas personas que son nuevas en el manejo de la aplicación o para aquellas que necesiten en un momento dado recordar cómo se usa, se ha integrado la opción de mostrar la ayuda sobre el manejo del visor desde el menú lateral de opciones.

2.1.5. Decisiones de diseño

Las decisiones de diseño del visor se han basado en el diagrama de actividades de la figura 6. Como se ve, al iniciar la aplicación se carga una página del curso (en este caso la portada), se puede navegar a través de los documentos con pulsaciones en la pantalla. La forma de interactuar con la aplicación se verá en el diseño de la interfaz del apartado siguiente.

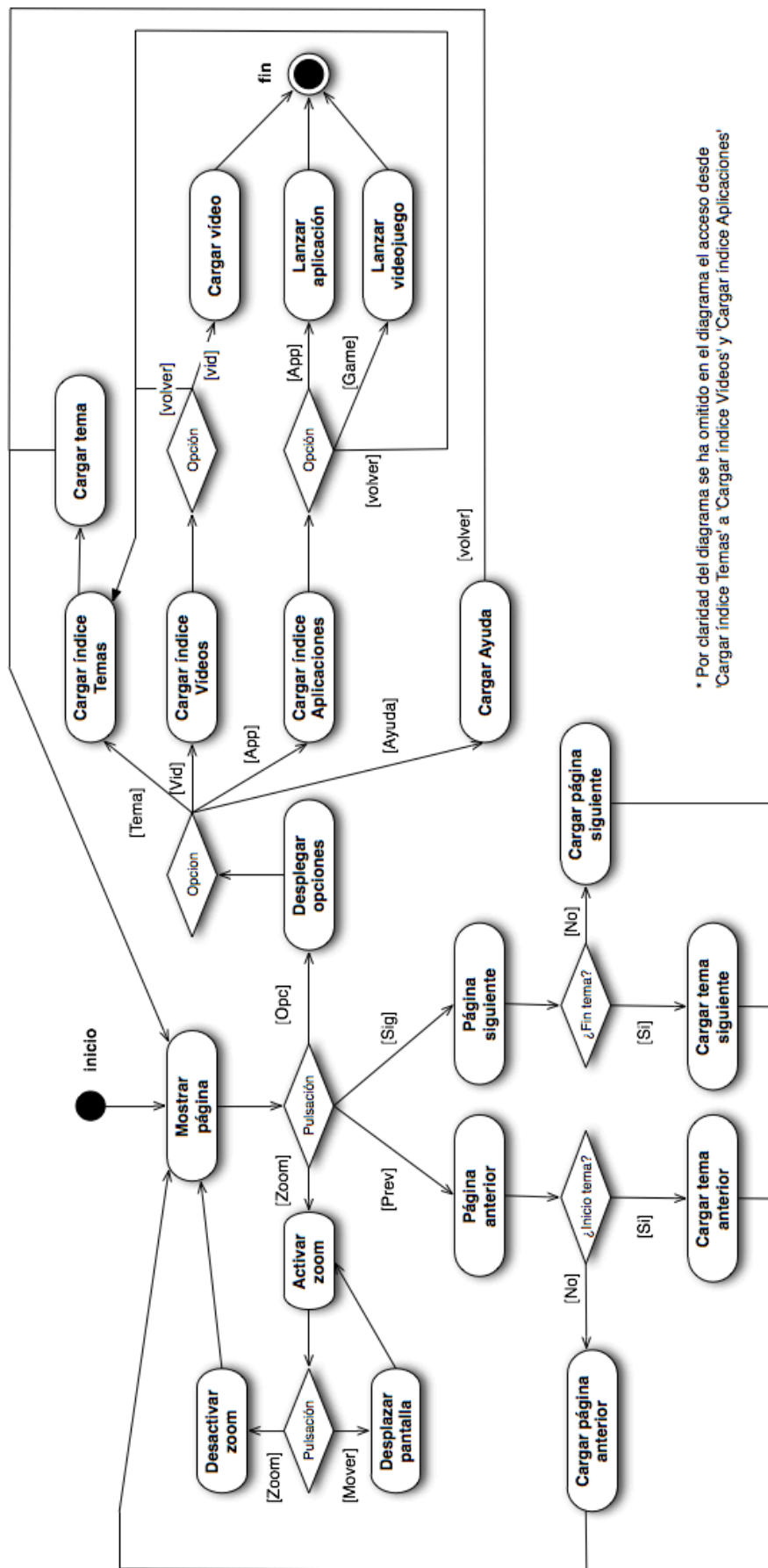
Como se ha comentado, para navegar a través de los documentos, es necesario pulsar en la pantalla, de esta forma se carga la página siguiente, la anterior o el zoom. Otra opción que se puede activar por pulsación es el despliegue del menú de opciones, desde el que se puede acceder al índice de temas, de vídeos, de aplicaciones o a la ayuda.

Si se pulsa la pantalla para hacer zoom, se activará y solo se permitirá el desplazamiento de pantalla hasta que se salga del zoom.

Si se accede al índice de temas, se podrá cargar cualquier tema en el visor, así como acceder a los vídeos, aplicaciones o ayuda.

Si se accede al índice de vídeos, se podrá lanzar el reproductor para que cargue el vídeo mediante streaming o se podrá volver a cargar el índice de temas. No se ha añadido la comunicación desde 'Cargar índice Temas' hacia 'Cargar índice vídeos' y 'Cargar índice Aplicaciones' por motivos de claridad.

Si se accede al índice de aplicaciones, se podrá cargar cualquier aplicación interactiva o el ejemplo de videojuego completo. También se ofrece la opción de volver a cargar el índice de temas.



* Por claridad del diagrama se ha omitido en el diagrama el acceso desde 'Cargar índice Temas' a 'Cargar índice Videos' y 'Cargar índice Aplicaciones'

Figura 6. Diagrama de actividades del visor

2.1.6. Diseño de la interfaz

A partir del análisis de requisitos y atendiendo a criterios de usabilidad se ha creado el diseño de la interfaz de usuario. Para el diseño de la interfaz se ha seguido el siguiente diagrama de navegación:

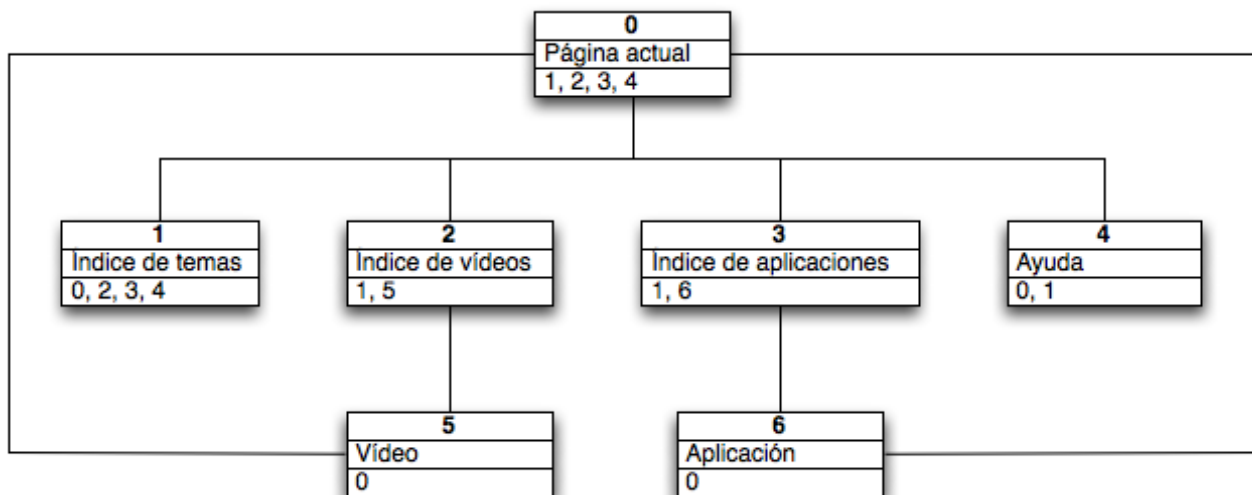


Figura 7. Jerarquía de ventanas del visor

En la figura 7 se observa la navegación que se permite en el visor de acuerdo al diseño comentado en el apartado anterior.

Los gestos que se reconocen en la interfaz de usuario viene recogidos en el anexo G.2, en el que se describen con detalle todas las posibilidades de interacción que existen con el dispositivo.

A continuación se mostrarán las capturas de pantalla de la aplicación en las que se pueden observar las características y el diseño de la interfaz de usuario.



Figura 8. Aspecto de la interfaz del curso con una página de texto cargada

En esta pantalla (figura 8) se muestra el aspecto de la interfaz con una página del texto del curso cargada.

Las interacciones permitidas con el usuario son:

- ▶ Pulsar en el lado derecho de la pantalla hará que se cargue la siguiente página del curso (si se llega a la última imagen, no se cargará ninguna página).
- ▶ Pulsar en el lado izquierdo de la pantalla hará que se cargue la página anterior (si se está en la primera página se cargará el índice de temas).
- ▶ Si se pulsa con dos dedos a la vez sobre la pantalla, se activa el zoom. Una vez activado el zoom, se permite el movimiento de la pantalla hasta alcanzar los límites de la hoja. Para desactivar el zoom, se pulsará de nuevo con dos dedos sobre la pantalla.

- ▶ Otra interacción permitida es la de activar el menú de opciones, para ello se pulsará sobre la cinta de color rojo, al pulsarla se mostrará el menú de opciones de la imagen siguiente:



Figura 9. Aspecto de la interfaz con el menú de opciones desplegado

Como se puede observar, aparecen las opciones indicadas durante los apartados previos:

- ▶ Si se pulsa sobre 'Ejemplos', se cargará el índice con todos los vídeos del curso.
- ▶ Si se pulsa sobre 'Índice', se cargará el índice de temas.
- ▶ Al pulsar sobre 'Ayuda', se cargará la ayuda en la que se muestran instrucciones acerca del manejo de la aplicación.

► Por último si se pulsa sobre 'Vídeos' se cargará el índice con todos los vídeos del curso. Este índice se puede observar en la imagen siguiente:



Figura 10. Aspecto de la interfaz con el menú de vídeos cargado

En la figura 10 se observa el índice de vídeos, pulsando sobre cualquiera de ellos se lanza el reproductor de vídeos (dentro de la aplicación) y se realiza la carga del video mediante streaming desde el servidor web donde se encuentran alojados.

En el anexo G.2 existe documentación ampliada con ejemplos sobre la interfaz de usuario de la aplicación.

2.1.7. Implementación del visor

Para la implementación del visor se ha empleado el patrón Modelo - Vista - Controlador (MVC). El patrón de diseño MVC adoptado se muestra en la siguiente figura:

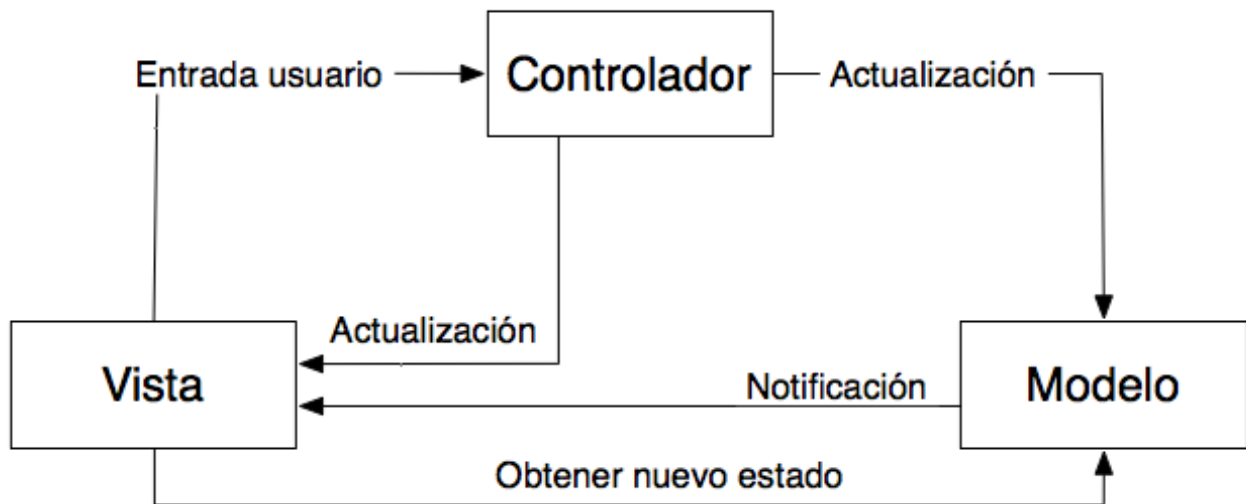


Figura 11. Diagrama del patrón de diseño Modelo - Vista - Controlador

Este patrón es ampliamente empleado en la implementación de interfaces gráficas de usuario (GUI). Por ejemplo es el patrón típico con alguna variante mínima para programar todas las ventanas de una aplicación iOS programada en Xcode con Objective-C.

Atendiendo a este patrón (figura 11), cuando el usuario realiza alguna acción en la pantalla del dispositivo, como por ejemplo pulsar en la parte derecha de la pantalla, la vista envía la entrada de usuario al controlador, si el controlador determina que se debe realizar alguna modificación en el modelo, como por ejemplo es el caso, es decir, se debe pasar a la siguiente página, el controlador se comunica con el modelo para indicarle que debe actualizar los datos. Una vez que este actualiza los datos, el modelo se lo notifica a la vista, que solicitará el nuevo estado al modelo, tras esto el controlador dará la instrucción de carga a la vista actualizándose así la pantalla.

El ejemplo que se acaba de describir se puede observar en el siguiente diagrama:

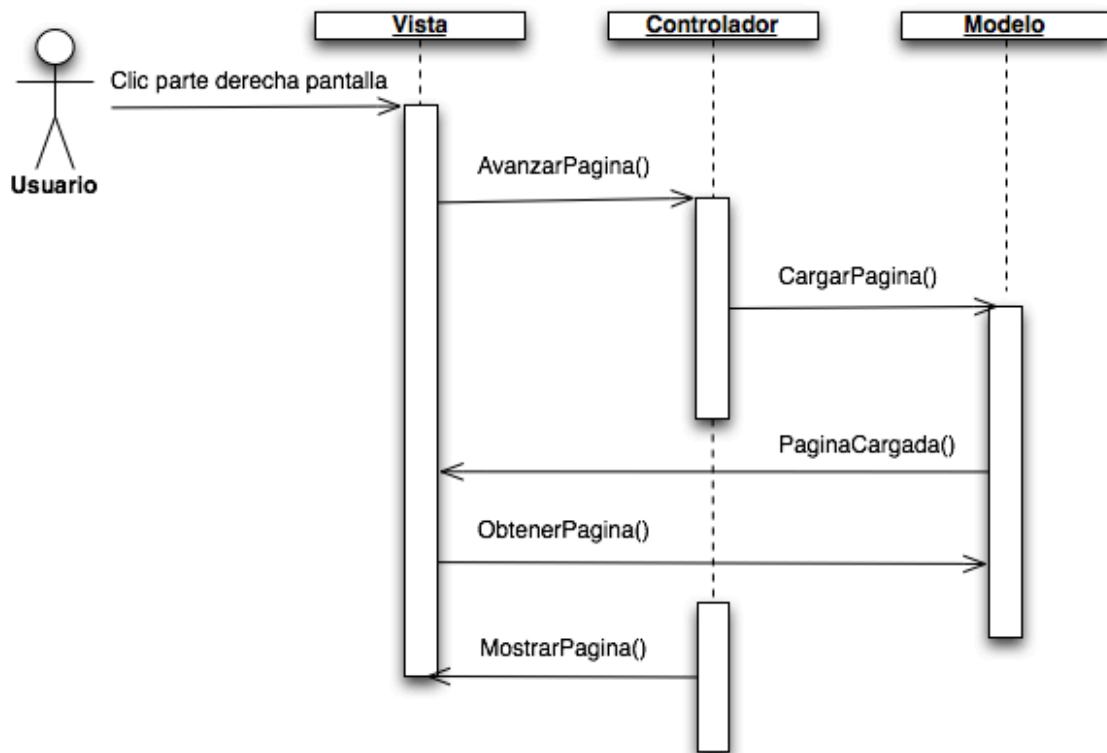


Figura 12. Diagrama de secuencia del caso de uso avanzar página

2.2. Videojuego “La venganza de Timan”

2.2.1. Introducción

El videojuego completo creado como ejemplo ha sido una de las partes que más tiempo ha consumido del desarrollo del PFC. Esto es debido a grandes rasgos a varios factores: estudio de las diferentes tecnologías necesarias para el desarrollo del videojuego, análisis y diseño del mismo, modelado y animación de los elementos del juego, programación del videojuego y pruebas.

Se ha creado un juego en 3D de plataformas para iPad en el que el personaje principal interactúa con el jugador mediante pulsaciones en la pantalla.

2.2.2. Análisis y diseño

Los requisitos fundamentales que se han planteado para el videojuego se listan a continuación (los requisitos se amplían en el anexo F.2).

Requisitos funcionales:

- ▶ El personaje se debe mover mediante pulsaciones por la pantalla.
- ▶ El personaje eliminará a los enemigos al caer encima de ellos, como ocurre en los juegos normales de plataformas.
- ▶ El personaje perderá una vida si entra en contacto con un enemigo por una parte que no sea la superior y también perderá una vida si se ve afectado por ciertos elementos.
- ▶ El juego debe tener dos niveles de dificultad.
- ▶ El juego debe tener tres niveles o escenas.
- ▶ El juego deberá tener otros elementos con los que interaccionar.
- ▶ El juego debe tener y gestionar un sistema de puntuaciones interno.
- ▶ El juego deberá almacenar máximas puntuaciones del usuario.
- ▶ El juego deberá mostrar estadísticas de las partidas.
- ▶ Se deberá mostrar una ayuda sobre la interfaz y el movimiento.
- ▶ El jugador podrá registrarse en el sistema. Sin registro no deberá haber acceso al juego.
- ▶ El juego permitirá cambiar los datos del jugador.
- ▶ Al terminar una partida, el juego deberá comunicar a una aplicación web la puntuación del jugador.
- ▶ Se deberá poder borrar todos los datos almacenados.
- ▶ Se permitirá volver al visor de la aplicación principal.

Requisitos no funcionales:

- ▶ Al igual que ocurría con el visor, el videojuego debe ejecutarse en un iPad.

- ▶ El juego debe tener alguna historia de fondo.
- ▶ Se deberá intentar crear ambientación adecuada para cada nivel del juego.
- ▶ La base de datos de las partidas debe usar el gestor MySQL.

Tras determinar los requisitos más importantes se ha realizado el diagrama de casos de uso (anexo F.4.1) y el diagrama de flujo de datos (DFD) que se muestra en la figura 13. De esta forma se logra definir las entradas, procedimientos y salidas de la información del videojuego. También permite mostrar la interacción entre el videojuego y las entidades externas, en este caso el usuario. El resto de niveles del DFD se encuentran en el anexo F.4.2.

En el DFD de la figura 13 se observa que el movimiento de datos se produce de la siguiente forma:

- ▶ El usuario proporciona al juego sus datos personales (Datos usuario).
- ▶ El juego muestra al usuario las estadísticas de todas las partidas jugadas (Estadísticas partidas). También se puede mostrar al usuario las máximas puntuaciones locales (Máximas puntuaciones). Se ofrecen los datos de la partida actual (datos partida). Se pueden dar dos casos de error, uno debido a una incorrecta entrada de datos por parte del usuario (Error datos) y la otra porque el usuario ya exista en la base de datos al crear o modificar un usuario (Existe nombre).
- ▶ Validar usuario será lo que se envíe y retorne la base de datos al consultar la existencia de un usuario.
- ▶ Puntuación es la puntuación obtenida en la partida, se debe comprobar con la aplicación web si está dentro del top 100, si es un record se actualiza la base de datos de máximas puntuaciones.

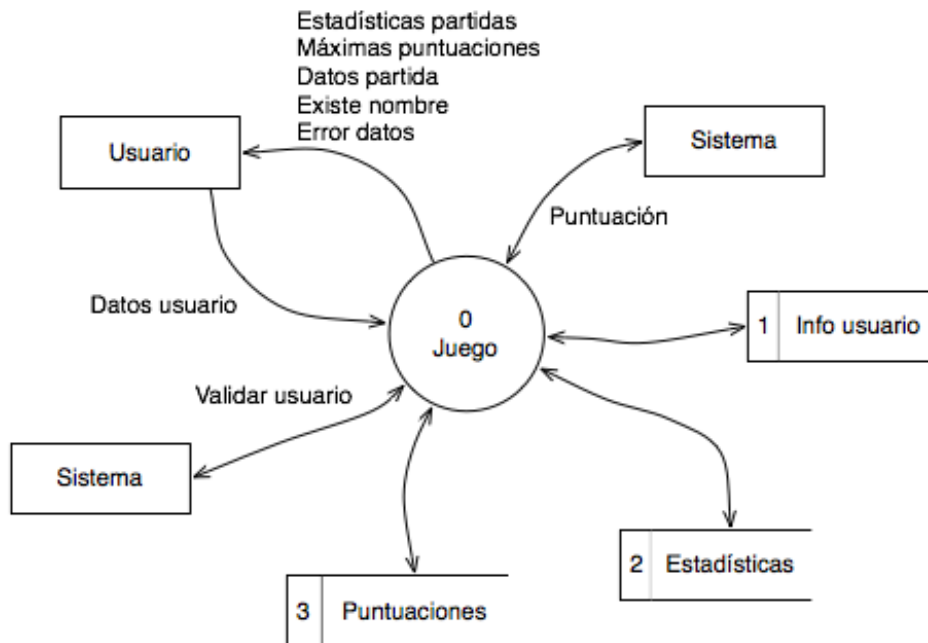


Figura 13. Diagrama de flujo de datos de nivel 0 del juego

En cuanto a las decisiones de diseño, existen gran cantidad de ellas que se han tenido que tomar, en este apartado se comentan las más importantes dejando el resto en el anexo F.4 para que el lector pueda consultarlas.

Uno de los aspectos importantes a diseñar ha sido el de definir el almacenamiento de los datos del juego. Se ha hecho uso de la clase PlayerPrefs de Unity que permite almacenar información entre las sesiones de usuario y entre niveles o escenas (también se ha hecho uso de ficheros para almacenar datos entre niveles o cargar valores iniciales del juego). De esta forma se guardan de forma permanente y local todos los datos necesarios para el videojuego. Se explica su uso en el anexo F.4.3.

En el momento de idear un juego se tienen que tener en cuenta cuatro aspectos fundamentales que se recogen en la figura 14. En la figura se muestran aspectos que van a ser decisivos a la hora de diseñar un juego. Al crear un videojuego este debe ser bonito, divertido, interesante y 'jugable' (término muy empleado en la comunidad de videojuegos para indicar que un videojuego se maneja de forma adecuada). Hay ciertos elementos que potencian estas características, por ejemplo la estética hará que el juego sea bonito, la mecánica que sea divertido, la historia que sea interesante y la tecnología que sea 'jugable'.

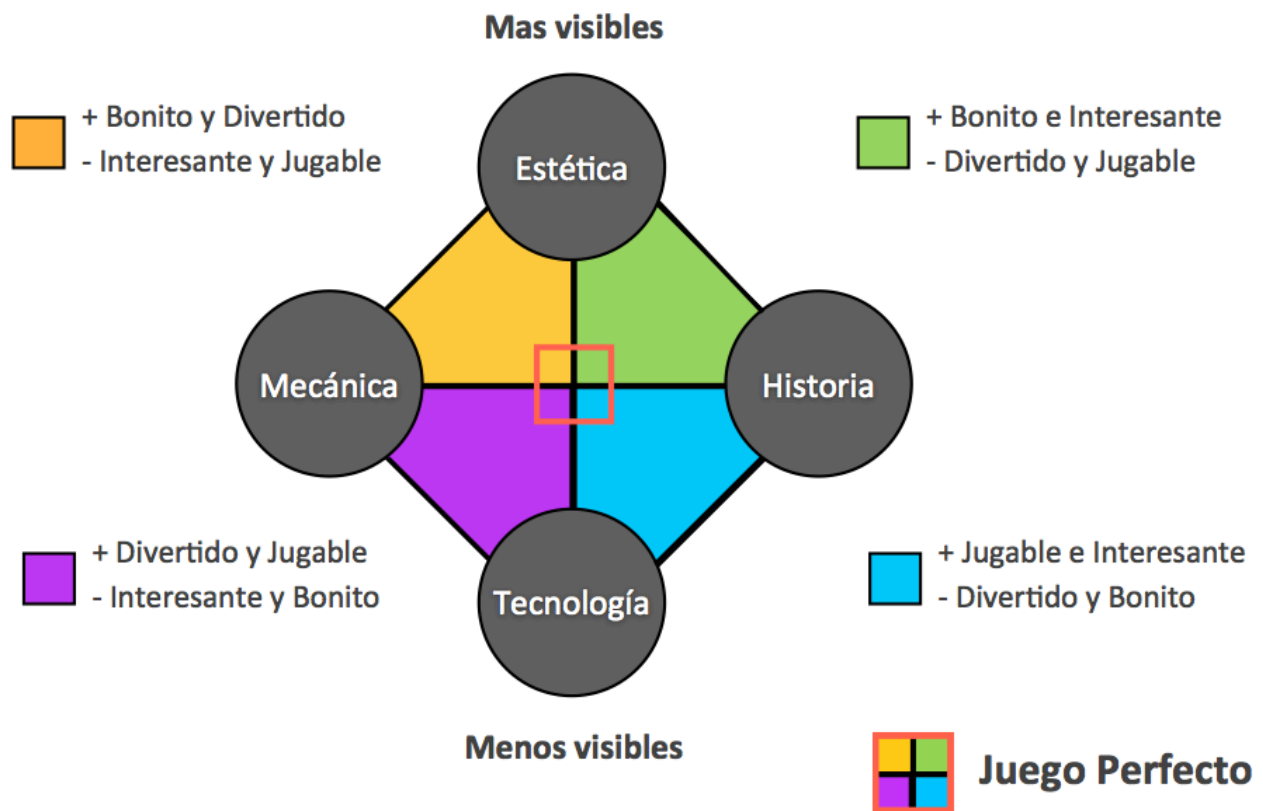


Figura 14. Elementos básicos de un videojuego

Estos elementos se examinan en mayor profundidad en el anexo I. Se comentan a continuación los aspectos más relevantes que han influido en el diseño de los elementos del juego.

Personaje principal:

- ▶ El personaje principal debe reaccionar frente a las interacciones del usuario, la primera decisión es qué tipo de acciones se pueden realizar con el personaje. El personaje podrá saltar, incluso se permitirá un doble salto para acceder a determinadas zonas del nivel y para facilitar el eliminar por ejemplo algunos enemigos, como se muestra en la figura 15.
- ▶ El personaje se moverá de izquierda a derecha. Este movimiento es suficiente al ser un juego de plataformas.
- ▶ La vista del juego será en tercera persona que también es algo típico en los juegos de plataformas. Se deberá colocar la cámara a suficiente distancia como para ver el entorno, enemigos y otros objetos pero sin que queden demasiado pequeños.

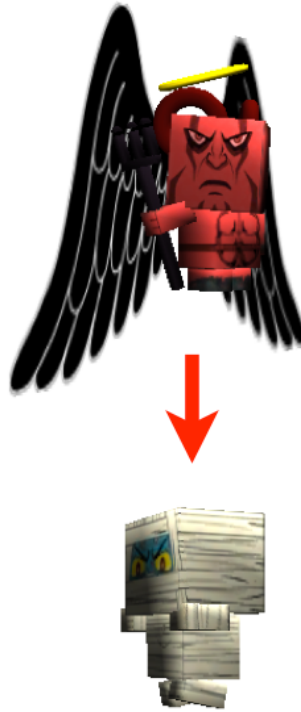


Figura 15. Ejemplo de interacción con el personaje y enemigos en el videojuego

- ▶ Otra consideración es cómo se va a ver afectado el personaje después de realizar los saltos, es decir, que fuerzas van a intervenir. Se ha decidido que la única fuerza que se va a aplicar es la de la gravedad, por lo tanto al objeto le afectará la física creada del entorno.
- ▶ Respecto a la interacción con los enemigos, el personaje principal eliminará a un enemigo cuando caiga encima. El personaje principal perderá una vida si es tocado lateralmente por un enemigo. Por último el personaje puede perder una vida si entra en contacto con superficies mortales como lava o trampas.

Enemigos:

- ▶ Los enemigos son objetos del juego que van a suponer un reto para el jugador. Los enemigos presentan características comunes en cuanto a lógica y movimiento y se diferencian en el aspecto externo.
- ▶ La interacción con el personaje ha quedado descrita anteriormente.

► Al ser un juego de plataformas, los enemigos no están dotados de inteligencia artificial y la dificultad o reto se fija en la velocidad de movimiento y en su localización en las escenas dificultando el paso del personaje principal.



Figura 16. Ejemplo de los enemigos del personaje principal

Superficies mortales:

► Como se ha comentado en los enemigos, estos no disponen de inteligencia artificial, por lo que para crear dificultad en el juego, aparte de los enemigos, se han creado superficies que al entrar en contacto con el personaje provocan la pérdida de una vida.

► Se han creado superficies de dos tipos, unas fijas y otras con desplazamiento.

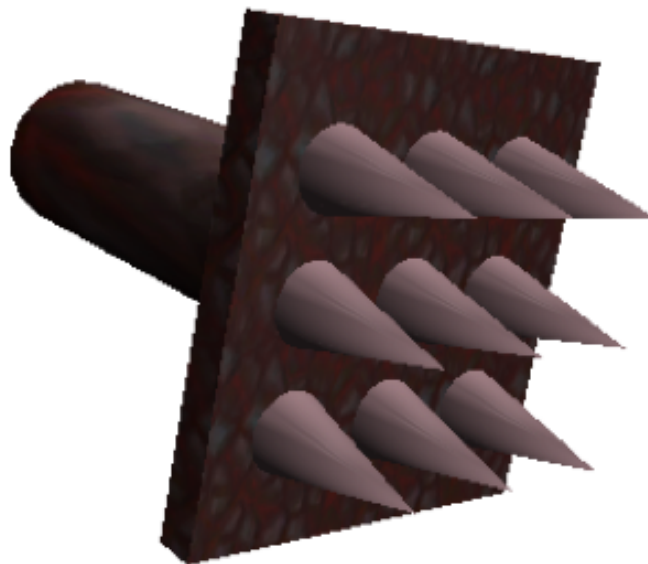


Figura 17. Ejemplo de superficie mortal

Elementos con interacción con el personaje principal:

- ▶ Hay una serie de objetos que van a interactuar con el personaje. Algunos de estos proporcionan puntos, otros sirven de puntos de guardado y otros recuperan vidas perdidas.
- ▶ Se han creado elementos como estrellas y cofres que se encuentran distribuidos por los niveles y que el personaje puede recoger.



Figura 18. Ejemplo de estrella y cofre del videojuego

- ▶ Los puntos de guardado permiten recuperar la última posición guardada por si se muere en alguna zona determinada de un nivel, estos permiten progresar por la escena hasta terminal el nivel.



Figura 19. Ejemplo de checkpoint

- ▶ Por último se han modelado corazones que se encuentran colocados en determinadas zonas especiales y que si se recogen recuperan las vidas perdidas.



Figura 20. Ejemplo de Corazón que representan las vidas del personaje

Sistema de puntuaciones:

- Se ha creado el sistema de puntuaciones que se indica en la tabla 1. Se consiguen puntos por matar enemigos, recoger estrellas y cofres y por acabar en determinado tiempo un nivel.

Elemento	Condición	Puntuación
Vida	Por cada vida	50 puntos
Tiempo (t) en minutos	t < 5 5 < t < 7 7 < t < 10 10 < t < 15	1000 puntos 750 puntos 300 puntos 100 puntos
Estrella	Por cada estrella	10 puntos
Cofre	Por cada cofre	100 puntos
Enemigo	Por cada enemigo	10 puntos

Tabla 1. Resumen de puntuaciones del videojuego

Diseño de los niveles:

- Como se ha comentado en los requisitos, se planteó la necesidad de crear tres niveles. El diseño de estos tres niveles se ha basado en la historia creada entorno al personaje principal (ver anexo I). A continuación se muestra en las siguientes figuras el esquema de los tres niveles. En cada nivel se han empleado distintos diseños que varían completamente el desarrollo de una partida. Más ejemplos de los niveles y las decisiones de diseño empleadas se encuentran en el anexo I.

Leyenda

 Plataforma	 Zona de cajas	 Comienzo nivel
 Plataforma móvil	 Trono	 Fin nivel
 Lava	 Paredes	 Punto de guardado

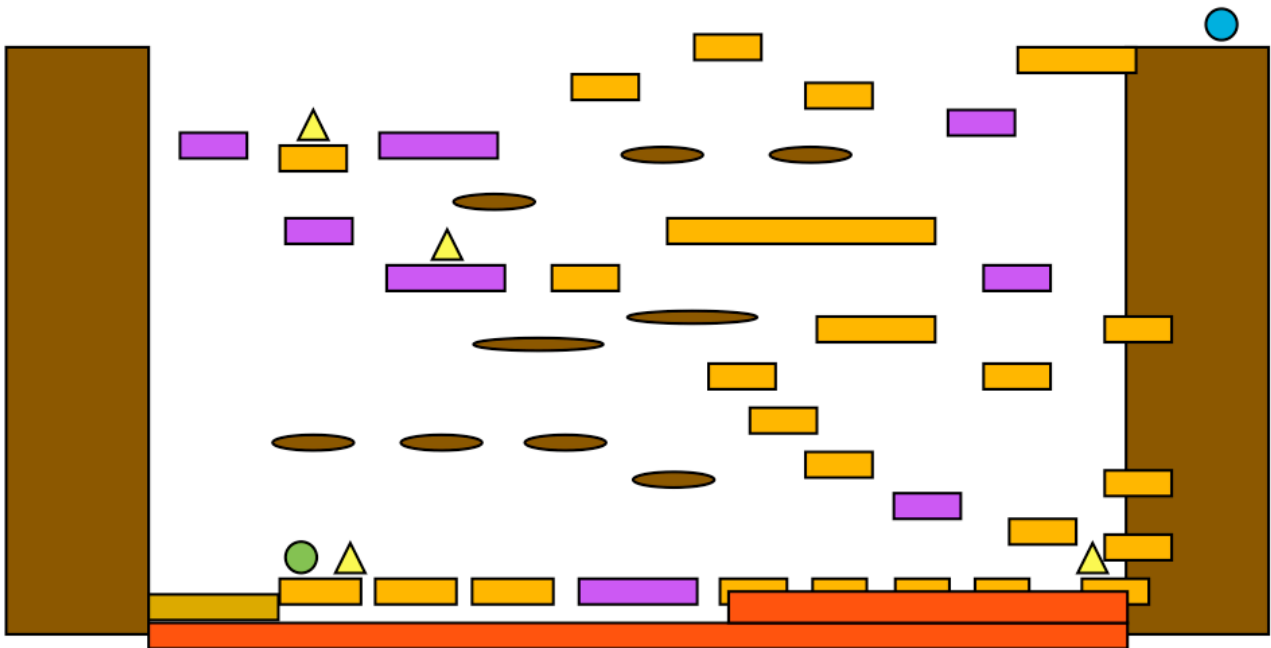









Figura 21. Esquema del primer nivel del videojuego



Figura 22. Captura de pantalla real del primer nivel del videojuego

Legenda

 Nube	 Comienzo nivel	 Zona de cajas
 Puertas del cielo	 Fin nivel	
 Lava	 Punto de guardado	

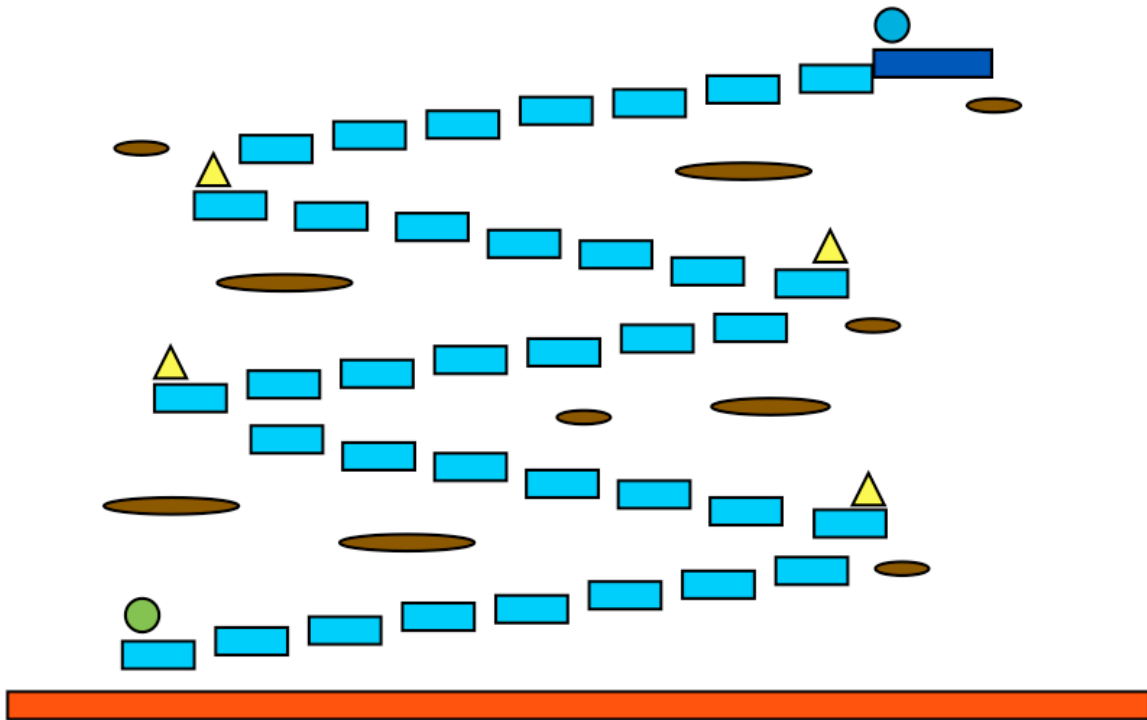


Figura 25. Esquema del tercer nivel del videojuego



Figura 26. Captura de pantalla real del tercer nivel del videojuego

2.2.3. Modelado de objetos en 3D

Es el proceso mediante el cual se han creado los modelos en 3D de los objetos que serán empleados en el juego. En los apartados previos se ha visto una pequeña muestra de los modelos realizados.

El modelado de los objetos del juego, incluidos los personajes, se ha realizado casi por completo mediante el uso de Blender (ver anexo C).

En la siguiente figura se presenta a 'Timan', el personaje principal del videojuego y el aspecto que presenta en el nivel 3, con sus alas negras.



Figura 27. Modelo del personaje principal para uno de los niveles del juego

Los enemigos se han mostrado en la figura 16; corazones, cofres, estrellas y checkpoints se han mostrado en las figuras 18, 19 y 20.

A continuación se muestran otras dos superficies mortales, la lava y el foso de pinchos:



Figura 28. Modelo de lava con textura animada



Figura 29. Modelo de foso con pinchos

En la figura 30, se muestra una imagen con ejemplos de las plataformas empleadas:

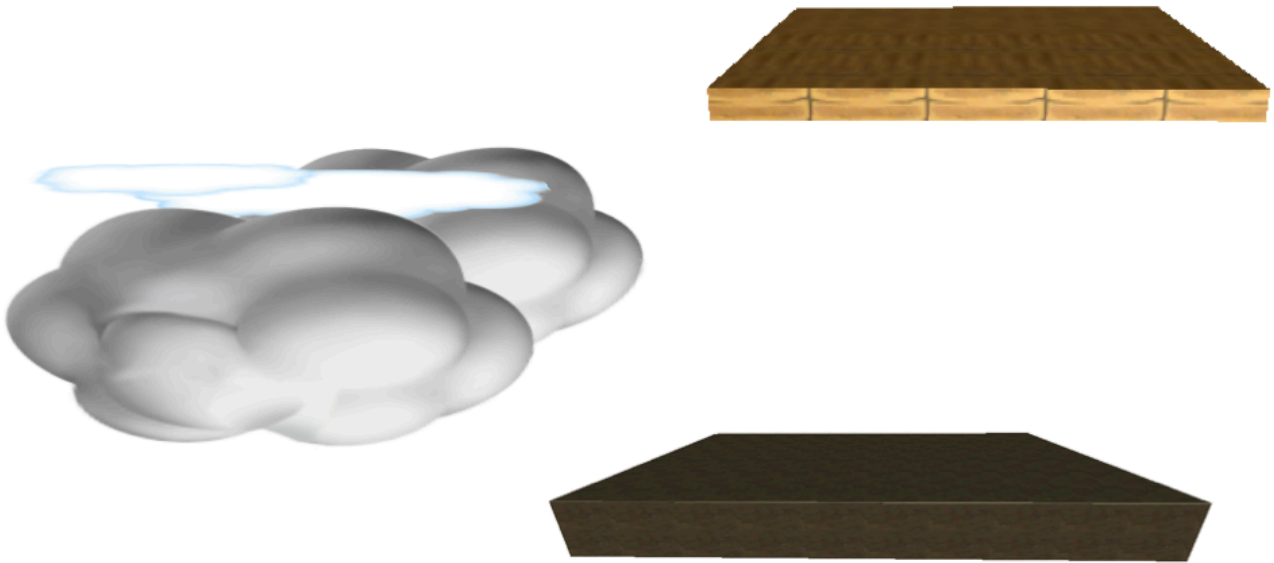


Figura 30. Modelos de las plataformas empleadas en los tres niveles

En el nivel tres se crean dinámicamente rayos durante la partida, el ejemplo de este modelo se muestra a continuación:



Figura 31. Modelo de rayo creado para el nivel 3

Otros elementos creados han servido para decorar y poblar las escenas, en la siguiente figura se muestra alguno de ellos:



Figura 32. Modelos de objetos empleados como decoración de los niveles

Un mayor ejemplo de los elementos creados, el proceso seguido para crearlos y las herramientas empleadas se muestra en el anexo I.

2.2.4. Implementación

Para la implementación del juego se ha hecho necesario crear las clases de cada uno de los elementos mencionados en el diseño del juego. Por ejemplo para el **personaje principal** se han creado las de la siguiente figura:

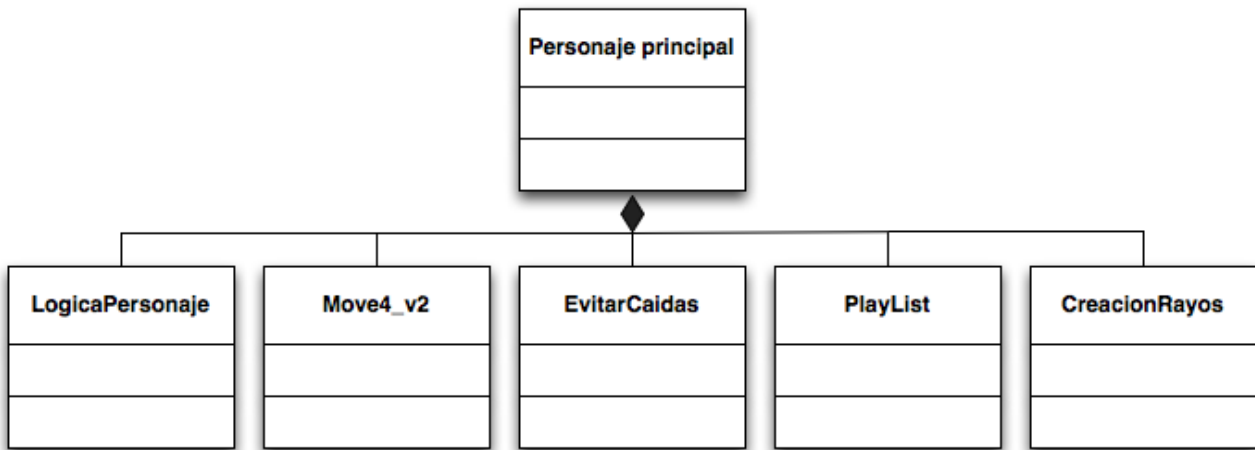


Figura 33. Diagrama de las clases empleadas en el personaje principal del videojuego

La clase principal es ‘LogicaPersonaje’, es la encargada de inicializar estructuras de datos para cada nivel, llevar un control de las vidas que se posee en cada momento y determinar en el nivel difícil si se ha llegado al final de partida (esto sucede cuando no quedan vidas), modificar la interfaz actualizando los datos que muestra, reproducir los efectos de sonido cuando se producen ciertos eventos, activar elementos del juego ocultos y cargar el siguiente nivel al terminal el actual.

El movimiento del personaje está controlado por la clase ‘Move4_v2’. Esta clase se encarga de controlar la interacción del usuario con el dispositivo y determinar en función de ella qué hacer (mover el personaje, pararlo o saltar). Es la encargada de aplicar la fuerza de la gravedad al personaje después de realizar un salto.

La clase que completa el movimiento del personaje es ‘EvitarCaidas’, esta clase lo que hace es evitar que el personaje caiga de superficies que están en movimiento.

Otra clase que forma parte del personaje principal es ‘Lista de reproducción’, es la encargada de reproducir la música del juego en orden aleatorio creando ambientación en las partidas.

La última clase ‘CreacionRayos’ forma parte del personaje en la última escena y es la encargada de generar rayos de forma aleatoria a lo largo del nivel.

La relación entre los objetos del juego se puede ver con diagramas de secuencia como el siguiente:

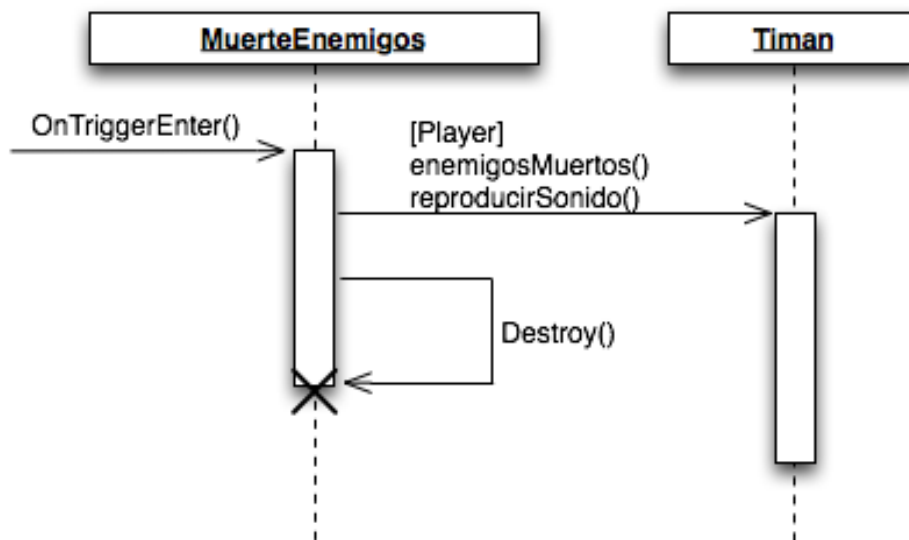


Figura 34. Diagrama de secuencia de la interacción entre enemigos y personaje principal

En el diagrama se muestra la interacción entre el personaje principal y un enemigo, cuyo resultado es la muerte del enemigo. Este caso se produce cuando el personaje cae encima del enemigo, el enemigo detecta la colisión al recibir la llamada de la función `OnTriggerEnter()`, reacciona a ella enviando dos mensajes al personaje principal en los que se le indica que aumente el número de enemigos muertos y que actualice la interfaz y que a continuación emita un sonido relacionado con la muerte del enemigo. Por último el enemigo se destruye y se elimina de la escena liberando recursos.

El resto de clases, su explicación y la interacción con el resto de elementos se encuentran en los anexos F.4.3 y F.4.4.

Otro de los aspectos que es necesario implementar es el tratamientos de datos entre niveles y almacenamiento de la información. Esto se ha logrado gracias a la implementación de la clase 'FinNivel' que fijado a un objeto del juego permite determinar si el jugador ha alcanzado la zona de cambio de nivel y gestionar toda la información. Esta clase se explica en el anexo F.4.4.

Por último comentar que en el anexo I se muestra el aspecto de cada uno de los niveles y otros aspectos de su creación, también se puede ver en formato vídeo en la dirección:

<http://ronroneos.com/PFC/Videos/t13v12.mp4>

2.2.5. Interfaz

Para explicar el diseño de la interfaz se pueden distinguir dos interfaces, una la de los menús que dan acceso a todas las partes del videojuego y la del propio videojuego.

La interfaz de los menús se resume en el siguiente diagrama:

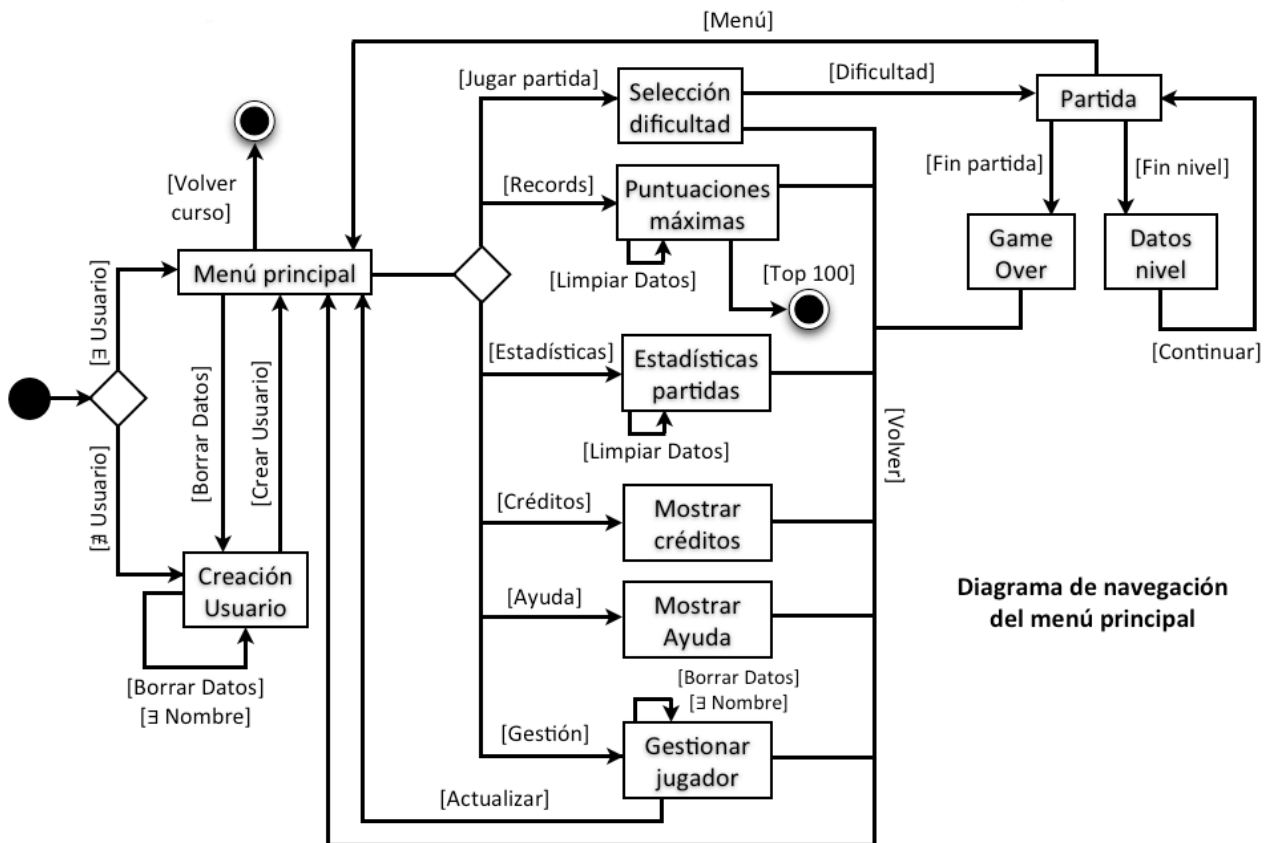


Figura 35. Diagrama de navegación del menú principal

Se ve como es necesario tener un usuario registrado para poder acceder al juego, si no existe se piden los datos para su creación. Se comprueba que no exista el nombre accediendo a la aplicación web, si no existe se crea el usuario y se accede al menú principal del juego.

Desde el menú principal del juego se accede al resto de opciones. Para comenzar una partida, se deberá seleccionar antes la dificultad. El diagrama con la jerarquía de ventanas se muestra en el anexo F.4.5.

En la figura 36 se muestra un ejemplo de la interfaz del menú principal:



Figura 36. Ejemplo de interfaz del menú principal

El aspecto de la interfaz y las ventanas de los menús se muestran en los anexos G.3 e I.

Una vez lanzada una partida, la interfaz del juego muestra el siguiente aspecto:



Figura 37. Interfaz del juego

Como se observa en la figura 37, en la interfaz se muestran las estrellas conseguidas respecto del total del nivel, los cofres recogidos respecto del total del nivel, el tiempo de partida, las vidas que quedan y el número de enemigos eliminados respecto del total de enemigos. En la parte inferior de la pantalla se encuentra el botón que permite acceder al menú principal.

Esta interfaz de juego se describe con más detalle en el los anexos G.3 e I.

2.3. Aplicación web

2.3.1. Introducción

La aplicación web del proyecto se ha creado para dar soporte al videojuego “La venganza de Timan” y mostrar un ejemplo de comunicación del sistema con el exterior (aparte de la reproducción de los vídeos). Surgió como

necesidad durante la fase de análisis del proyecto ya que se planteó como requisito la necesidad que se ha comentado de comunicar la aplicación (en este caso el videojuego) con el exterior y de almacenar fuera del dispositivo información referente a las partidas jugadas.

Esta aplicación web deberá dar de alta a un nuevo usuario con la restricción de que si existe se comunicará al jugador que el nombre elegido ya ha sido registrado debiendo elegir otro distinto.

Cuando el registro se ha realizado, se permite el uso del menú principal del juego. A partir de aquí, cada vez que un jugador termina una partida, el sistema hace una consulta sobre el resultado a la aplicación web, si el resultado está dentro del top 100, teniendo en cuenta su nivel de dificultad, se almacenarán los datos de la partida y del usuario en la base de datos de la aplicación web y se mostrarán en la página web principal de la aplicación los 100 mejores resultados por nivel de dificultad.

Otra opción que se puede plantear es que un usuario decida cambiar su información personal, en ese caso, el videojuego comunicará a la aplicación web la necesidad de cambiar esta información. Si entre los datos que se decide cambiar se encuentra el nombre de usuario, se volverá a comprobar que este no exista y si no existe se actualizarán aquellos datos que el usuario haya decidido modificar. En caso de haber cambiado el nombre de usuario se actualizarán todas las partidas ya registrados en la base de datos pudiendo comprobarse el cambio en la página web.

En los siguientes apartados se comentarán las decisiones de diseño tomadas durante el desarrollo de la aplicación web además de las funcionalidades extras de las que se dispone.

2.3.2. Decisiones de diseño

Entre las decisiones de diseño se encuentra el decidir qué información será necesaria almacenar en la base de datos para satisfacer las necesidades del videojuego descritas en el apartado anterior.

A continuación se muestra el diagrama entidad / relación creado durante la fase de diseño.

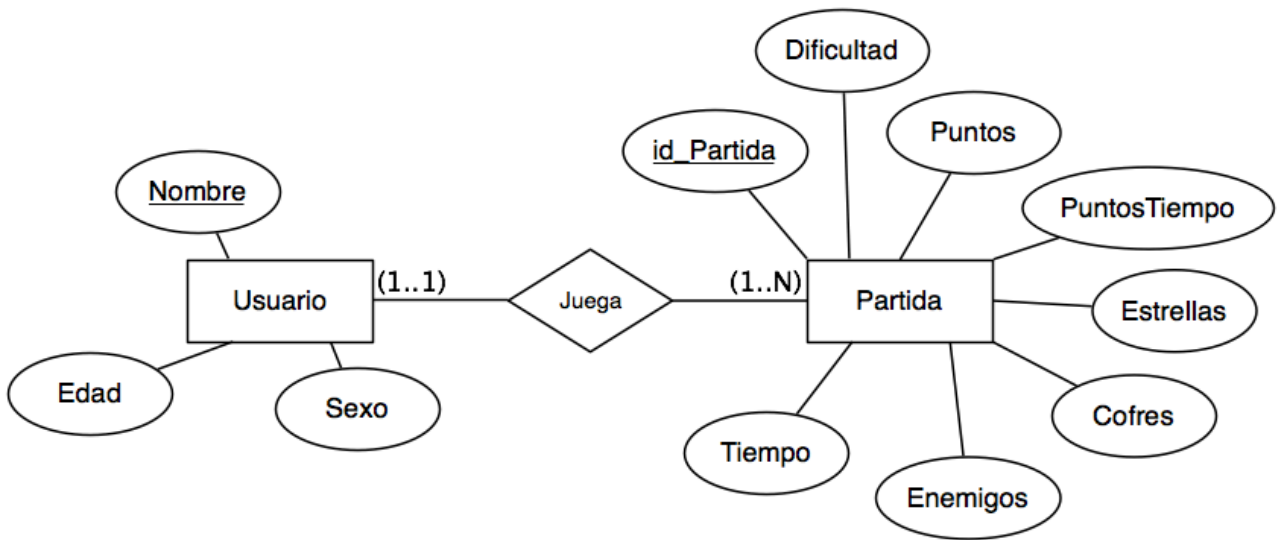


Figura 38. Diagrama Entidad / Relación de la aplicación web

Como se ha comentado en el apartado 1.4 el sistema gestor de la base de datos empleado ha sido MySQL ya que no se necesita una base de datos de gran tamaño, no está prevista una alta concurrencia en la modificación y en caso de darse un acceso masivo sería en lectura de datos (mostrar las puntuaciones).

Para la interacción con el gestor MySQL se ha utilizado PHP que junto con JavaScript y AJAX permiten crear páginas web dinámicas como la que se ha desarrollado. Con PHP y JavaScript se ha realizado la implementación de la web de gestión de puntuaciones y usuarios. Se han programado scripts para la consulta, creación y modificación de los usuarios del videojuego, así como scripts que procesan los datos de las partidas, con esto se ha creado un sistema de puntuaciones que se puede consultar por internet.

Para la eliminación de usuarios se ha tenido cuidado de borrar todas las partidas asociadas a un usuario determinado y no limitarse solamente a la eliminación de sus datos personales manteniendo así en todo momento la integridad de la información almacenada.

En la figura 39 se muestra mediante un diagrama de actividades la relación de comunicación entre el videojuego (iPad) y la aplicación web. Los ficheros creados en PHP para la comunicación desde el dispositivo iOS con la aplicación web reciben los datos mediante el método POST, estos pueden recibir como entradas:

► Consulta de un nombre de usuario (Comprobar Usuario): los datos recogidos por la aplicación (Solicitar Datos) se envían al servidor una vez que el jugador ha introducido su nombre y pulsa sobre el botón de creación de un nuevo usuario. Esto ocurre la primera vez que se juega al videojuego, una vez creado un usuario el sistema permite entrar al videojuego. Si no existe un usuario, no se permite acceder al menú principal del juego.

Otra consulta que se realiza es cuando un usuario ya está registrado y decide modificar sus datos (Modificar Usuario), si la modificación afecta también al nombre, se comprueba que esté libre y si lo está se modifican los datos. Esta opción también actualiza la tabla de puntuaciones que es la que se muestra por web. Si los cambios no afectan al nombre directamente se actualizan los datos sin necesidad de cambiar el nombre.

► Creación de un nuevo usuario (Crear Usuario): con la información del usuario (una vez verificado que no existe el nombre en la base de datos) recopilada desde el dispositivo iOS se crea una nueva entrada en la base de datos.

► El último tipo de comunicación videojuego - aplicación web es al terminar una partida. Al terminar una partida, se realiza una consulta a la aplicación (Comprobar Record) para determinar si es una nueva máxima puntuación (dentro del top 100) o no, si lo es se introduce esta nueva puntuación en la base de datos y se muestra vía web.

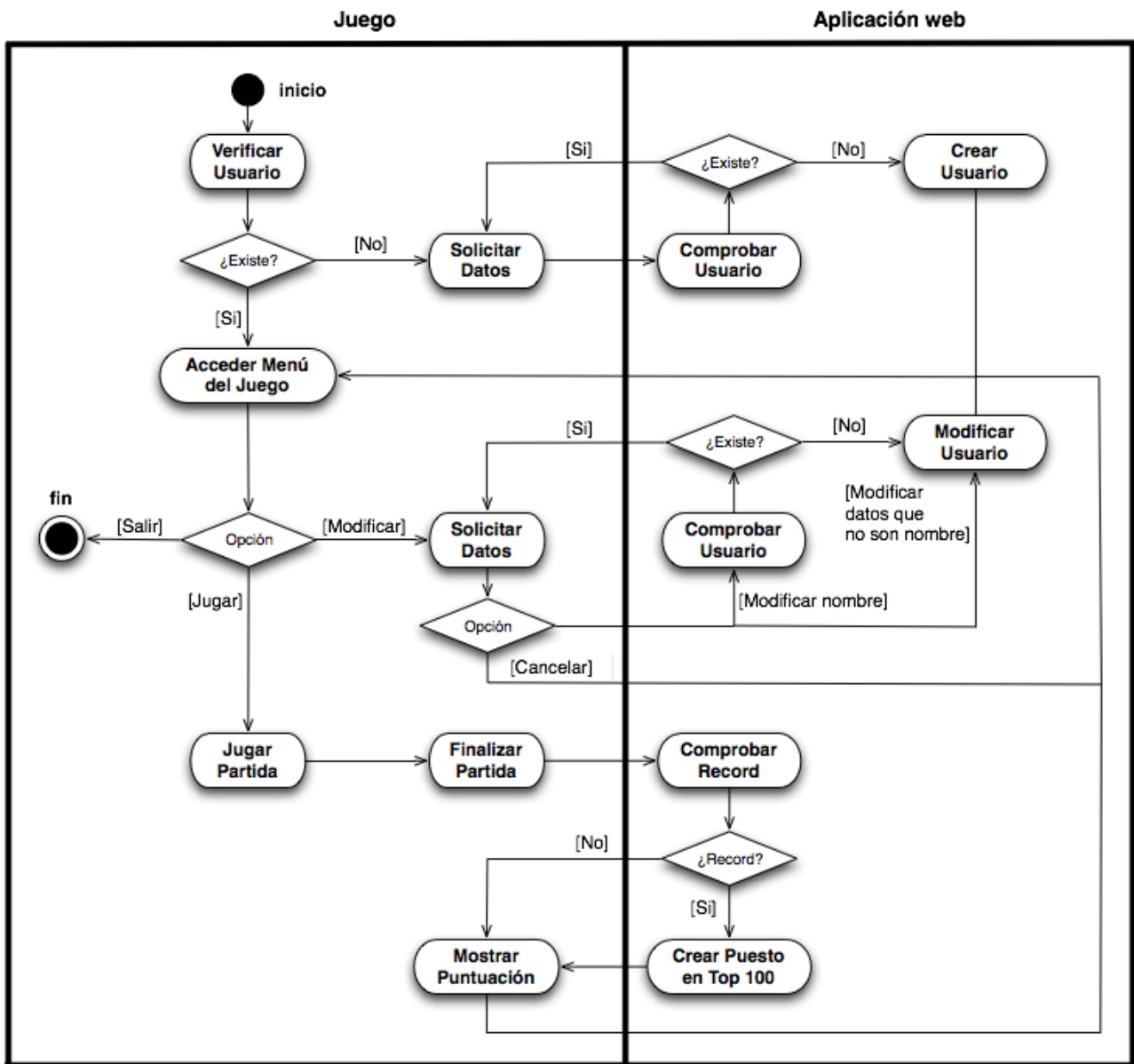


Figura 39. Diagrama de actividades del Juego - Aplicación web

2.3.3. Web de puntuaciones

Para simular una web que recopila información sobre partidas de un juego se ha creado un script en PHP que genera puntuaciones para el nivel normal y difícil del juego. De esta forma a la vez que se comprueba el diseño de la página se permite comprobar el funcionamiento de la misma.

Otro servicio que ya se ha comentado que ofrece la aplicación web, es el de mostrar la tabla de puntuaciones. Esta se puede observar en la captura de pantalla mostrada a continuación:

Timan's Revenge

Puntuaciones

TOP

100



Dificultad difícil

[Ver normal](#)

Selecciona un orden: ▾

Posición	Nombre	Puntos Totales	Puntos por Partida	Puntos por Tiempo	Estrellas	Cofres	Enemigos	Tiempo Partida
1	Zahira	5250	250	5000	3	0	2	0
2	Mar	1080	80	1000	6	0	2	0
2	Mar	1080	80	1000	6	0	2	0
4	Zahira	1050	50	1000	3	0	2	0
4	Zahira	1050	50	1000	3	0	2	0
4	Pabel	1050	50	1000	3	0	2	0

Figura 40. Captura de pantalla de la página de resultados máximos

Como se ve en la figura 40, en la página de resultados máximos se muestran los datos seleccionados como más relevantes de una partida jugada que son los almacenados en la base de datos.

Por defecto se muestran los datos ordenados por los puntos totales obtenidos, pero se pueden ordenar tomando como criterio de ordenación cualquiera de las otras columnas. Cualquier reordenación que se haga una vez cargada la página no provoca que esta se vuelva a cargar ya que se ha hecho uso de la tecnología AJAX y solo se actualiza dinámicamente el contenido de la tabla.

Timan's Revenge
Puntuaciones

TOP
100

Dificultad difícil [Ver normal](#)

Estrellas

Posición	Nombre	Puntos Totales	Puntos por Partida	Puntos por Tiempo	Estrellas	Cofres	Enemigos	Tiempo Partida
1	Usuario26	566	276	290	20	1	3	742
1	Usuario28	427	205	222	20	5	11	1889
3	Usuario23	306	152	154	19	2	11	959
3	Usuario3	470	234	236	19	0	17	635
3	Usuario24	407	221	186	19	0	14	547
6	Usuario7	329	158	171	18	2	9	1568

Figura 41. Captura de pantalla de resultados máximos ordenados por el total de estrellas

En la figura 41 se muestra la ordenación de la tabla por el número de estrellas conseguidas.

También se puede observar en la captura de pantalla de la figura 41 que las puntuaciones están agrupadas según el nivel de dificultad. En la imagen se muestra la dificultad nivel difícil pero pulsando sobre el botón “Ver normal” se muestran los datos para la dificultad normal.

La tecnología AJAX se menciona en el E.1.

2.4. Resto de elementos de la aplicación

En este apartado se comentará resumidamente el resto de elementos que forman parte de la aplicación. El hecho de englobarlos todos en un único apartado y mencionarlos resumidamente en esta parte de la memoria no debe restarles la importancia que se merecen, ya que el ponerlos en un mismo apartado es debido a la limitación en extensión de la memoria. La mayor parte de estos elementos se han ampliado en el anexo del PFC.

2.4.1. Textos del curso

Los textos de los temas forman la parte fundamental de los conceptos que se desean explicar en el curso. Estos textos han sido preparados cuidadosamente por el autor del presente documento de forma que cumpliesen la característica de ser progresivos en cuanto a la enseñanza y dificultad, es decir, comenzar por los aspectos fundamentales de la programación de videojuegos y del motor gráfico hasta terminar con aspectos avanzados de la programación de los mismos.

Se han incluido gran cantidad de ejemplos para ayudar a entender y asimilar las explicaciones de cada tema.

Se ha revisado gran cantidad de documentación sobre programación de videojuegos con Unity 3D y de otras temáticas que no tenían relación directa con el curso para lograr una mejor explicación de los temas.

En esta parte del trabajo se han invertido gran cantidad de horas para crear la documentación, adaptarla al formato del curso y darle el aspecto deseado para que la información ofrecida resultase sencilla de entender.

El fundamento seguido para la creación de la documentación y de su explicación es que fuese lo más visual posible.

Se han elaborado trece temas con un total de 1002 páginas a una resolución de 1024 x 768 pixeles. En el anexo I se ha incluido un ejemplo de capítulo del curso.

2.4.2. Vídeos

Los vídeos del curso son ejemplos de aplicaciones creadas en Unity siguiendo las explicaciones de los temas. Estos ejemplos se han creado en

formato vídeo y no de aplicación debido a que a la altura del curso en el que han sido creados no era posible utilizarlos como aplicación (por ejemplo por la necesidad de usar un teclado físico) o porque mediante imágenes se conseguía explicar mejor ciertos aspectos del curso.

En el anexo F.6 se describen los vídeos grabados para el curso.

Estos vídeos como se ha comentado, han sido alojados en un servidor web, de forma que se puedan reproducir por streaming desde el reproductor de la aplicación.

2.4.3. Aplicaciones interactivas

Por último se ha creado un conjunto de aplicaciones interactivas (aparte del juego) para ser ejecutadas durante la explicación de los temas. Se han creado con la intención de mostrar la capacidad de programación de videojuegos y de contenido multimedia de Unity 3D.

Pretenden ser como se ha comentado en otros apartados un complemento a los textos y a los vídeos de ejemplo.

No se ha abusado en cuanto al número de aplicaciones interactivas puesto que se quería dar la máxima importancia al ejemplo de videojuego completo, pero sí se quería mostrar sobre todo en los primeros temas la interactividad que se desea tenga el curso.

Se van a comentar un par de aplicaciones interactivas del curso, en el anexo F.7 se indica el resto de ejemplos de aplicaciones interactivas creadas y sus objetivos.

La primera aplicación interactiva que se va a mostrar (que es la primera que se muestra en el curso) es un 'Hola Mundo' en 3D. Con esta aplicación se muestra al alumno cómo crear un nuevo proyecto, crear una escena, añadir elementos a una escena, añadirles texturas creadas, posicionar la cámara, añadir iluminación a la escena, redimensionar y posicionar elementos, a programar un script, a asignar componentes a los objetos y a probar una aplicación haciendo uso del entorno y de un dispositivo iOS. Los resultados se muestran en la siguiente figura:



Figura 42. Hola Mundo 3D. Ejemplo de aplicación interactiva

En la figura 42 se ve el resultado de la aplicación, gracias al script creado y asignado a la esfera que tiene asignada la textura del mundo, al ser ejecutada el mundo interacciona con el usuario. Cuando se gira el dispositivo, gira el mundo y al inclinarlo, se acerca o se aleja. También se ha añadido un botón que permite volver al curso.

Otro ejemplo de aplicación interactiva se muestra en la figura 43. Esta aplicación es un ejemplo que recoge la creación de los distintos puntos de unión (joints) entre objetos. Esto permite crear un objeto que está formado por un conjunto de otros, que reaccionan como una única entidad. Para mostrar el efecto de creación de joints la aplicación permite aplicar fuerzas positivas o negativas en el eje X de la bola y del plano. Así el ejemplo aparte de mostrar los joints sirve de ejemplo de creación de física y de fuerzas sobre los objetos.

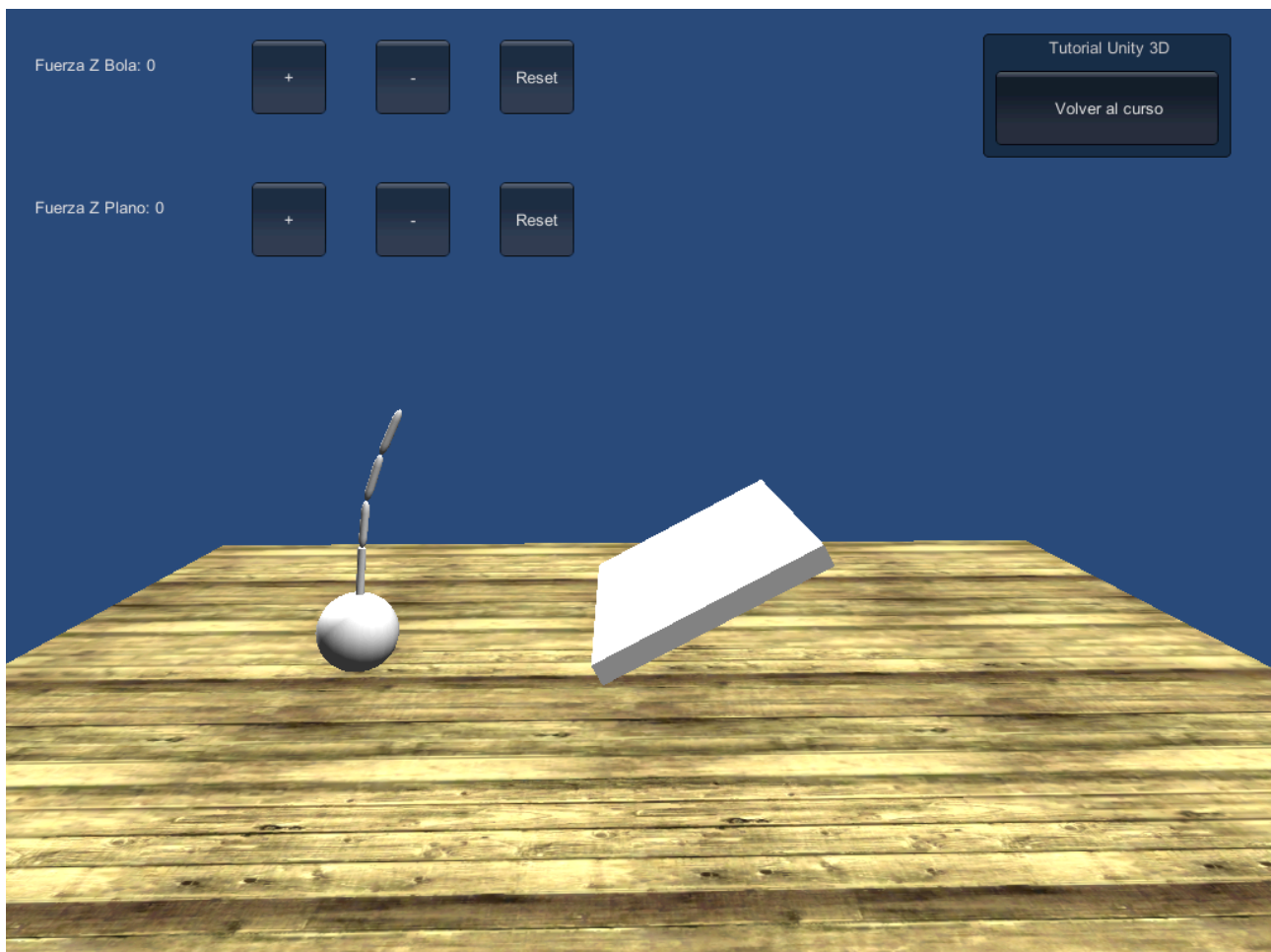


Figura 43. Ejemplo de aplicación de fuerzas sobre objetos

2.5. Pruebas

En este punto de la memoria se analiza el proceso de pruebas llevado a cabo para la verificación y validación de todos los elementos del sistema.

El objetivo principal es el de asegurar el cumplimiento de los objetivos fijados dentro del marco del PFC y asegurar el funcionamiento correcto de la aplicación en su conjunto.

El plan de pruebas diseñado ha consistido en:

- ▶ Pruebas de caja blanca que se encarguen de realizar pruebas estructurales para la verificación del correcto funcionamiento.
- ▶ Pruebas de caja negra que verifique la entrada / salida de datos.

- ▶ Pruebas de inspección de código basadas en los errores más comunes en la programación de aplicaciones informáticas.
- ▶ Se han realizado pruebas unitarias por cada parte principal del sistema por separado.
- ▶ Una vez realizadas las pruebas unitarias se han realizado pruebas de integración y de sistema para verificar el correcto funcionamiento del sistema en su conjunto.
- ▶ Por último se han realizado pruebas de aceptación utilizando la aplicación instalada en un dispositivo real.

La descripción de las pruebas se muestran en el anexo F.5.

3. Conclusiones

3.1. Cumplimiento de objetivos

El objetivo principal del proyecto era la creación de un curso de programación de videojuegos en Unity 3D para iPad. Para la consecución de este objetivo el proyecto se dividió en cuatro partes.

La primera parte consistía en implementar un visor de documentos que presentase como característica de uso el reaccionar a interacciones del usuario con el dispositivo iOS. Estas interacciones consistían en navegar por el documento mediante pulsaciones en la pantalla, activar y desactivar el zoom, disponer de un menú de opciones que permitiese acceder a los índices del curso y a la ayuda, reproducir los vídeos de ejemplo, ejecutar las aplicaciones interactivas y el videojuego completo de ejemplo. Este visor actuaría como enlace entre todas las partes de la aplicación.

La segunda parte consistía en un videojuego completo de ejemplo, que sirviese como ejemplo de aplicación de lo que se puede lograr con lo explicado en todos los temas escritos. Además debía de cumplir unos requisitos mínimos como: que tuviese tres niveles o escenas, que presentase dos niveles de dificultad, que el personaje respondiese a pulsaciones en la pantalla, que se almacenasen localmente datos de las partidas jugadas y que permitiese la creación de un usuario y posterior gestión de su información. También debía poder borrarse toda la información almacenada y sobre todo y fundamental se debía poder volver al visor de la primera parte.

La tercera parte consistía en una aplicación web que debía guardar los datos de los usuarios creados en el videojuego así como crear un top 100 de las mejores partidas jugadas en dificultad normal y difícil. Esta aplicación serviría también para controlar la gestión de los usuarios e impedir que se crease un usuario ya existente en la base de datos. Esta parte serviría como ejemplo de interacción entre la aplicación (o una parte de ella) con el exterior. Además se debía poder consultar el top 100 por internet a través de una página web.

La cuarta parte consistía en la creación de los ejemplos en vídeo y en formato de aplicación interactiva que complementarían las explicaciones del

curso. Y por último en esta cuarta parte también se crearía toda la documentación del curso que sería la base fundamental del mismo.

Estos requisitos son de obligado cumplimiento para considerar el PFC como terminado, al cumplirse satisfactoriamente se puede concluir que el objetivo principal del proyecto ha sido alcanzado con éxito.

3.2. Mejoras y ampliaciones

En cuanto a las mejoras, se podría realizar una implementación que fuese independiente de la resolución del dispositivo y que se adaptase a la resolución del dispositivo que tuviese cada usuario.

Otra mejora podría ser la de lanzar el curso para un mayor número de plataformas destino, como Android y aplicación de escritorio (Windows, Mac, Linux).

Pensando en futuros cursos y continuando con el presente curso realizado, se podría ampliar la aplicación introduciendo nuevos temas más avanzados. Esta característica se puede lograr sin muchas complicaciones debido al diseño modular realizado para la aplicación. Se podría reutilizar todo el código y con ligeras modificaciones se podría añadir nuevos temas, vídeos y aplicaciones interactivas.

Otras ampliaciones podrían ser la de sacrificar la simplicidad de la interfaz introduciendo nuevas funcionalidades típicas de otros visores como por ejemplo el saltar a una determinada página de los documentos de texto.

Otra ampliación podría ser la de introducir ejercicios que tuviese que realizar un alumno y que el sistema pudiese corregir y evaluar.

Crear un sistema que simulase un buzón de preguntas que pudiese realizar el alumno al profesor directamente desde la aplicación.

3.3. Perspectivas de futuro del proyecto

Las perspectivas de futuro del proyecto son amplias, la industria de los videojuegos mueve miles de millones al año, el sector demanda cada vez más especialistas en desarrollo de videojuegos y la introducción de la tecnología en forma de dispositivos móviles en la vida cotidiana de las

personas impulsa un consumo cada vez mayor de videojuegos. Esto hace que cada día haya más personas interesadas en formarse como desarrolladores de videojuegos.

Estamos en una era en la que las personas demandan cada vez más formación e información gracias a la disponibilidad de la tecnología. El curso al estar realizado en formato aplicación puede tener una doble vertiente, una que sirva como medio a un profesor para la impartición de un curso y otra vía que sirva a un alumno como libro digital interactivo para seguir el curso o incluso autoformarse. La aplicación se podrá colgar para ser comprada en los mercados de aplicaciones por lo que satisfaría las necesidades recién comentadas.

Además Unity comenzó una revolución dentro del sector gracias a la licencia gratuita de una de sus versiones, que permite desarrollar contenido a coste cero (respecto al motor gráfico). Gracias a su continua evolución y sus acuerdos comerciales Unity tiene cada día más peso dentro de la comunidad de desarrollo de videojuegos.

Por todo esto el potencial de la aplicación es alto.

3.4. Valoración personal

Dentro del mundo de los videojuegos mi experiencia había sido la de estar al otro lado de la ventana, es decir, la de ser una persona que disfrutase de la experiencia aportada por un videojuego.

Durante los años de carrera, el contacto con el desarrollo de videojuegos había sido de pasada, inteligencia artificial, diseño gráfico, modelado visual y animación ...

El desarrollo del proyecto me ha servido para afianzar y ampliar conocimientos adquiridos durante la carrera, lo cual valoro muy positivamente.

También me ha introducido de lleno en el proceso de creación de un videojuego, tener en cuenta todas las fases que implica su desarrollo, experimentar con herramientas que desconocía y sobre todo valorar la complejidad y el esfuerzo que tiene desarrollar un buen juego.

A nivel laboral, debido a la creciente demanda dentro de la industria de los videojuegos, el proyecto me abre importantes posibilidades y no descarto el dedicarme en un futuro no muy lejano a su desarrollo.

Con la realización de este proyecto mi visión del mundo de los videojuegos ha cambiado totalmente. He aprendido a valorar mucho más el trabajo que hay detrás de cada videojuego y puedo afirmar que me gusta mucho más formar parte del desarrollo de un videojuego que jugarlo.

4. Bibliografía

4.1. Libros

- (1) Smith, C. (2010) "Game Development with Unity. En: More iPhone Cool Projects". Ed: Apress.
- (2) Wiebe, R. (2011) "Unity iOS Essentials. Develop high performance, fun iOS games using Unity 3D". Ed: Packt Publishing.
- (3) Pierce, G. (2012) "Unity iOS Game Development. Develop iOS games from concept to cash flow using Unity". Ed: Packt Publishing.
- (4) Cellary, W. y Walczak, K. (2012) "Interactive 3D Multimedia Content. Models for Creation, Management, Search and Presentation". Ed: Springer.
- (5) Gerasimov, V. y Kraczla, D. (2012) "Unity 3.x Scripting. Write efficient, reusable scripts to build custom characters, game environments, and control enemy AI in your Unity game". Ed: Packt Publishing.
- (6) Watkins, A. (2011) "Creating Games with Unity and Maya. How to Develop Fun and Marketable 3D Games". Ed: Focal Press.
- (7) Mullen, T. (2007) "Introducing Character Animation with Blender". Ed: Wiley.
- (8) Goldstone, W. (2009) "Unity Game Development Essentials". Ed: Packt Publishing.
- (9) Blackman, S. (2011) "Beginning 3D Game Development with Unity: The World's Most Widely Used Multiplatform Game Engine". Ed: Apress
- (10) McDermott, W. (2011) "Creating 3D Game Art for the iPhone with Unity: Featuring modo and Blender pipelines". Ed: Focal Press.

4.2. Artículos

- (11) McAllister, G. (2012) "Introduction to scripting with Unity. Unity Tutorials".

4.3. Proyectos Fin de Carrera

(12) Gutiérrez Ayuso, D. (2012) “Implementación de catálogo virtual mediante realidad aumentada en dispositivos móviles”.

(13) Martínez Millá, D. (2012) “Interface de usuario multimodal asistido con agente virtual”.

Anexos

Anexo A

Estudios preliminares

Para el PFC se han realizado varios estudios de la situación del mercado de forma que cubran las distintas perspectivas que se pueden tener sobre la aplicación final realizada.

- ▶ El primer estudio que se ha hecho es el de la situación de la industria de los videojuegos. Es importante centrar la importancia del contexto global en el que se desarrolla el proyecto.
- ▶ Un segundo estudio es el de la importancia de los dispositivos iOS en distintos tipos de mercado y comparativas respecto a dispositivos Android.
- ▶ Por último se ha realizado un estudio de la importancia de Unity 3D como motor gráfico en el desarrollo de videojuegos.

A.1. Situación de la industria de los videojuegos

Como se observa en la figura A.1, desde 2008 hasta 2012, las ventas de videojuegos han crecido hasta alcanzar un 11,75% de tasa de crecimiento anual compuesto¹ (TCAC) y se espera alcancen los 86.900 millones de dólares en 2017. Se prevé que desde 2013 hasta 2017 el TCAC sea del 7,7%. En porcentaje, supusieron en 2012 el 14% de ventas de videojuegos para móviles respecto del total de ventas de videojuegos y se estima que alcancen el 15% a finales de 2013. Este porcentaje se estima que siga creciendo hasta el 17% en 2017 lo que muestra que se mantendrá la importancia del sector móvil dentro de los videojuegos.

¹ La tasa de crecimiento anual compuesto es un término específico de negocios e inversión empleado para determinar la ganancia anual. Se utiliza para describir el crecimiento sobre un periodo de tiempo.

² Fuente de los datos: *Pricewaterhouse Coopers*.

“Los videojuegos es uno de los pocos mercados de los medios de comunicación y del entretenimiento en los que Japón es el líder incluso por encima de China. Con un TCAC de un 7%, el mercado de videojuegos en Japón alcanzará un valor de 13.700 millones de dólares en 2017.

Existen una serie de países con un mercado de los videojuegos emergente, como Nigeria (22% TCAC), Kenya (20% TCAC), India (18% TCAC) y Vietnam (15% TCAC).

En Europa, Rusia es el país que domina en cuanto a valor de mercado de la industria de los videojuegos y se prevé que en 2017 sea el octavo país del mundo con el mercado más importante. Dentro de Europa, otros países que presentan muestras de crecimiento importante en el sector son la República Checa, Hungría y Polonia.

Se estima que el sector de móviles sea el de crecimiento más rápido para los próximos cinco años, pasando de 8.800 millones de dólares de ingresos en EE.UU. en 2012 a los 14.400 millones en 2017 con un TCAC del 10% por venta de smartphones dedicados al entretenimiento (figura A.1)”.

Ingresos globales (10¹⁰ dólares) de videojuegos para móviles y proporción (%) respecto del total del segmento de videojuegos 2008 - 2017

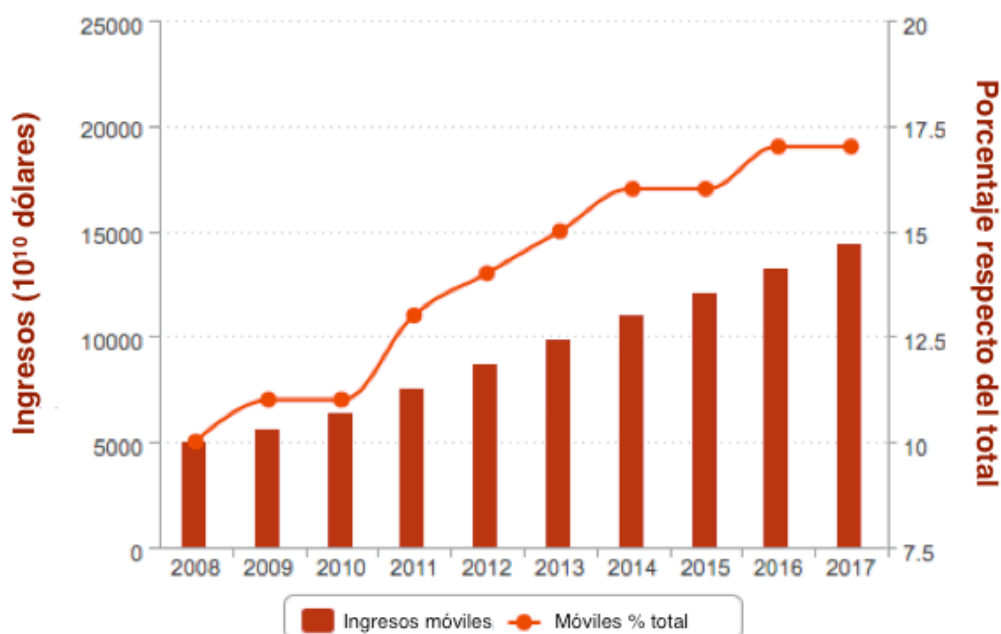


Figura A.1. Ingresos globales en móviles (10¹⁰ dólares) y participación del total del segmento de videojuegos (%) de 2008 a 2017

Otra muestra de la importancia de la industria de los videojuegos se observa en el mercado de las consolas indicado por la figura A.2. Se prevé que los consumidores incrementen el consumo de consolas de videojuegos con un TCAC del 4,6% pasando de 24.900 millones de dólares en 2012 a 31.200 millones en 2017. Este aumento se cree que será debido al interés que se ha suscitado por la anunciada salida de la PlayStation 4 de Sony y los rumores de una nueva Xbox.

Ingresos por venta de consolas por zonas (10¹⁰ dólares) 2011-2017

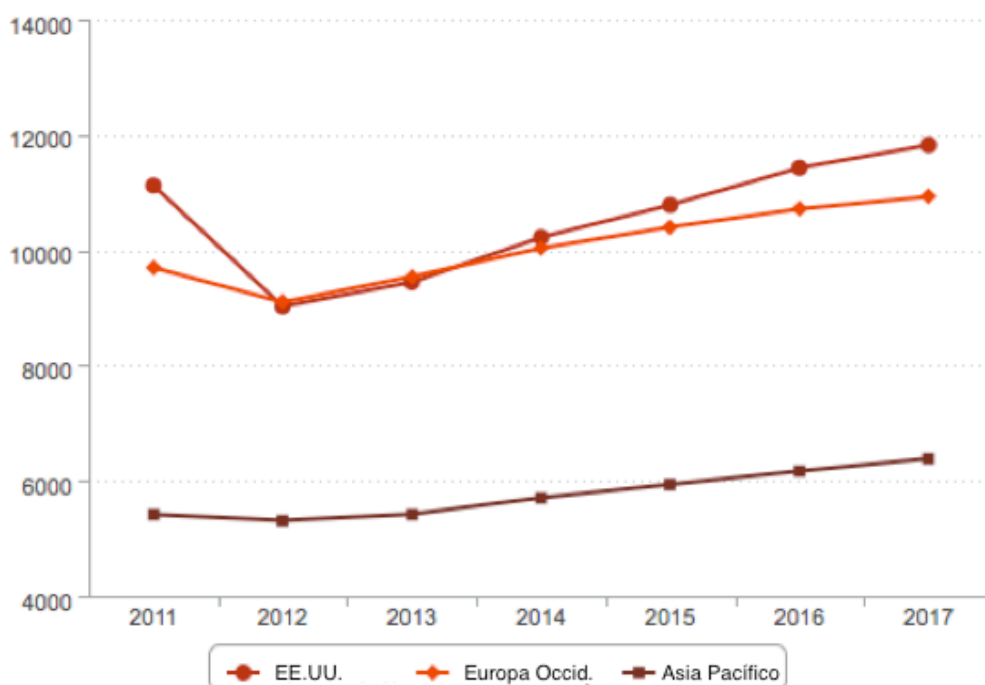


Figura A.2. Ingresos por venta de consolas por zonas (10¹⁰ dólares) de 2011 a 2017

También se puede observar como el mercado de consolas en los EE.UU. superará al europeo a partir de 2013, convirtiéndose en el principal mercado de consumo de consolas.

“En 2017 en muchos segmentos de los medios de comunicación y el entretenimiento China estará por encima de Japón en importancia de mercado; en algunos de ellos ya está por delante. Pero en el caso de los videojuegos, como se muestra en la figura A.3 no lo está, se prevé que Japón tendrá un volumen de mercado de 13.700 millones de dólares, situándose detrás de EE.UU. con 18.200 millones pero por encima de China con 11.400 millones”.

Ingresos totales por videojuegos en los tres principales mercados (10¹⁰ dólares) 2008-2017

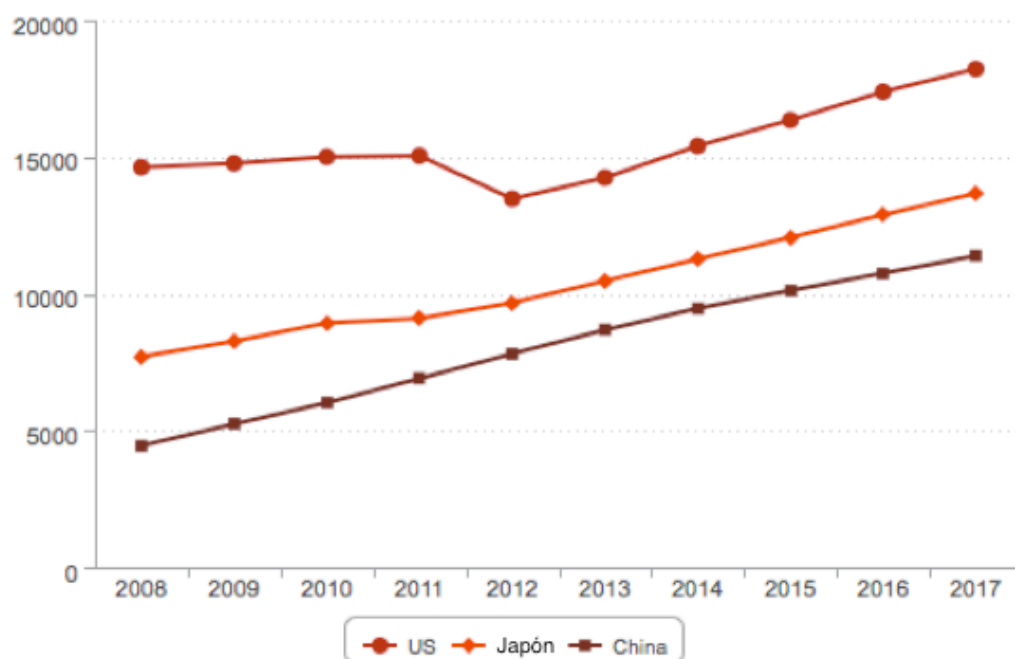


Figura A.3. Ingresos por venta de videojuegos en los tres mercados más importantes (10¹⁰ dólares) de 2008 a 2017

En la figura A.4 se observa el tipo de consumos que se prevé realicen los consumidores hasta 2017. Se ve como el consumo preferente de los usuarios es el de consolas y se prevé que lo seguirá siendo hasta 2017.

En 2017 los juegos en línea casi alcanzarán a las ventas de consolas, y es que este mercado se está convirtiendo poco a poco en uno de los más importantes. La mayoría de juegos, por no decir todos tienen un modo de juego en línea, eso es debido a que en general a los jugadores de videojuegos les gusta más interactuar con otras personas que solo con la máquina.

El mercado de móviles también irá en aumento, debido principalmente al avance tecnológico de este tipo de dispositivos que permiten ejecutar aplicaciones cada vez más parecidas a las que se pueden encontrar en las consolas.

Por último comentar que la venta de PC's se mantendrá estancada, pero los consumidores no la abandonarán.

La tendencia general de los usuarios es de pasar de ‘pagar para poseer’ a ‘pagar por jugar’.

Gasto global por usuario por tipo (10¹⁰ dólares) 2008-2017

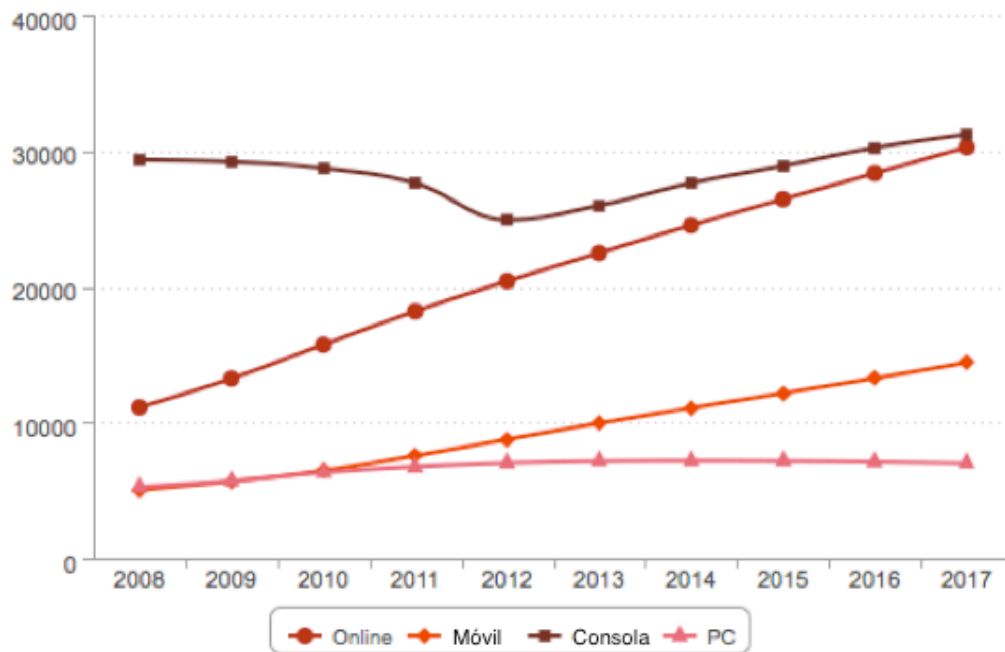


Figura A.4. Gasto global de los usuarios por tipo de dispositivo (10¹⁰ dólares) de 2008 a 2017

En la figura A.5 se ve como las ventas en línea se incrementarán un 8% de media por año durante los cinco años próximos. Para 2017, las plataformas de juego en línea alcanzarán la paridad con la venta de consolas.

Para 2017 se prevé que por cada 97 dólares que se gasten en juegos en línea se gastará 100 en consolas.

Ingresos globales por juegos online (10¹⁰ dólares) y proporción respecto del total del segmento de videojuegos (%) 2008 - 2017

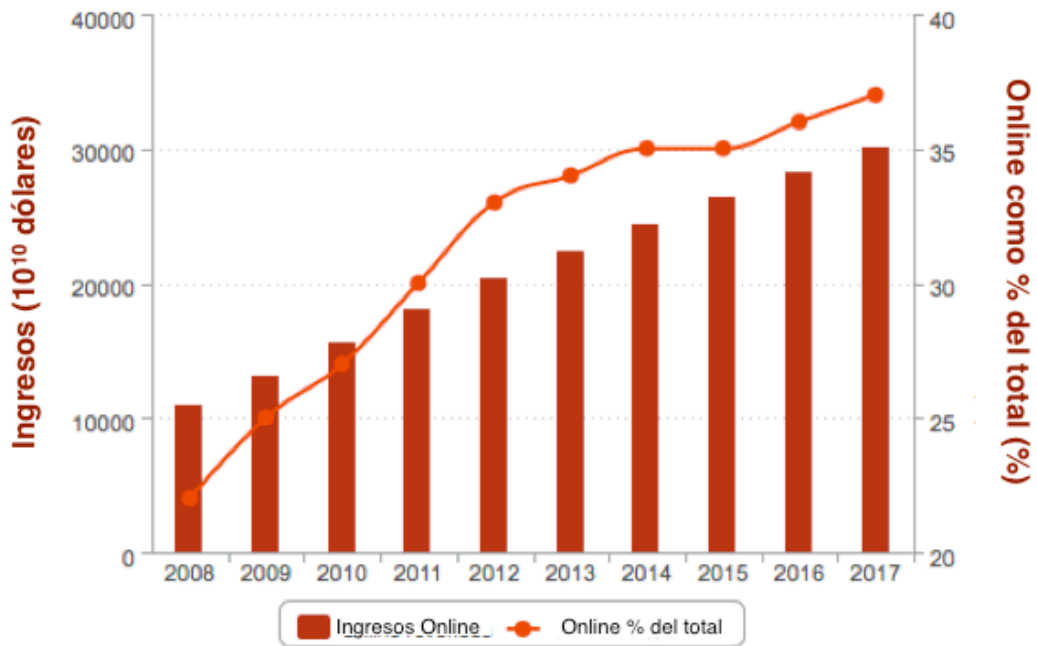


Figura A.5. Total de ingresos por juegos en línea (10¹⁰ dólares) y participación del total del segmento de videojuegos (%) de 2008 a 2017

Ingresos de la industria de los videojuegos en U.S. (10⁹ dólares) 2012

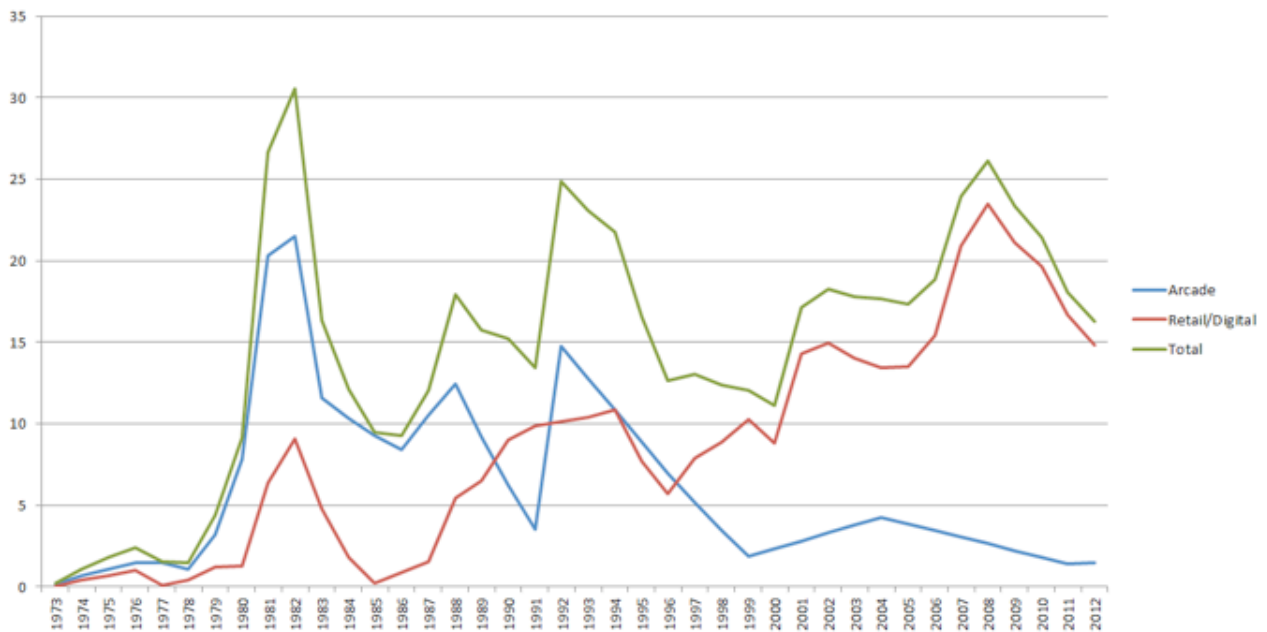


Figura A.6. Ingresos totales (10⁹ dólares) de la industria de los videojuegos en EE.UU. (2012)

En la figura A.6 se muestra un estudio de los ingresos de la industria de videojuegos en el que se compara el sector arcade³ de juegos (línea azul) respecto a consolas, PDA, PC, móviles y tabletas (línea roja). En línea verde se muestra el total de ingresos. Se puede ver la tendencia de consumo y uso de los usuarios.

También se puede ver que el sector arcade estuvo dominando el mercado hasta 1992, a partir de ahí, con las nuevas tecnologías, aparición de PC y consolas y posteriormente móviles y tabletas, su uso ha caído sin acabar de desaparecer, estancándose en menos de 2.000 millones de dólares en 2012.

Otros estudios realizados por la Asociación de Software del Entretenimiento (ESA) muestran los siguientes datos:

¿Quién juega videojuegos?

- ▶ El 58% de los americanos (EE.UU.) juegan a videojuegos.
- ▶ De media hay dos personas que juegan habitualmente a videojuegos en los hogares americanos (EE.UU.).
- ▶ De media, existe una consola, PC o smartphome dedicado a videojuegos.
- ▶ La edad media de los jugadores es de 30 años repartiéndose la edad de jugadores de la siguiente forma: 32% tiene menos de 18 años, 32% tiene entre 18-35 años y el 36% tiene más de 36 años.
- ▶ En cuanto al sexo de los jugadores, el 55% son hombres y el 45% mujeres. Las mujeres mayores de 18 años representan gran parte de la población que juega a videojuegos siendo el 31%. En cuanto a hombres, los menores de 17 años representan el 19% de la población jugadora.

¿Quién compra videojuegos y computadoras?

- ▶ La media de edad del comprador habitual es de 35 años.

³ **Arcade** es el término genérico de las **máquinas recreativas** de videojuegos disponibles en lugares públicos de diversión, centros comerciales, restaurantes, bares, o salones recreativos especializados.

- ▶ De los compradores habituales, el 54% son hombres y el 46% mujeres.

¿Cómo se juega?

- ▶ En cuanto al tipo de juegos que se tienden a comprar en consolas, en la figura A.7 se observan los videojuegos por género que más se venden:

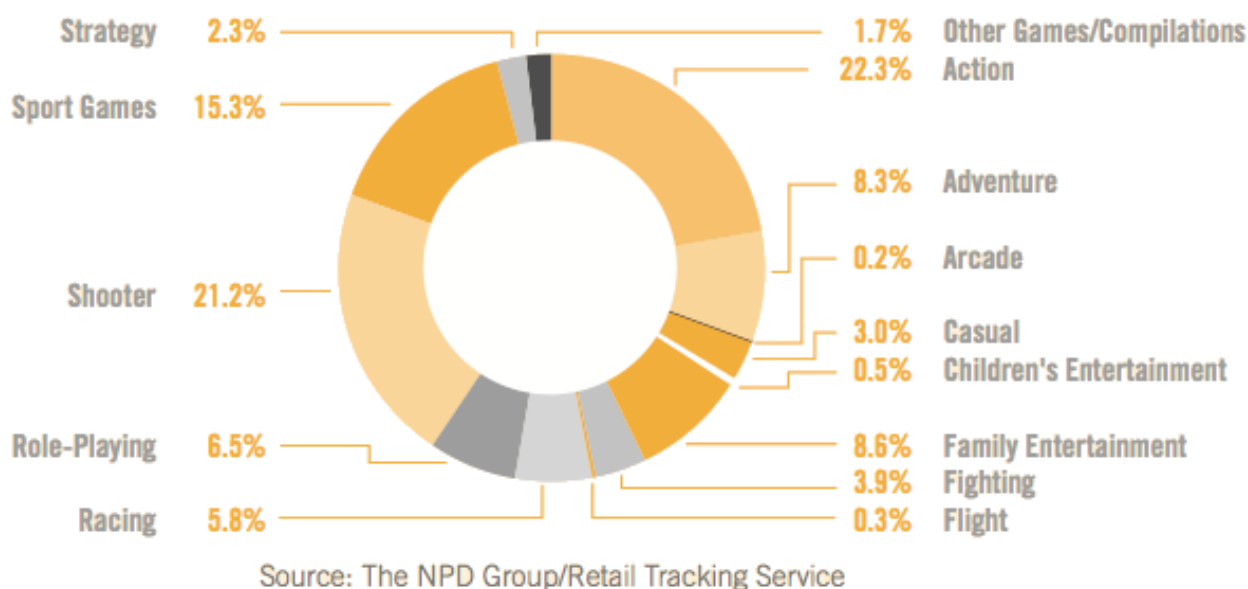


Figura A.7. Videojuegos más vendidos por género en consolas en unidades (2012)

Los tipos de juegos más vendidos son los de acción, seguidos cercanamente por los de disparos (shooter). Otro género que también se vende bastante es el de deportes.

En un término medio de ventas se encuentran los de rol, carreras, aventuras y entretenimiento familiar.

Entre los que generan menos interés se encuentran los ocasionales, arcades, los de entretenimiento de niños, vuelo, estrategia y lucha.

En la siguiente figura, se muestra el mismo gráfico pero para los juegos de computadoras:

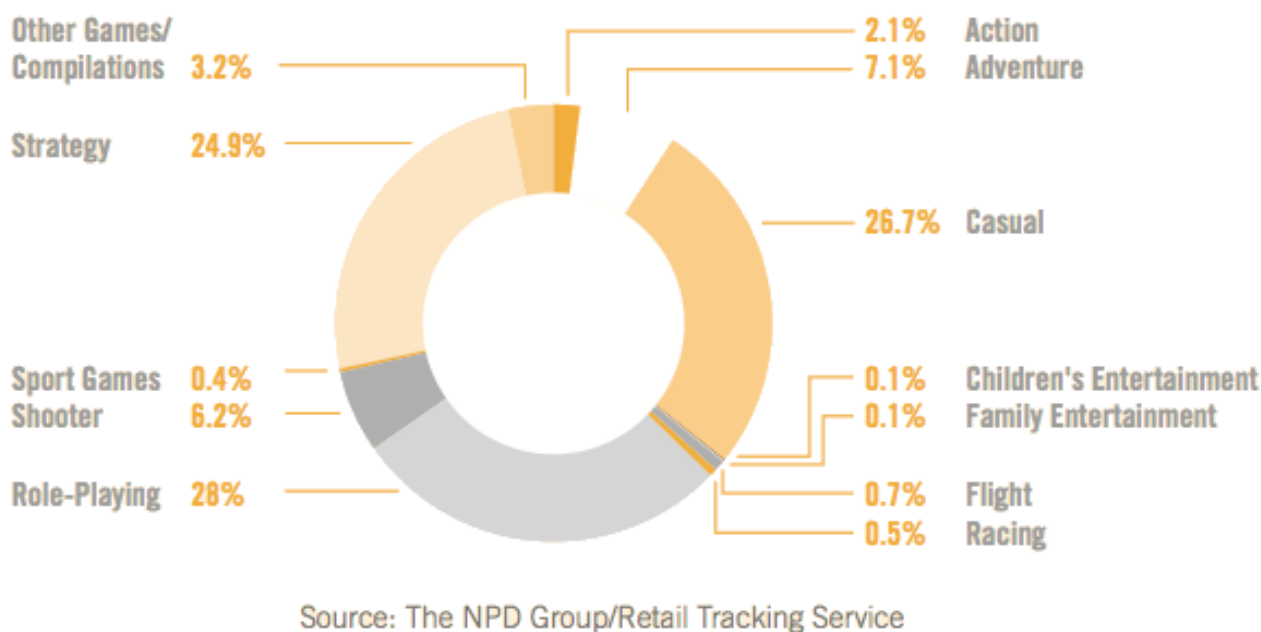


Figura A.8. Videojuegos más vendidos por género en computadoras en unidades (2012)

En los videojuegos para computadoras cambian los gustos de los jugadores, el tipo de juegos más vendido es el de rol, seguido por los ocasionales y estrategia.

Con un valor medio estarían los de aventura y de disparos.

Y con menor interés el resto de tipo de juegos.

A.2. Estudio de mercado de los dispositivos iOS

En esta parte del estudio se trata de evaluar la importancia de los dispositivos iOS en el mercado actual. La comparación se realiza con los dispositivos Android siendo estos los competidores actuales.

En la figura A.9 se observa la tendencia de ser iOS la plataforma que domina en cuanto a nuevos proyectos empezados, pasando de casi un 65% a principios de 2011 a más del 70% pasado mediados de 2011.

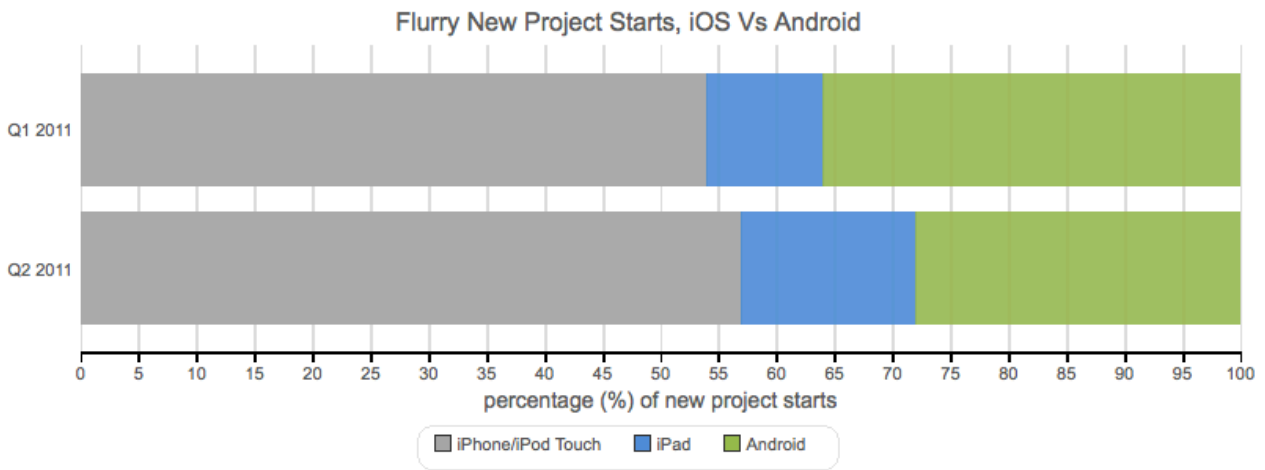


Figura A.9. Distribución de nuevos proyectos en iOS Vs Android

Otra comparación que se va a realizar es la de ver la relación entre el total de dispositivos activos iOS frente a los Android. Desde Apple y Samsung se han hecho oficiales los datos que confirman que Android gana la carrera en cuanto a dispositivos puestos en el mercado como se ve en la figura 10.

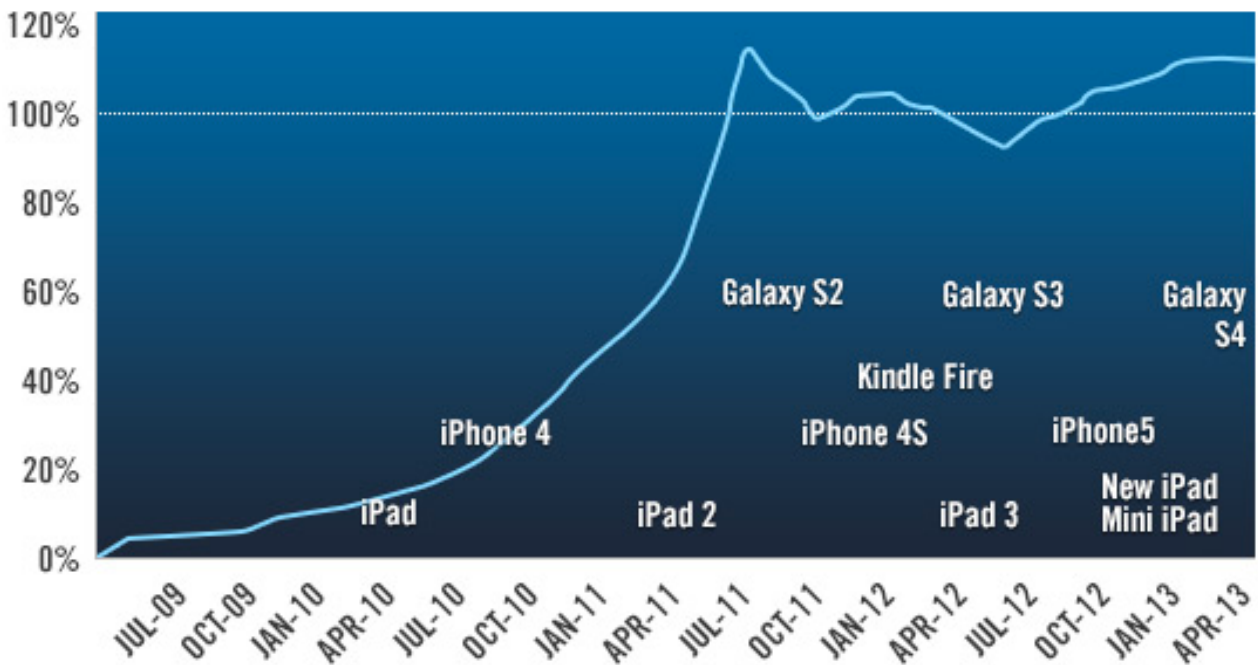


Figura A.10. Dispositivos Android activos como porcentaje de dispositivos iOS

La comparativa se hace a partir de 2009 ya que el primer dispositivo Android que salió al mercado fue en 2008. Desde 2009 hasta mediados de 2011 Apple ha dominado el mercado de dispositivos móviles, pero es a partir de esta fecha cuando pasa de manos el dominio del mercado de ventas y es

Android el que más dispositivos tiene en el mercado. Pese a que Android domina en dispositivos, Apple está cerca, considerándose el número de dispositivos iOS en mercado elevado.

En la figura se muestra también el momento en el que se lanzaron al mercado los dispositivos.

En la figura A.11 se muestra el tiempo pasado por los usuarios en aplicaciones Android respecto al tiempo pasado por los usuarios en las aplicaciones iOS. La comparación está hecha por dispositivo, ya que como se ha comentado en la figura anterior el total de dispositivos Android en el mercado es superior.

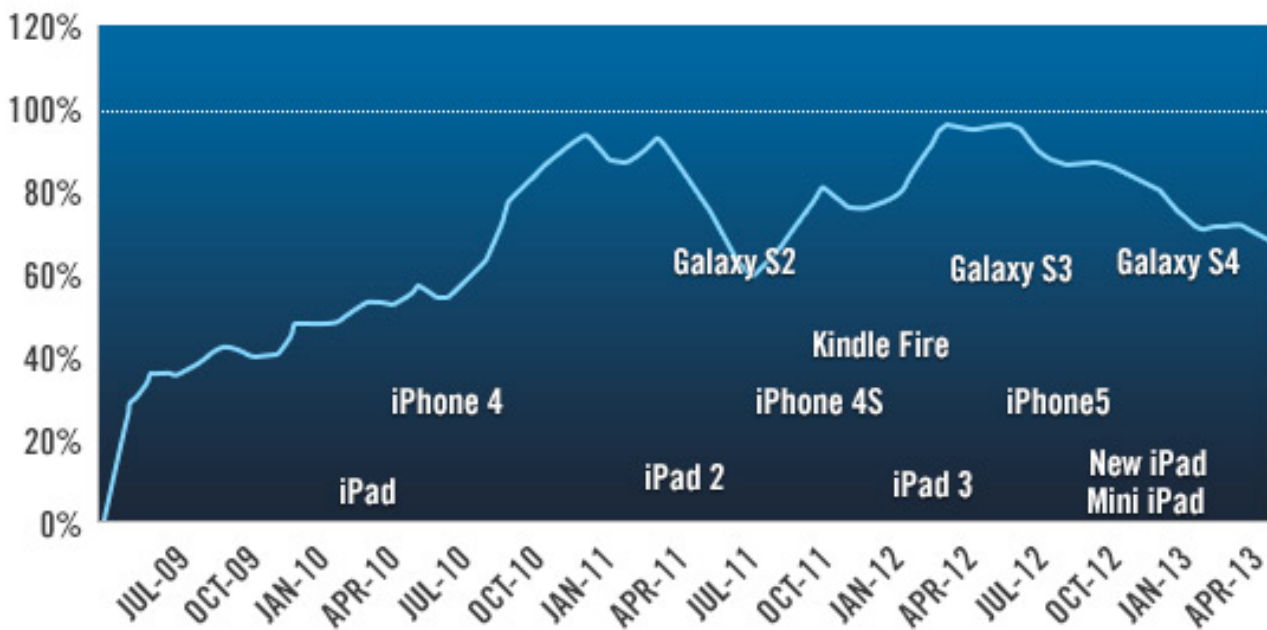


Figura A.11. Total de tiempo pasado en aplicaciones por Android como porcentaje del total de tiempo en aplicaciones por iOS

En la figura A.11 se observa que pese a que Android dispone de más dispositivos en el mercado, es iOS quien domina en tiempo empleado por usuario en un dispositivo. Aunque hay dos momentos en los que prácticamente se igualan, debido a la salida al mercado del iPad 2 y el iPad tercera generación iOS vuelve a distanciarse respecto de Android.

⁴ Fuente de los datos: *Flurry Analytics*.

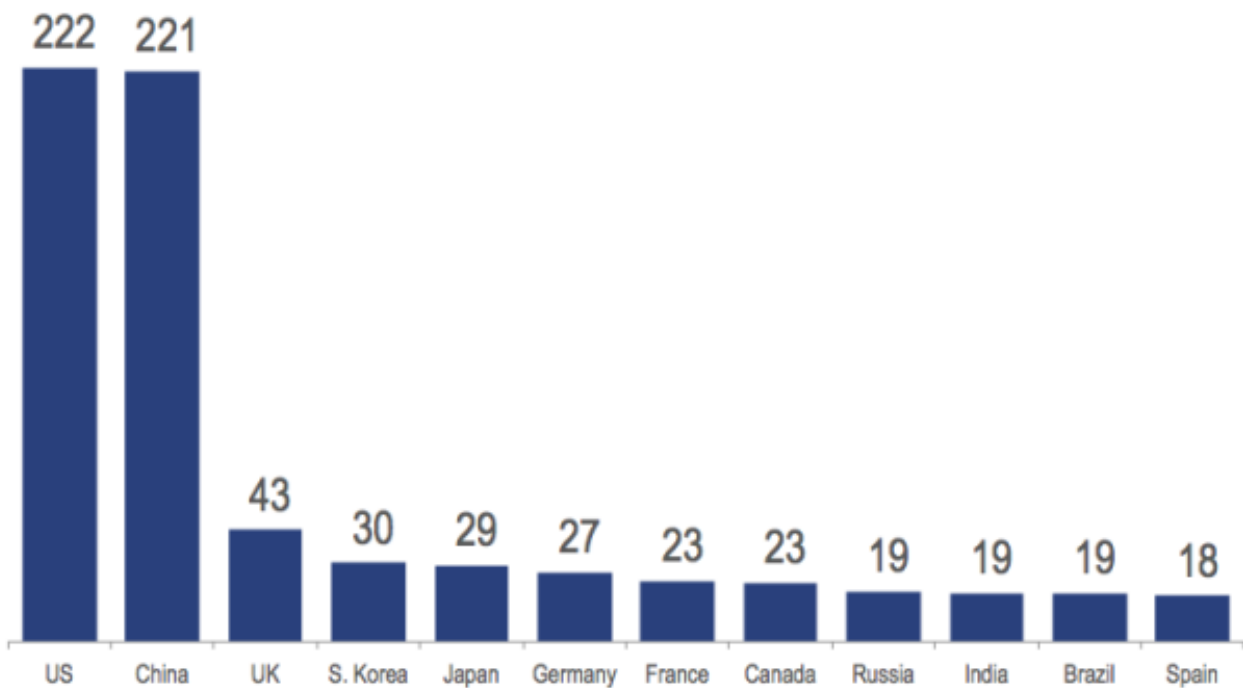


Figura A.12. Países con el mayor número de dispositivos iOS y Android activos (millones) (2013)

Por último en la figura A.12 se observa la distribución de países con más dispositivos iOS y Android activos en la que España está en el puesto doce. Dominando el mercado están EE.UU. y China. En Europa domina el Reino Unido, con un poco menos de la mitad que Reino Unido está Alemania y Francia.

En cuanto a la importancia de los iPad como dispositivos iOS, en la figura A.13 se muestran los millones de iPad vendidos desde el segundo cuarto de 2010 hasta el segundo cuarto de 2013. Notar que en 2013 se han vendido 57 millones de iPads.

Las ventas por cuarto son oscilantes, pero una conclusión que se puede sacar de la tabla es que cada año el número de iPads vendidos se incrementa.

En la tabla A.1, se hace una comparativa entre los iPad de Apple y tabletas de otras compañías. Apple pese a ver reducida las ventas respecto al segundo cuarto del año pasado, como se vio en la tabla anterior, se prevé que el total de ventas aumente.

La tableta de Apple es sin duda la que más ventas registra con 14,6 millones en el tercer cuarto de 2013 con una cuota de mercado del 32% frente al rival más importante, Samsung, con 8,1 millones de ventas y una cuota de

mercado del 18%. Se observa que otras tabletas, como las de Samsung están teniendo un aumento considerable y puede que en un futuro sea una competidora para los iPad de Apple.

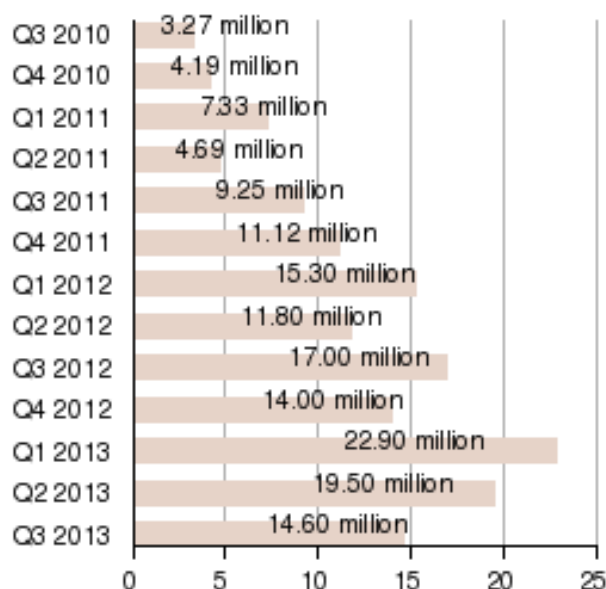


Figura A.13. Millones de ventas de iPad (2010-2013)

Otro dato importante que merece la pena comentar es el crecimiento anual de ventas de este tipo de dispositivos, aumentando casi un 60% en el global de ventas de tabletas.

Vendedor	2°Q2013 ventas	2°Q2013 % Mercado	2°Q2012 ventas	2°Q2012 % Mercado	Crecimiento anual
Apple	14,6	32,4%	17,0	60,3%	-14,1%
Samsung	8,1	18,0%	2,1	7,6%	277,0%
ASUS	2,0	4,5%	0,9	3,3%	120,3%
Lenovo	1,5	3,3%	0,4	1,3%	313,9%
Acer	1,4	3,1%	0,4	1,4%	247,9%
Otros	17,5	38,8%	7,4	26,2%	136,6%
Total	45,1	100,0%	28,3	100,0%	59,6%

Tabla A.1. Millones de ventas de las tabletas de distintos fabricantes (2012-2013)

A.3. Importancia de Unity 3D como motor gráfico

A día de presentación del proyecto, otros datos que muestran la importancia del motor gráfico respecto a su uso son los siguientes:

- ▶ Existen más de 2 millones de desarrolladores registrados para el uso del motor.
- ▶ Al mes hay 400.000 desarrolladores activos que hacen uso de Unity 3D.
- ▶ Se han realizado más de 225 millones de instalaciones del reproductor web de Unity.
- ▶ En julio se registraron 6,6 millones de sesiones del editor.
- ▶ Unity ha anunciado una colaboración estratégica con Microsoft que consiste en ampliar a la versión gratuita el soporte para Windows 8 y Windows Phone 8. Con esto se extiende el uso gratuito que ya existía para Android, iOS y Xbox 360.

A.4. Conclusiones

Las computadoras y los videojuegos están alcanzando niveles de ventas históricos. Hoy en día, cualquier dispositivo con una pantalla permite jugar a videojuegos. Existen gran cantidad de juegos en distintos formatos que llegan a todas las plataformas.

Los consumidores demandan nuevos productos (videojuegos) continuamente consolidando la industria y haciendo que en algunos países como EE.UU. sea un sector realmente importante.

Ningún sector ha sufrido un crecimiento tan fuerte como la industria de los videojuegos y computadoras, esto se hace más significativo en tiempos de crisis como los actuales. Las innovaciones para mejorar la conectividad de los jugadores ha incrementado la demanda de productos relacionados con el sector y ha incrementado la progresión de la expansión y diversificación de los consumidores base.

Por último en la gráfica siguiente se ve la distribución del tiempo empleado por los usuarios según el tipo de aplicaciones en Android e iOS.

En la figura A.14 se observa el tiempo empleado en término medio por los usuarios según la categoría de aplicación para los dispositivos Android e iOS. En ambos destaca sobre el resto de categorías los juegos, independientemente que el usuario tenga un dispositivo iOS y/o Android.

En el caso de Android, el entretenimiento ocupa el segundo lugar, pero con menos de la mitad del tiempo empleado en juegos.

En iOS el segundo tipo de entretenimiento es el de las noticias estando 17 puntos por debajo del uso de juegos.

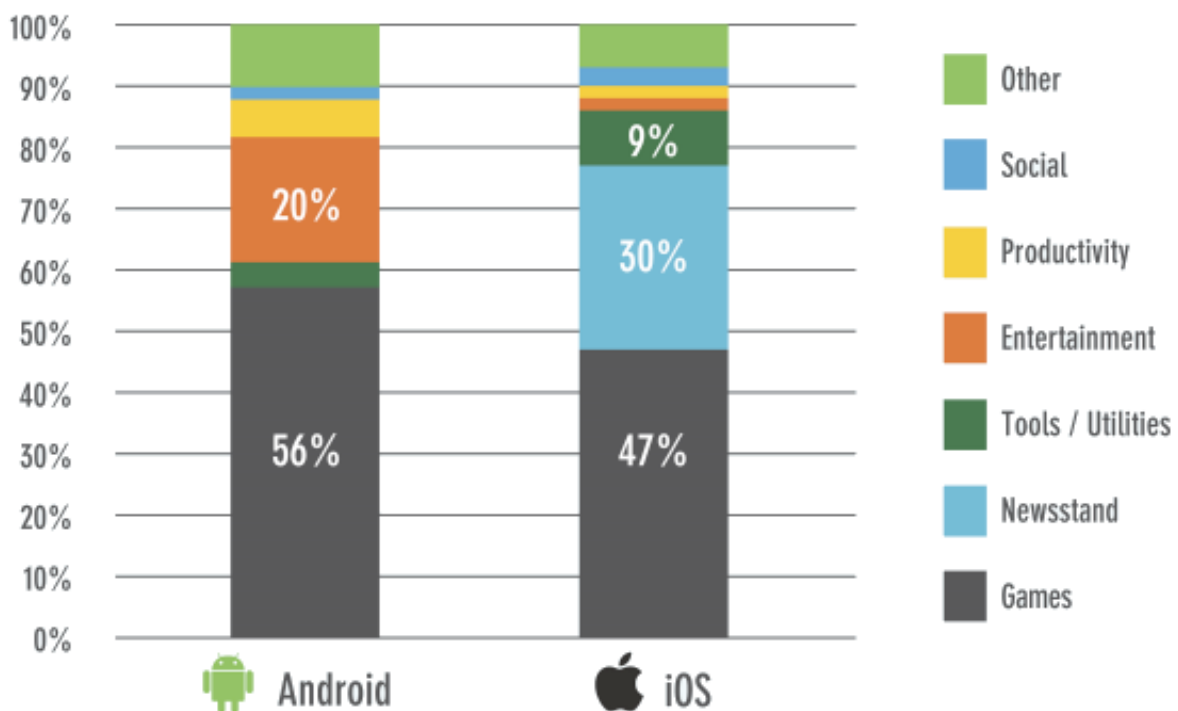


Figura A.14. Tiempo empleado por categoría de aplicación

Por todo lo indicado en el anexo A, la elección de un iPad como dispositivo destino del curso y Unity 3D como motor gráfico a estudiar queda justificada.

Anexo B

Tecnologías empleadas

En este anexo se describen brevemente algunas de las tecnologías empleadas en las cuatro partes del proyecto. El resto por su importancia tienen un anexo propio.

La primera parte como se ha indicado en la memoria es la parte en la que se crean los textos del curso, por lo que su uso fundamental es la redacción de textos, maquetación, creación de imágenes, retoque de imágenes, creación de tablas y creación de figuras.

B.1. Pages

Pages es una herramienta avanzada de escritura y maquetación de Apple que permite crear documentos, boletines, informes y mucho más (<http://www.apple.com/es/iwork/pages/>).



Figura B.1. Icono de la aplicación Pages

Pages forma parte de la suite de productividad de Apple iWork para los sistemas operativos OS X e iOS. La primera versión de Pages es de 2005 y la última a día de creación del presente documento la cuarta, que es la que ha sido empleada en el trabajo.

En 2010 Apple anunció su versión para iPad con una interfaz táctil. En 2011 apareció la versión 1.4 con la posibilidad de uso en iPad, iPhone y en iPod Touch. La versión 4.3 ha sido la empleada en el proyecto

B.2. Keynote

Keynote es un software de presentación de aplicaciones desarrollado por Apple como parte de la suite de productividad iWork.



Figura B.2. Icono de la aplicación Keynote

La versión más reciente es la 5.1.1 que fue anunciada en diciembre de 2011 y que es la que se ha empleado para elaborar las presentaciones del curso. En 2010 Apple desarrolló una versión para iPad con una nueva interfaz táctil diseñada para el dispositivo.

B.3. Numbers

Numbers es una hoja de cálculo desarrollada por Apple como parte de la suite de productividad iWork, junto con Keynote y Pages.



Figura B.3. Icono de la aplicación Numbers

La primera versión de Numbers, la 1.0 se presentó en 2007 siendo la aplicación más reciente de la suite. En enero de 2010 se anunció su versión para iPad con su nueva interfaz táctil. La versión actual es la 2.3 y es la que se ha empleado en el proyecto.

B.4. GIMP

GIMP (GNU Image Manipulation Program) es una herramienta para el retocado y edición fotográfica desarrollado bajo la licencia LGPLv3 como software libre.

GIMP está disponible para la mayor parte de sistemas operativos, Linux, OS X, y Microsoft Windows.



Figura B.4. Icono de la aplicación GIMP

GIMP ha sido una herramienta utilizada en bastantes ocasiones para la realización del proyecto, se ha empleado para crear imágenes para ser usadas en los documentos de texto, también se ha usado para crear o modificarlas texturas de objetos en 3D, para el retoque de capturas de pantalla y para el proceso de mapeado UV que ha permitido crear texturas para objetos del juego más complejos.

Los mapeados UV se han realizado a partir de imágenes de los modelos en 3D creadas con Blender e importadas con GIMP. Desde GIMP se han añadido texturas a los mapeados UV. Una vez terminados los mapeados, se han vuelto a exportar a Blender para ser incorporados al modelo que después sería empleado en Unity 3D.

Un mapeado UV consiste en recortar a partir de sus aristas un objeto en 3D obteniendo una figura plana del modelo que podrá ser manipulada, en este caso con GIMP para crear el aspecto externo visible del objeto 3D.

Por ejemplo, para crear la textura del modelo del enemigo 'Cosa', que se muestra en la figura B.5, se parte del modelo en 3D creado en Blender que se puede ver en la figura B.6.

Una vez creado el modelo en 3D en Blender, es necesario marcar las aristas que harán de unión cuando se cree el mapa UV. Estas aristas marcadas se muestran en la imagen B.7.

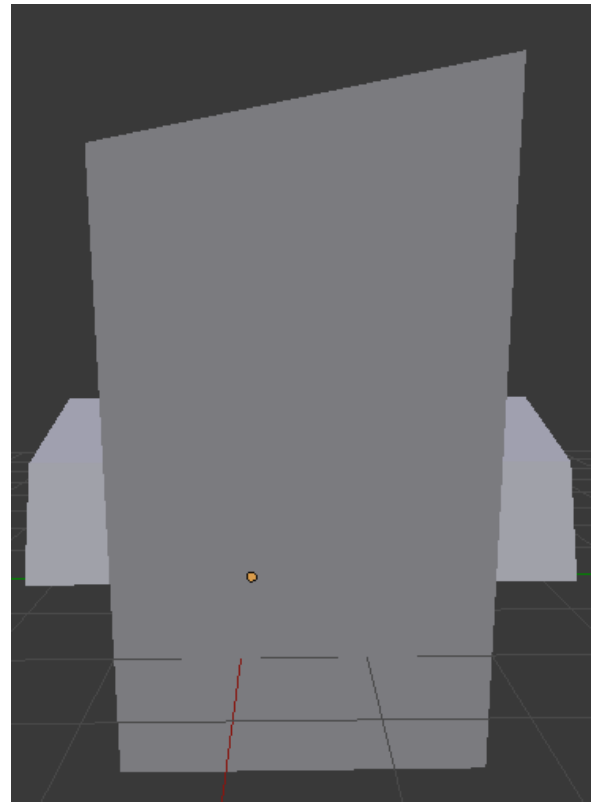
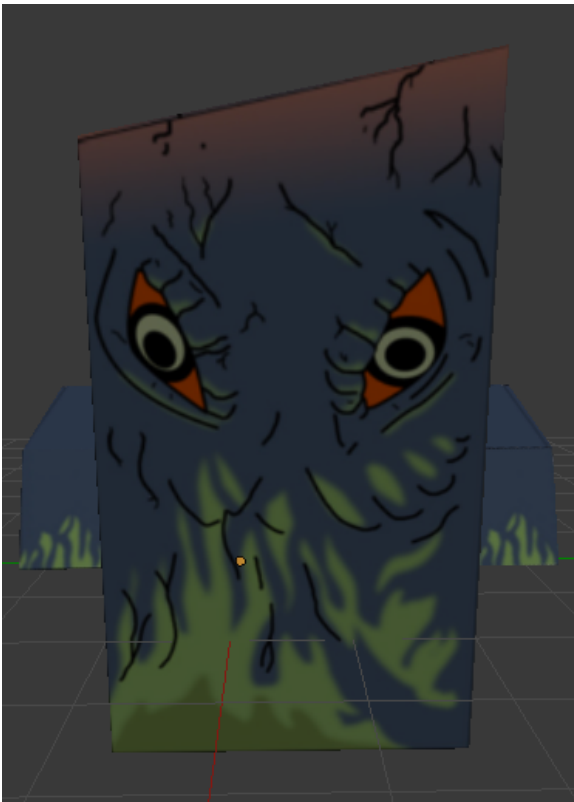


Figura B.5. Modelo 3D del enemigo Cosa con textura aplicada
Figura B.6. Modelo 3D del enemigo Cosa sin textura

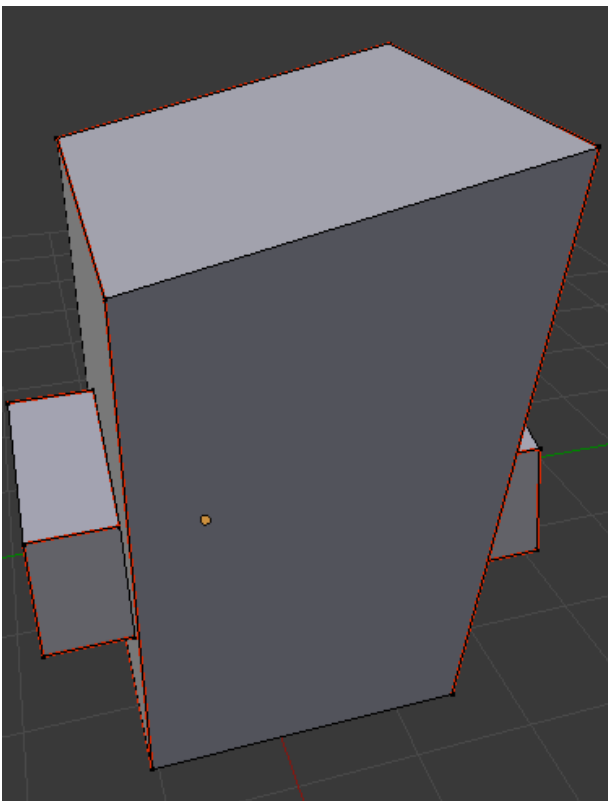


Figura B.7. Modelo 3D del objeto con las aristas marcadas en rojo

Con las aristas marcadas se crea el mapa UV en Blender, que se guardará en un formato adecuado, en el ejemplo en png. Una vez guardado se abrirá con GIMP, figura B.8.

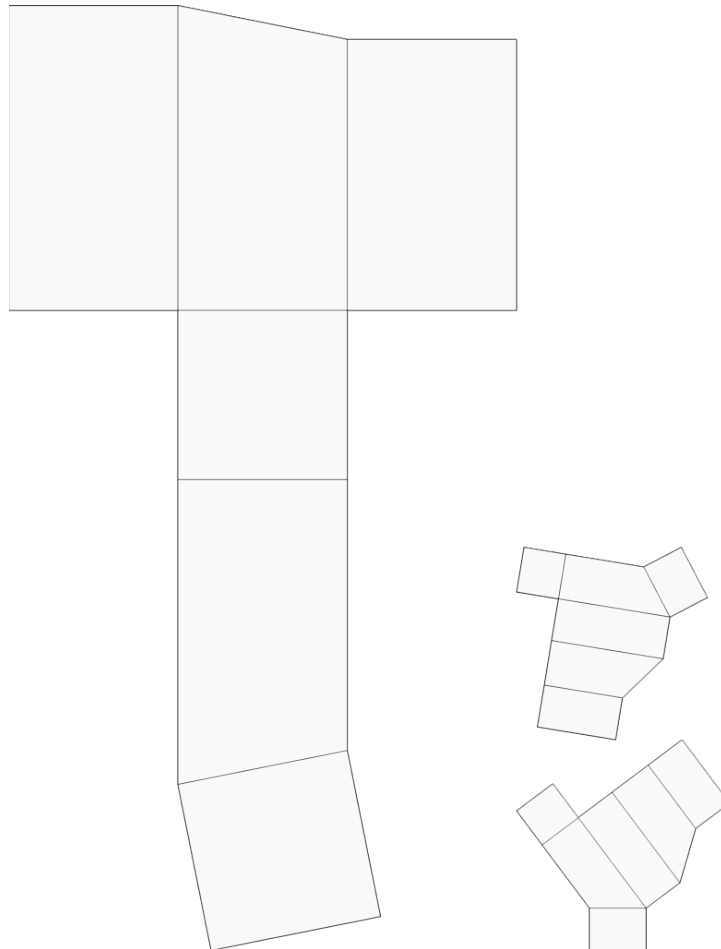


Figura B.8. Mapa UV en formato png importado con GIMP

Tras importar el mapa UV con GIMP, se creará la textura deseada, para el ejemplo, se ha creado una textura de un monstruo con aspecto enfadado. Para ello se han utilizado distintas herramientas de GIMP así como texturas y colores ya creados.

El aspecto que tiene la textura creada para el monstruo 'Cosa' se muestra en la figura B.9.



Figura B.9. Textura creada a partir del mapa UV

Tras crearse, se importará a Blender y se aplicará la textura, mostrando el objeto 3D el aspecto final de la figura B.5.

B.5. Inkscape

Inkscape es un software libre para la edición de gráficos vectoriales. Su principal objetivo es el de dar total soporte a la implementación del formato estándar SVG (Scalable Vector Graphics).

Las características principales son:

- ▶ Creación de objetos.
- ▶ Manipulación de objetos.

- ▶ Decorar objetos.
- ▶ Operaciones sobre rutas.
- ▶ Soporte a la creación de texto.
- ▶ Renderizado.
- ▶ Importación / Exportación de imágenes.



Figura B.10. Logotipo de la aplicación InkScape

InkScape se ha utilizado principalmente para la creación de los fondos de los niveles del juego y de otros elementos como las nubes que decoran la escena. En la figura B.11 se observa el fondo del nivel uno con las nubes, todo creado con InkScape.



Figura B.11. Imagen del primer nivel del videojuego. Fondo y nubes creados con InkScape

Anexo C

Modelado de objetos con Blender

Blender es un software libre para la creación de gráficos 3D para computadoras, usado para la creación de películas animadas, arte gráfico, modelos en 3D, aplicaciones interactivas en 3D y videojuegos.

Entre las características de Blender se incluye el modelado en 3D, creación de un mapa UV a partir de un modelo en 3D, asignación de texturas a objetos, rigging y skinning¹, simulación de humo y fluidos, simulación de partículas, animación, seguimiento de cámara, renderizado y edición de video entre otras.



Figura C.1. Logotipo de la aplicación Blender

La mayor parte de objetos en 3D del videojuego completo se han creado utilizando Blender como herramienta.

Para la creación de los modelos del personaje principal y algunos enemigos se ha seguido el proceso mostrado en la figura C.2:

¹ Rigging es el proceso por el cual cuando al terminar el modelado de un objeto 3D se le crea un sistema de huesos para que pueda ser animado.

¹ Skinning un modelo 3D es el proceso por el que la superficie visible del modelo es pintado y colocado sobre la malla del modelo. Es una parte del proceso de creación de un objeto que puede requerir gran cantidad de trabajo.

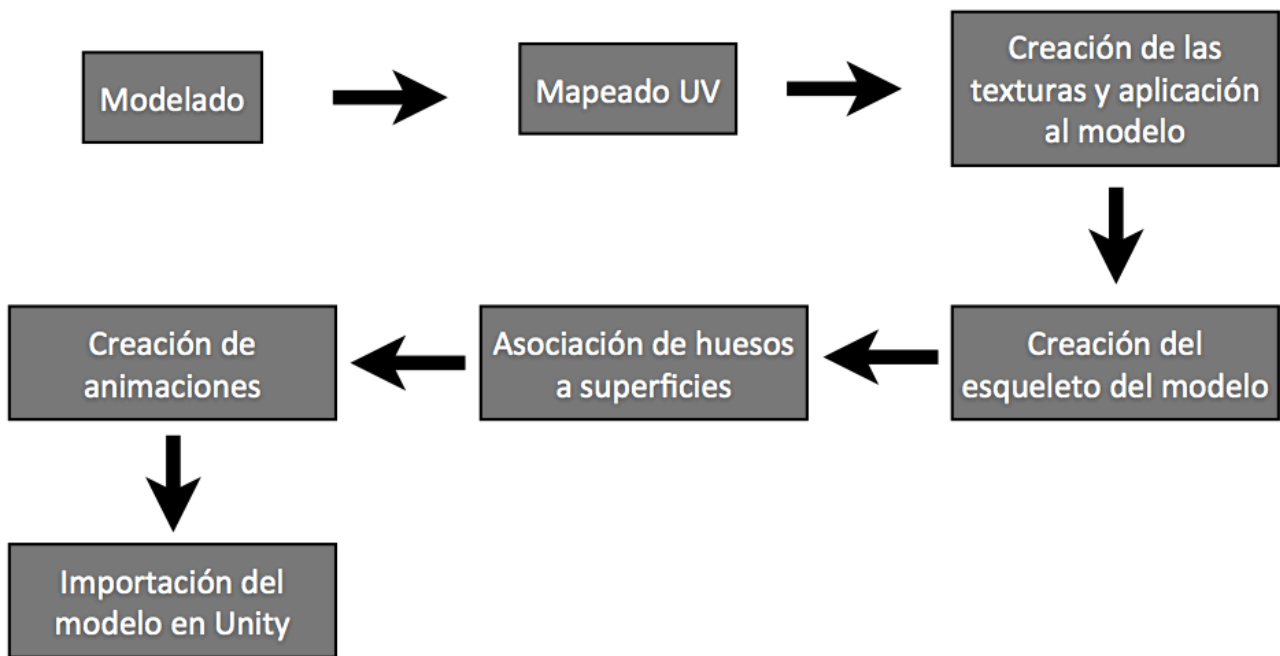


Figura C.2. Proceso de creación del personaje principal y de algunos enemigos

En el anexo I (páginas 678 - 693) se muestra en detalle el ejemplo de la creación del modelo del personaje principal que sirve como ejemplo para cualquiera de los elementos del juego.

Para cada modelo se recomienda hacer una plantilla como la que se muestra en la figura C.4.

El modelo de plantilla se muestra en la figura C.3. El modelo es una recomendación de tipo de plantilla que muestra un boceto para cada elemento del juego.

Logotipo

Nombre del videojuego _____

Para todos los personajes, monstruos, objetos y objetos únicos

Nombre: _____

Tipo: _____

Entorno: _____

Alzado	Alzado posterior	Perfil izquierdo	Perfil derecho
Notas:			Colores clave: <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

Figura C.3. Modelo de plantilla para la creación de los modelos de un videojuego



La venganza de Timan

Para todos los personajes, monstruos, objetos y objetos únicos

Nombre: Timan Tipo: Personaje principal Entorno: Nivel 1, 2, 3

Alzado	Alzado posterior	Perfil izquierdo	Perfil derecho
			
Alzado	Alzado posterior	Perfil izquierdo	Perfil derecho
Notas: aspecto del personaje principal para los niveles 1 y 2. En el nivel 3 Timan tiene alas negras. El resto de elementos y colores se mantienen.			Colores clave: <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

Figura C.4. Ejemplo de boceto de Timan

Anexo D

Unity 3D

El avance tecnológico tanto en computación como en la mejora de las comunicaciones junto al desarrollo de contenido en 3D multiplataforma ha abierto el camino al desarrollo de aplicaciones interactivas en 3D en una variedad de formas muy amplia, como educación, cultura, entrenamientos virtuales, turismo, comercio electrónico y videojuegos entre otras. Gracias a estos progresos, los videojuegos se están convirtiendo en algo común en la mayoría de entornos en los que nos desenvolvemos, por ello, el desarrollo de los mismos supone una actividad de gran importancia.

Unity 3D es un motor gráfico multiplataforma (Mac OS X, Windows, Linux). Unity 3D se distribuye junto a un IDE (Entorno de Desarrollo Integrado) que permite desarrollar aplicaciones interactivas, animaciones en tiempo real y en 3D, visualizaciones y juegos.

El destino de una aplicación desarrollada en Unity 3D puede ser múltiple, desde un navegador web, aplicaciones de escritorio (Mac, Windows, Linux), Flash, dispositivos móviles (iOS, Android) incluso consolas como PS3, Xbox 360 y Wii.

Unity permite la creación de contenido 3D (especialmente videojuegos) de forma más rápida y sencilla que la mayoría de entornos existentes.

Las aplicaciones desarrolladas con Unity se estructuran en escenas, cada escena puede ser cualquier parte de la aplicación. Los menús creados dentro de la aplicación, el área principal de la misma, etc ... son escenas dentro del entorno.

El motor gráfico incluye también un editor de terrenos, en los que se pueden crear mapas con distintas alturas, utilizar distintas geometrías usando herramientas visuales, dotar a las escenas de distintas texturas y objetos (como arbustos, rocas, ...). También dispone de librerías que facilitan el uso de física en los objetos creados, uso de audio, control de colisiones, mensajes y eventos, incorporación de luces y mucho más.

El motor gráfico Unity 3D ha provocado una revolución en el mundo del desarrollo de aplicaciones multimedia. En poco mas de cinco años de existencia ha ganado premios de revistas especializadas (Grand Prix Award). En la actualidad (principios de 2013) existen mas de 2 millones de desarrolladores que usan Unity, algunos tan importantes como Cartoon Network, Coca-Cola, Disney, Electronic Arts, LEGO, Microsoft, Nasa, U.S. Army, Warner Bros. También se ha hecho popular dentro de pequeños estudios, profesionales independientes, estudiantes y aficionados.

D.1. UnityScript

UnityScript no es JavaScript, es un lenguaje que presenta una similitud pero también tiene diferencias respecto a JavaScript.

UnityScript es más rápido, eso es debido a que se compila, no como JavaScript que es interpretado.

No hay diferencias de velocidad entre el uso de UnityScript, C# y Boo, todos lenguajes que se pueden usar para la programación en Unity. Hay otros pros y contras pero la velocidad no es uno de ellos.

A continuación se indican las características más relevantes de UnityScript y C#.

D.1.1. Generalidades

D.1.1.1. Nombre

El nombre JavaScript es un nombre genérico que se puede referir a cualquier implementación de la especificación ECMAScript. UnityScript es un lenguaje creado que no se ajusta a esa especificación y tampoco lo intenta. UnityScript es un lenguaje propietario y no sigue ninguna especificación concreta, es un lenguaje creado por los desarrolladores de Unity.

UnityScript es más parecido a Microsoft JScript.NET que a JavaScript, aunque tampoco es idéntico.

D.1.1.2. JavaScript se encuentra liberado del uso de clases

JavaScript no tiene clases, esto es debido a que es un lenguaje basado en prototipos, la herencia ocurre entre objetos en vez de con clases.

UnityScript posee clases, una vez que se define una clase, esa clase existe durante la ejecución del programa. Un ejemplo de clase en UnityScript podría ser:

```
class Movimiento {
    private var velocidad : int; // Variable privada

    function Movimiento (x : int) {
        this.velocidad = x;
    }

    function MostrarVelocidad () {
        print(this.velocidad);
    }
}

var c = new Movimiento (10);

c.MostrarVelocidad();
```

D.1.1.3. Nombre del fichero como clase

UnityScript intenta ahorrar tiempo en la escritura de código, para ello, la mayoría de ficheros representan simples clases, así automáticamente el nombre de un fichero es usado para definir una clase que el propio fichero se asume implementará.

Así por ejemplo el siguiente fichero Enemigo.js:

```
function MostrarNombre () {
    Debug.Log("Vampiro");
}

function void Wait () {
    while () {
    }
}

function Muerte () {
    Application.Quit();
}
```

Se interpreta como el siguiente código C#:

```
using UnityEngine;

class Enemigo : MonoBehaviour {
    public void MostrarNombre() {
        Debug.Log("Vampiro");
    }

    public void Wait () {
        while () {
        }
    }

    public void Muerte () {
        Application.Quit();
    }
}
```

D.1.2. Sintaxis

D.1.2.1. Punto y coma obligatorio

Mientras que en JavaScript los punto y coma son opcionales, en UnityScript son obligatorios. Se requiere un punto y coma en las siguientes situaciones:

- ▶ Después de las instrucciones *return*, *continue* o *break*.
- ▶ Después de expresiones del tipo:
`transform.Translate(0, 0, 5);`
- ▶ Después de declaración de variables.
- ▶ Después de asignación de variables.
- ▶ Después de declaraciones de métodos sin cuerpo como por ejemplo en interfaces.
- ▶ Entre los parámetros de una instrucción *for*.

D.1.2.2. Una declaración de variable por línea

JavaScript soporta múltiples declaraciones de variables en una sola sentencia:

```
var x = 3, y = 4;
```

UnityScript no lo permite.

D.1.2.3. Las asignaciones no pueden ser una expresión

En JavaScript las asignaciones se tratan como expresiones:

```
var x = 3;           // x es 3  
var y = (x = x + 2); // x es 5, y es 5
```

En UnityScript la asignación no puede ser una expresión:

```
var x = 3;    // x es 3  
x = x + 2;    // x es 5  
var y = x;    // y es 5
```

Hay excepción con pre/post incrementos y decrementos:

```
var x = 3;  
var y = x++; // x es 4, y es 3  
  
x = 3;  
var z = ++x; // x es 4, z es 4
```

D.1.2.4. Tipo privado

En JavaScript el tipo privado se debe encapsular dentro de una función:

```
function Enemigo () {
    var vidas = 3;      // privada

    this.MostrarVidas = function() { print(vidas); };
}

var enemigo1 = new Enemigo();
print(enemigo1.vidas);           // Indefinido
enemigo1.MostrarVidas();        // 3
```

En UnityScript es más intuitivo:

```
class Enemigo () {
    private var vidas : int;

    funtion Enemigo () {
        vidas = 3;
    }

    function MostrarVidas () {
        print(vidas);
    }
}

var enemigo1 = new Enemigo();
print(enemigo1.MostrarVidas()); // Indefinido
enemigo1.MostrarVIDas();       // 3
```

D.1.2.5. #pragma strict

Para el desarrollo de aplicaciones para iOS es un buen hábito el uso de la directiva *#pragma strict*. Esta directiva fuerza a la realización de una comprobación de tipos, generando mensajes de error más pronto e introduciendo buenos hábitos de programación.

D.1.2.6. Otras características

- ▶ Se deben declarar las variables antes de usarlas.
- ▶ Ambos soportan el tipado dinámico.
- ▶ No se permite el uso en UnityScript del símbolo \$ como identificador.
- ▶ No existe la instrucción *with* en UnityScript.
- ▶ UnityScript tiene características de .NET. UnityScript soporta clases así como niveles de protección (*public*, *private*, *protected*) y palabras reservadas como *static*.
- ▶ UnityScript tiene soporte para tipos genéricos mientras que JavaScript no tiene notación para ello.
- ▶ UnityScript no permite el uso de *delete*, por lo que no se pueden eliminar variables declaradas del espacio de nombres.
- ▶ No existen literales para expresiones regulares en UnityScript. En programación de videojuegos no suele ser común utilizar este tipo de expresiones.
- ▶ Se pueden usar funciones virtuales para sobrescribir funciones.
- ▶ El manejo de cadenas se hace con la clase String de mono y no string.

D.2. C#

C# (leído en inglés “C Sharp” y en castellano “C Almohadilla”) es un lenguaje de propósito general diseñado por Microsoft para su plataforma .NET.

Sus principales creadores son Scott Wiltamuth y Anders Hejlsberg, este último también conocido por haber sido el diseñador del lenguaje Turbo Pascal y la herramienta RAD Delphi.

Aunque es posible escribir código para la plataforma .NET en muchos otros lenguajes, C# es el único que ha sido diseñado específicamente para ser utilizado en ella, por lo que programarla usando C# es mucho más sencillo e intuitivo que hacerlo con cualquiera de los otros lenguajes ya que C# carece

de elementos heredados innecesarios en .NET. Por esta razón, se suele decir que C# es el lenguaje nativo de .NET.

La sintaxis y estructuración de C# es muy similar a la de C++, ya que la intención de Microsoft con C# es facilitar la migración de códigos escritos en estos lenguajes a C# y facilitar su aprendizaje a los desarrolladores habituados a ellos. Sin embargo, su sencillez y el alto nivel de productividad son equiparables a los de Visual Basic.

En resumen, C# es un lenguaje de programación que toma las mejores características de lenguajes preexistentes como Visual Basic, Java o C++ y las combina en uno solo.

D.2.1. Características de C#¹

► **Sencillez:** C# elimina muchos elementos que otros lenguajes incluyen y que son innecesarios en .NET. Por ejemplo:

- El código escrito en C# es autocontenido, lo que significa que no necesita de ficheros adicionales al propio fuente tales como ficheros de cabecera.
- El tamaño de los tipos de datos básicos es fijo e independiente del compilador, sistema operativo o máquina para la que se compile (no como en C++), lo que facilita la portabilidad del código.
- No se incluyen elementos de lenguajes como C++ tales como macros, herencia múltiple o la necesidad de un operador diferente del punto (.) para acceder a miembros de espacios de nombres (::).

► **Modernidad:** C# incorpora en el propio lenguaje elementos que a lo largo de los años han ido demostrándose son muy útiles para el desarrollo de aplicaciones y que en otros lenguajes como Java o C++ hay que simular, como un tipo básico decimal que permita realizar operaciones de alta precisión con reales de 128 bits (muy útil en el mundo financiero), la inclusión de una instrucción *foreach* que permita recorrer colecciones con facilidad y es ampliable a tipos definidos por el usuario, la inclusión de un tipo básico *string* para representar cadenas o la distinción de un tipo *bool* específico para representar valores lógicos.

¹ Las características de C# han sido obtenidas del libro “El lenguaje de programación C#” de José Antonio González Seco.

► **Orientación a objetos:** como todo lenguaje de programación de propósito general actual, C# es un lenguaje orientado a objetos. Una diferencia de este enfoque orientado a objetos respecto al de otros lenguajes como C++ es que el de C# es más puro en tanto que no admiten ni funciones ni variables globales sino que todo el código y datos han de definirse dentro de definiciones de tipos de datos, lo que reduce problemas por conflictos de nombres y facilita la legibilidad del código.

C# soporta todas las características propias del paradigma de programación orientada a objetos: encapsulación, herencia y polimorfismo.

En lo referente a la encapsulación es importante señalar que aparte de los típicos modificadores *public*, *private* y *protected*, C# añade un cuarto modificador llamado *internal*, que puede combinarse con *protected* e indica que al elemento a cuya definición precede solo puede accederse desde su mismo ensamblado.

Respecto a la herencia, a diferencia de C++ y al igual que Java, C# solo admite herencia simple de clases, esta puede ser simulada con facilidad mediante herencia múltiple de interfaces.

► **Orientación a componentes:** la propia sintaxis de C# incluye elementos propios del diseño de componentes que otros lenguajes tienen que simular mediante construcciones más o menos complejas. Es decir, la sintaxis de C# permite definir cómodamente propiedades (similares a campos de acceso controlado), eventos (asociación controlada de funciones de respuesta a notificaciones) o atributos (información sobre un tipo o sus miembros).

► **Gestión automática de memoria:** todo lenguaje de .NET tiene a su disposición el recolector de basura. Esto tiene el efecto en el lenguaje de que no es necesario incluir instrucciones de destrucción de objetos. Sin embargo, dado que la destrucción de los objetos a través del recolector de basura es indeterminista y solo se realiza cuando este se active (ya sea por falta de memoria, finalización de la aplicación o solicitud explícita en el fuente), C# también proporciona un mecanismo de liberación de recursos determinista a través de la instrucción *using*.

► **Seguridad de tipos:** C# incluye mecanismos que permiten asegurar que los accesos a tipos de datos siempre se realicen correctamente, lo que permite evitar que se produzcan errores difíciles de detectar por acceso a memoria no perteneciente a ningún objeto y es especialmente necesario en

un entorno gestionado por un recolector de basura. Para ello se toman medidas del tipo:

- Solo se admiten conversiones entre tipos compatibles. Esto es, entre un tipo y antecesores suyos, entre tipos para los que explícitamente se haya definido un operador de conversión, y entre un tipo y un tipo hijo suyo del que un objeto del primero almacenase una referencia del segundo (downcasting).
- No se pueden usar variables no inicializadas. El compilador da a los campos un valor por defecto consistente en ponerlos a cero y controla mediante análisis del flujo de control del fuente que no se lea ninguna variable local sin que se le haya asignado previamente algún valor.
- Se comprueba que todo acceso a los elementos de una tabla se realice con índices que se encuentren dentro del rango de la misma.
- Se puede controlar la producción de desbordamientos en operaciones aritméticas, informándose de ello con una excepción cuando ocurra. Sin embargo, para conseguirse un mayor rendimiento en la aritmética estas comprobaciones no se hacen por defecto al operar con variables sino solo con constantes (se pueden detectar en tiempo de compilación)
- A diferencia de Java, C# incluye delegados, que son similares a los punteros a funciones de C++ pero siguen un enfoque orientado a objetos, pueden almacenar referencias a varios métodos simultáneamente, y se comprueba que los métodos a los que apunten tengan parámetros y valor de retorno del tipo indicado al definirlos.
- Pueden definirse métodos que admitan un número indefinido de parámetros de un cierto tipo, y a diferencia lenguajes como C/C++, en C# siempre se comprueba que los valores que se les pasen en cada llamada sean de los tipos apropiados.

► **Instrucciones seguras:** para evitar errores muy comunes, en C# se han impuesto una serie de restricciones en el uso de las instrucciones de control más comunes. Por ejemplo, la guarda de toda condición ha de ser una expresión condicional y no aritmética, con lo que se evitan errores por confusión del operador de igualdad (==) con el de asignación (=); y todo caso de un switch ha de terminar en un break o goto que indique cuál es la

siguiente acción a realizar, lo que evita la ejecución accidental de casos y facilita su reordenación.

► **Sistema de tipos unificado:** a diferencia de C++, en C# todos los tipos de datos que se definan siempre derivarán, aunque sea de manera implícita, de una clase base común llamada `System.Object`, por lo que dispondrán de todos los miembros definidos en esta clase (es decir, serán “objetos”).

A diferencia de Java, en C# esto también es aplicable a los tipos de datos básicos. Además, para conseguir que ello no tenga una repercusión negativa en su nivel de rendimiento, se ha incluido un mecanismo transparente de boxing y unboxing con el que se consigue que solo sean tratados como objetos cuando la situación lo requiera, y mientras tanto puede aplicárseles optimizaciones específicas.

El hecho de que todos los tipos del lenguaje deriven de una clase común facilita enormemente el diseño de colecciones genéricas que puedan almacenar objetos de cualquier tipo.

► **Extensibilidad de tipos básicos:** C# permite definir, a través de estructuras, tipos de datos para los que se apliquen las mismas optimizaciones que para los tipos de datos básicos. Es decir, que se puedan almacenar directamente en pila (luego su creación, destrucción y acceso serán más rápidos) y se asignen por valor y no por referencia. Para conseguir que lo último no tenga efectos negativos al pasar estructuras como parámetros de métodos, se da la posibilidad de pasar referencias a pila a través del modificador de parámetro *ref*.

► **Extensibilidad de operadores:** para facilitar la legibilidad del código y conseguir que los nuevos tipos de datos básicos que se definan a través de las estructuras estén al mismo nivel que los básicos predefinidos en el lenguaje, al igual que C++ y a diferencia de Java, C# permite redefinir el significado de la mayoría de los operadores (incluidos los de conversión, tanto para conversiones implícitas como explícitas) cuando se apliquen a diferentes tipos de objetos.

► **Versionable:** C# incluye una política de versionado que permite crear nuevas versiones de tipos sin temor a que la introducción de nuevos miembros provoquen errores difíciles de detectar en tipos hijos previamente desarrollados y ya extendidos con miembros de igual nombre a los recién introducidos.

- ▶ **Eficiente:** en principio, en C# todo el código incluye numerosas restricciones para asegurar su seguridad y no permite el uso de punteros. Sin embargo, y a diferencia de Java, en C# es posible saltarse dichas restricciones manipulando objetos a través de punteros. Para ello basta marcar regiones de código como inseguras (modificador *unsafe*) y podrán usarse en ellas punteros de forma similar a cómo se hace en C++, lo que puede resultar vital para situaciones donde se necesite una eficiencia y velocidad de procesamiento muy grandes.
- ▶ **Compatible:** para facilitar la migración de programadores, C# no solo mantiene una sintaxis muy similar a C, C++ o Java que permite incluir directamente en código escrito en C# fragmentos de código escrito en estos lenguajes, sino también ofrece la posibilidad de acceder a código nativo escrito como funciones sueltas.

Anexo E

Tecnologías web empleadas

E.1. AJAX

El término AJAX se presentó por primera vez en el artículo "Ajax: A New Approach to Web Applications" publicado por Jesse James Garrett el 18 de Febrero de 2005. Hasta ese momento, no existía un término normalizado que hiciera referencia a un nuevo tipo de aplicación web que estaba apareciendo.

En realidad, el término AJAX es un acrónimo de Asynchronous JavaScript + XML, que se puede traducir como "JavaScript asíncrono + XML".

Ajax no es una tecnología en sí mismo. En realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes.



Figura E.1. Logotipo de AJAX

Las tecnologías que forman AJAX son:

- ▶ XHTML y CSS, para crear una presentación basada en estándares.
- ▶ DOM, para la interacción y manipulación dinámica de la presentación.
- ▶ XML, XSLT y JSON, para el intercambio y la manipulación de información.

- ▶ XMLHttpRequest, para el intercambio asíncrono de información.
- ▶ JavaScript, para unir todas las demás tecnologías.

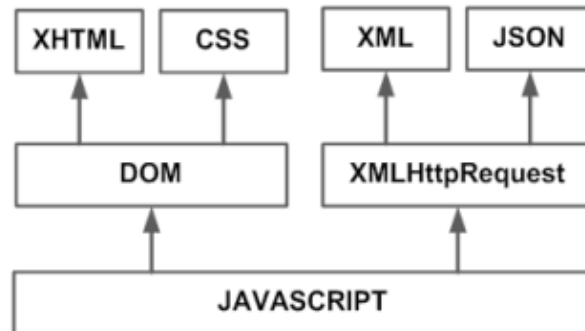


Figura E.2. Tecnologías que forman parte de AJAX

En las aplicaciones web tradicionales, las acciones del usuario en la página (pinchar en un botón, seleccionar un valor de una lista, etc.) desencadenan llamadas al servidor. Una vez procesada la petición del usuario, el servidor devuelve una nueva página HTML al navegador del usuario.

Esta técnica tradicional para crear aplicaciones web funciona correctamente, pero no crea una buena sensación al usuario. Al realizar peticiones continuas al servidor, el usuario debe esperar a que se recargue la página con los cambios solicitados. Si la aplicación debe realizar peticiones continuas, su uso se convierte en algo molesto.

AJAX permite mejorar completamente la interacción del usuario con la aplicación, evitando las recargas constantes de la página, ya que el intercambio de información con el servidor se produce en un segundo plano.

Las aplicaciones construidas con AJAX eliminan la recarga constante de páginas mediante la creación de un elemento intermedio entre el usuario y el servidor. La nueva capa intermedia de AJAX mejora la respuesta de la aplicación, ya que el usuario nunca se encuentra con una ventana del navegador vacía esperando la respuesta del servidor.

Las peticiones HTTP al servidor se sustituyen por peticiones JavaScript que se realizan al elemento encargado de AJAX. Las peticiones más simples no requieren intervención del servidor, por lo que la respuesta es inmediata. Si la interacción requiere una respuesta del servidor, la petición se realiza de forma asíncrona mediante AJAX. En este caso, la interacción del usuario

tampoco se ve interrumpida por recargas de página o largas esperas por la respuesta del servidor.

E.2. PHP

PHP, acrónimo de "PHP: Hypertext Preprocessor", es un lenguaje de 'scripting' de propósito general y de código abierto que está especialmente pensado para el desarrollo web y que puede ser embebido en páginas HTML. Su sintaxis recurre a C, Java y Perl, y es fácil de aprender. La meta principal de este lenguaje es permitir a los desarrolladores web escribir dinámicamente y rápidamente páginas web generadas; aunque se puede hacer mucho más con PHP.



Figura E.3. Logotipo de MySQL

El código de PHP está encerrado entre las etiquetas especiales de comienzo y final `<?php` y `?>` que permiten entrar y salir del "modo PHP".

Lo que distingue a PHP de otros lenguajes como Javascript es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente. El cliente recibirá el resultado de ejecutar el script, aunque no recibirá el código subyacente. El servidor web puede ser incluso configurado para que procese todos los ficheros HTML con PHP.

Lo mejor de usar PHP es que es extremadamente simple para el principiante, pero a su vez ofrece muchas características avanzadas para los programadores profesionales.

PHP está enfocado principalmente a la programación de scripts del lado del servidor, por lo que se puede hacer cualquier cosa que pueda hacer otro programa CGI, como recopilar datos de formularios, generar páginas con contenidos dinámicos, o enviar y recibir cookies. Aunque PHP puede hacer mucho más.

Existen principalmente tres campos principales donde se usan scripts de PHP.

- ▶ Scripts del lado del servidor. Este es el campo más tradicional y el foco principal. Se necesitan tres cosas para que esto funcione. El analizador de PHP (módulo CGI o servidor), un servidor web y un navegador web.
- ▶ Scripts desde la línea de comandos. Se puede crear un script de PHP y ejecutarlo sin necesidad de un servidor o navegador. Solamente es necesario el analizador de PHP para utilizarlo de esta manera.
- ▶ Escribir aplicaciones de escritorio. Probablemente PHP no sea el lenguaje más apropiado para crear aplicaciones de escritorio con una interfaz gráfica de usuario, pero si se conoce bien PHP, y se quisiera utilizar algunas características avanzadas de PHP en aplicaciones del lado del cliente, se puede utilizar PHP-GTK para escribir dichos programas. También es posible de esta manera escribir aplicaciones independientes de la plataforma. PHP-GTK es una extensión de PHP, no disponible en la distribución principal.

PHP puede usarse en todos los principales sistemas operativos, incluyendo Linux, muchas variantes de Unix (incluyendo HP-UX, Solaris y OpenBSD), Microsoft Windows, Mac OS X, RISC OS y probablemente otros más. PHP admite la mayoría de servidores web de hoy en día, incluyendo Apache, IIS, y muchos otros.

Con PHP, se tiene la posibilidad de utilizar programación por procedimientos o programación orientada a objetos (POO), o una mezcla de ambas.

Con PHP no se está limitado a generar HTML. Entre las capacidades de PHP se incluyen la creación de imágenes, ficheros PDF e incluso películas Flash (usando libswf y Ming) generadas sobre la marcha.

También se puede generar fácilmente cualquier tipo de texto, como XHTML y cualquier otro tipo de fichero XML. PHP puede autogenerar éstos ficheros y guardarlos en el sistema de ficheros en vez de imprimirlos en pantalla, creando una caché en el lado del servidor para contenido dinámico.

Una de las características más potentes y destacables de PHP es su soporte para un amplio abanico de bases de datos. Escribir una página web con acceso a una base de datos es simple utilizando una de las extensiones

específicas de bases de datos o conectarse a cualquier base de datos que admita el estándar de Conexión Abierta a Bases de Datos por medio de la extensión ODBC.

PHP también cuenta con soporte para comunicarse con otros servicios usando protocolos tales como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (en Windows) y muchos otros. También se pueden crear sockets de red puros e interactuar usando cualquier otro protocolo. PHP tiene soporte para el intercambio de datos complejos de WDDX entre virtualmente todos los lenguajes de programación web. Para la interconexión, PHP posee soporte para la instalación de objetos Java y usarlos de forma transparente como objetos de PHP.

PHP tiene útiles características de procesamiento de texto, las cuales incluyen las expresiones regulares compatibles con Perl (PCRE), y muchas extensiones y herramientas para el acceso y análisis de documentos XML. PHP estandariza todas las extensiones XML sobre el fundamento sólido de libxml2, y amplía este conjunto de características añadiendo soporte para SimpleXML, XMLReader y XMLWriter.

E.3. MySQL

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario.



Figura E.4. Logotipo de MySQL

MySQL es el mayor sistema gestor de bases de datos de código abierto SQL, es desarrollado, distribuido y mantenido por MySQL AB. MySQL AB es una compañía comercial, fundada por desarrolladores de MySQL.

¹ Las características de PHP y MySQL se han obtenido del manual de referencia.

MySQL es una base de datos relacional y fue originalmente desarrollado para manejar grandes bases de datos mucho más rápido que con otras soluciones existentes y ha sido utilizada con éxito en muchos entornos de producción de alta demanda durante varios años. A pesar del constante desarrollo, el Servidor MySQL ofrece hoy en día una rica y útil serie de funciones. Su conectividad, velocidad y seguridad hacen del Servidor MySQL altamente apropiado para acceder a bases de datos en Internet.

MySQL es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones.

Parte de la popularidad de MySQL se debe también al hecho de que MySQL puede ser utilizada o modificada por cualquier persona o empresa sin ningún costo. El programa incorpora dos tipos de licencias:

1. La licencia GNU GPL: la cual permite a cualquier persona, empresa o entidad usar el programa sin ninguna restricción. También se da la libertad de modificar el producto y nuevamente redistribuirlo bajo la misma licencia. Esta licencia se caracteriza por ser completamente gratuita.
2. MySQL también incorpora una licencia comercial con la cual las empresas pueden redistribuir el producto bajo sus propios términos. La licencia sin embargo tiene un precio pero no es costosa comparada con licencias de bases de datos comerciales como SQL Server de Microsoft.

Estos dos tipos de licencias se aplican tanto al servidor como a las interfaces y librerías clientes como el C client library, mysqladmin, MySQLCC, mysqldump, libmysqlclient, MySQL Connector/ODBC y J. Y no se aplican a la documentación producida por MySQL AB la cual está bajo la licencia de propiedad intelectual.

Anexo F

Documentación del proyecto

F.1. Estudio preliminar de visores

Para poder determinar los requisitos de la aplicación, y en especial del visor se ha realizado un estudio preliminar de algunos visores existentes en el mercado de aplicaciones.

La comparativa entre visores se ha realizado cargando un documento y una vez cargado se han hecho uso de las funciones disponibles.

Los visores analizados han sido: Adobe Reader, GoodReader y Stanza, todos en su versión para iPad y se destacan las siguientes características detectadas:

Adobe Reader: es el lector por excelencia de documentos en formato pdf.

- ▶ Zoom: si, con doble pulsación con un dedo se acerca y se aleja el documento. Se puede desplazar la pantalla una vez activado el zoom.
- ▶ Lectura: desplazando el dedo arriba y/o abajo y al revés se avanza y retrocede en el documento. También se puede avanzar y retroceder página desplazando el dedo de derecha a izquierda y viceversa o con pulsaciones en parte derecha o izquierda.
- ▶ Índice: no posee. Presenta la posibilidad de ir a una página concreta.
- ▶ Ayuda: no posee. Se descubre la funcionalidad entrando en cada opción de los menús.
- ▶ Interfaz: intuitiva, con pulsación en el centro se accede a las opciones del visor. En general fácil manejo.

GoodReader: es un programa de propósito general, permite crear carpetas, renombrar documentos, permite cargar y descargar contenido del iPad y posee un visor de documentos al nivel de Adobe Reader.

- ▶ Zoom: si, con doble pulsación de un dedo se acerca hasta un máximo y con doble pulsación con dos dedos se aleja hasta la visión inicial.
- ▶ Lectura: desplazando el dedo de derecha a izquierda y viceversa o con pulsaciones en la parte derecha e izquierda de la pantalla avanza y retrocede páginas respectivamente.
- ▶ Índice: no posee, pero permite acceder a una página determinada.
- ▶ Ayuda: solo la primera vez que se accede al visor.
- ▶ Interfaz: intuitiva, uso básico como Adobe Reader, con pulsación central se accede a las opciones del visor. Fácil manejo pero exceso de opciones que normalmente no se usan.

Stanza: es una aplicación cuya principal función es la de almacenar libros y luego reproducirlos. Los formatos que suele emplear son pdf y epub.

- ▶ Zoom: si, con doble pulsación con un dedo se acerca y se aleja el documento. Se puede desplazar la pantalla una vez activado el zoom.
- ▶ Lectura: desplazando el dedo de derecha a izquierda y viceversa o con pulsaciones en la parte derecha e izquierda de la pantalla avanza y retrocede respectivamente.
- ▶ Índice: si, se puede acceder al capítulo deseado, así como a la página deseada del documento.
- ▶ Ayuda: si, tras pulsar en el botón central, existe un botón que carga una pantalla en la que se indica con el nombre de la opción qué es lo que se puede realizar al pulsar sobre ella.
- ▶ Interfaz: intuitiva, uso básico como los otros dos visores, con pulsación central se accede a las opciones del visor. Fácil manejo, opciones las justas y con buena funcionalidad. Los documentos tienen un aspecto más anticuado.

Una vez analizadas las tres opciones se han escogido para ser implementadas las siguientes:

- ▶ Zoom: si, con doble pulsación con dos dedos se acerca y se aleja el documento. Se puede desplazar la pantalla una vez activado el zoom.

Se ha seleccionado esta opción puesto que es común en dos visores y el tercero activa el zoom de forma similar.

- ▶ **Lectura:** desplazando el dedo de derecha a izquierda y viceversa o con pulsaciones en la parte derecha e izquierda de la pantalla se avanza y se retrocede página respectivamente. Todos los visores tienen esta capacidad.
- ▶ **Índice:** se ha creado un menú lateral a través del cual se puede acceder al índice de textos, vídeos y aplicaciones.
- ▶ **Ayuda:** a través del menú lateral comentado se puede cargar en cualquier momento la ayuda para la navegación con el visor. El reproductor de vídeos es intuitivo y no requiere de una ayuda especial para su manejo y las aplicaciones tienen insertados los controles que permiten interactuar con ellas o volver al visor. Para el videojuego se ha creado dentro del menú principal su propia ayuda.
- ▶ **Interfaz:** se ha creado una interfaz de comunicación con el usuario sencilla, se han implementado los controles básicos y la ayuda a través de un menú lateral que se puede ocultar o mostrar en cualquier momento. Este mismo menú lateral permite acceder directamente a los vídeos o aplicaciones en los momentos en los que se hace explicación directa de ellos o tiene relevancia su uso. Se ha pensado en introducir más posibilidades pero al final se han desechado debido a que complicaban en exceso la interfaz y no aportaban grandes usos al visor.

F.2. Especificación de requisitos

Se han detectado los siguientes requisitos a partir de las necesidades del usuario del curso y del videojuego.

Requisitos funcionales:

Requisito	Descripción
RF1	El visor deberá hacer zoom sobre el documento cuando el usuario pulse con dos dedos a la vez sobre la pantalla.

Requisito	Descripción
RF2	El visor deberá salir del modo zoom al volver a pulsar con dos dedos a la vez sobre la pantalla.
RF3	Únicamente al estar en modo zoom, se deberá permitir mover el documento arrastrando un dedo hasta alcanzar los límites del mismo.
RF4	Se debe mostrar una zona con forma de cinta roja que será la que activará la aparición del menú de opciones cuando el usuario pulse sobre ella.
RF5	De igual forma que en el requisito anterior, al pulsar de nuevo sobre la cinta roja el menú de opciones deberá desaparecer.
RF6	Si el menú de opciones está desplegado al avanzar o retroceder una página, este deberá desaparecer.
RF7	Al desplegarse el menú de opciones se deberá mostrar un índice de temas.
RF8	El visor realizará la carga del índice de temas desde el menú que aparece en la interfaz. Esta carga se realizará al pulsar sobre el botón 'Índice'.
RF9	Al desplegarse el menú de opciones se deberá mostrar un índice de vídeos.
RF10	El visor realizará la carga del índice de vídeos desde el menú que aparece en la interfaz. Esta carga se realizará al pulsar sobre el botón 'Vídeos'.
RF11	Al desplegarse el menú de opciones se deberá mostrar un índice de aplicaciones.
RF12	El visor realizará la carga del índice de aplicaciones desde el menú que aparece en la interfaz. Esta carga se realizará al pulsar sobre el botón 'Ejemplos'.
RF13	Al desplegarse el menú de opciones se deberá mostrar una opción de ayuda al usuario. Al pulsar sobre el botón 'Ayuda' se cargará esta.
RF14	Al salir de la ayuda cargada desde el menú de opciones se deberá volver a la página en la que se encontraba el usuario.

Requisito	Descripción
RF15	El visor debe cargar un tema determinado del curso al ser seleccionado desde el índice de temas.
RF16	El visor debe cargar un vídeo determinado del curso al ser seleccionado desde el índice de vídeos.
RF17	El visor debe cargar una aplicación determinada del curso al ser seleccionada desde el índice de aplicaciones.
RF18	Desde el índice de temas se deberá poder cargar el índice de vídeos, de aplicaciones y la ayuda.
RF19	Desde el índice de vídeos y aplicaciones se deberá poder volver al índice de temas.
RF20	Al pulsar con un dedo en la parte derecha de la pantalla se deberá avanzar una página. Si se alcanza fin de tema, se deberá cargar el siguiente. Si es el final del curso, no se hará nada.
RF21	Al pulsar con un dedo en la parte izquierda de la pantalla se deberá retroceder una página. Si se alcanza el principio del tema, se deberá cargar el tema anterior. Si se está en la primera página del primer tema, se cargará el índice de temas.
RF22	El visor ocultará la cinta roja de acceso al menú de opciones al principio de cada tema y en el índice de contenidos del tema.
RF23	El visor deberá reproducir un vídeo desde el menú de opciones desplegado o sin desplegar en determinadas páginas marcadas.
RF24	El visor deberá reproducir una aplicación interactiva desde el menú de opciones desplegado o sin desplegar en determinadas páginas marcadas.
RF25	Al volver de un vídeo, el visor deberá mostrar la página del texto que esté relacionada directamente con el vídeo y desde la cual también se puede reproducir.
RF26	Al volver de una aplicación interactiva (incluido el videojuego), el visor deberá mostrar la página del texto que esté relacionada directamente con la aplicación y desde la cual también se puede ejecutar.

Requisito	Descripción
RF27	Todo el contenido, es decir, vídeos, aplicaciones interactivas y videojuego se debe poder ejecutar sin salir de la aplicación.
RF28	Desde el reproductor de vídeos se deberá poder volver al visor.
RF29	Desde la aplicación en ejecución se deberá poder volver al visor.
RF30	El personaje se debe mover mediante pulsaciones por la pantalla.
RF31	El personaje deberá poder saltar y hacer un salto especial si está en el aire.
RF32	Se debe distinguir la acción que realiza el personaje y ejecutar la animación correspondiente.
RF33	Se debe distinguir la acción que realiza el personaje y reproducir el sonido correspondiente.
RF34	El videojuego debe reproducir música que comience de forma aleatoria.
RF35	El personaje eliminará a los enemigos al caer encima de ellos, como ocurre en los juegos normales de plataformas.
RF36	El personaje perderá una vida si entra en contacto por una parte que no sea la superior con un enemigo y también perderá una vida si se ve afectado por ciertos elementos.
RF37	El juego debe tener y gestionar dos niveles de dificultad.
RF38	El juego debe tener, cargar y gestionar tres niveles o escenas.
RF39	El juego deberá tener otros elementos con los que interaccionar y producir algún resultado en el juego.
RF40	El juego debe tener y gestionar un sistema de puntuaciones interno.
RF41	El juego deberá almacenar máximas puntuaciones del usuario.
RF42	El juego deberá mostrar estadísticas de las partidas.

Requisito	Descripción
RF43	Se deberá mostrar una ayuda sobre la interfaz y el movimiento.
RF44	El jugador podrá registrarse en el sistema. Sin registro no deberá haber acceso al juego.
RF45	El juego permitirá cambiar los datos del jugador.
RF46	Al terminar una partida, el juego deberá comunicar a una aplicación web la puntuación del jugador.
RF47	Se deberá poder borrar todos los datos almacenados.
RF48	Se permitirá volver al visor de la aplicación principal.

Tabla F.1. Requisitos funcionales de la aplicación

Requisitos no funcionales:

Requisito	Descripción
RNF1	El visor se debe poder ejecutar en un iPad
RNF2	La interfaz debe estar adaptada a las características de navegación y usabilidad de dispositivos móviles.
RNF3	El usuario debe de interaccionar desde un entorno sencillo e intuitivo.
RNF4	La aplicación debe funcionar en dispositivos iOS 5.0 o superior.
RNF5	Los vídeos deberán almacenarse en un servidor web externo.
RNF6	Todos los datos de las partidas y de usuarios del top 100 se almacenarán en una base de datos MySQL alojada en un servidor remoto.
RNF7	La web será funcional desde los siguientes navegadores, Safari 5.0 o superior, Firefox 4.0 o superior y Google Chrome 29.0 o superior.

Requisito	Descripción
RNF8	La web de puntuaciones se almacenará en el mismo servidor que la base de datos.
RNF9	El videojuego completo se debe poder ejecutar en un iPad.
RNF10	El videojuego debe tener alguna historia de fondo.
RNF11	Se deberá intentar crear ambientación adecuada para cada nivel del juego.
RNF12	Cada nivel del juego debe tener su propia historia dentro de la historia general.
RNF13	Debe haber una variedad de enemigos en cuanto a aspecto externo.
RNF14	Deben de existir animaciones distintas en el personaje principal y variedad de ellas en los enemigos.
RNF15	Debe de existir variedad de sonidos para los efectos y para la música de ambiente.

Tabla F.2. Requisitos no funcionales de la aplicación

F.3. Planificación

F.3.1. Planificación inicial

La planificación del proyecto se realizó durante las primeras semanas de octubre de 2012. La planificación inicial se muestra en la siguiente tabla:

Actividad	Horas estimadas	Porcentaje
Aprendizaje	80	13,3
Análisis	60	10,0
Diseño	100	16,7
Implementación	200	33,3

Actividad	Horas estimadas	Porcentaje
Pruebas	80	13,3
Documentación	80	13,3
Total	600	100

Tabla F.3. Planificación inicial

La idea era la de programar una carga de trabajo de unas 100 horas de trabajo al mes, por lo que el proyecto debería durar unos 6 meses.

Los meses se considerarían de 4 semanas, así por semana se trabajarían 25 horas.

A la semana se trabajarían 5 días, por lo que la carga de trabajo al día debería ser de 5 horas.

En la figura F.1 se muestra esta distribución en porcentaje.

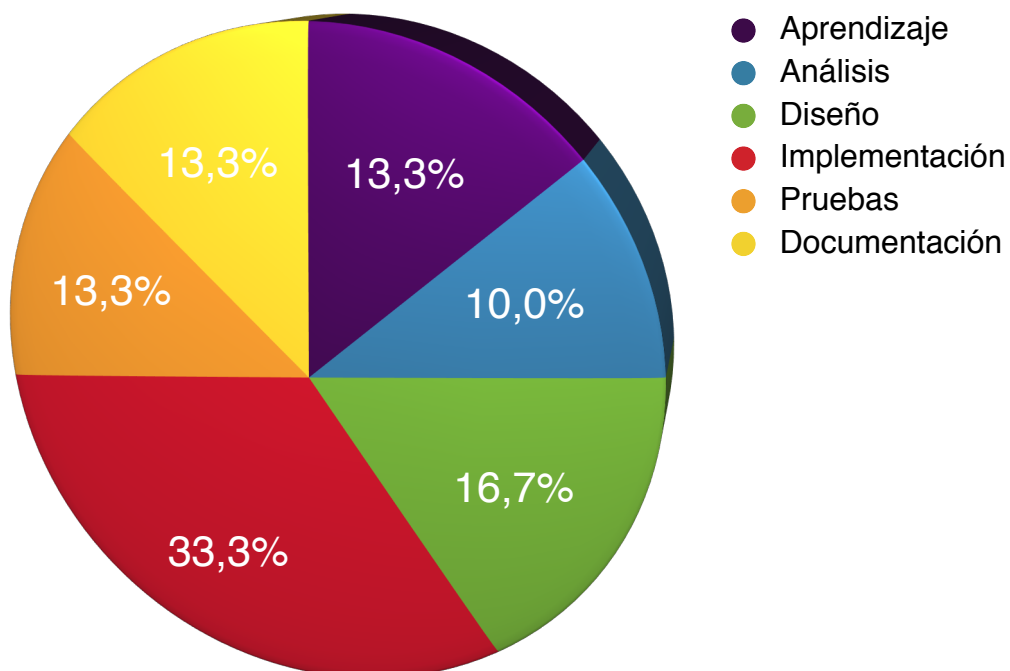


Figura F.1. Gráfica con el tiempo estimado en porcentaje

F.3.2. Planificación real

A continuación se muestra la distribución real del tiempo empleado en cada una de las actividades y la representación gráfica en porcentaje:

Actividad	Horas reales	Porcentaje
Aprendizaje	140	17,5
Análisis	80	10,0
Diseño	120	15,0
Implementación	300	37,5
Pruebas	80	10,0
Documentación	80	10,0
Total	800	100

Tabla F.4. Tiempo real empleado

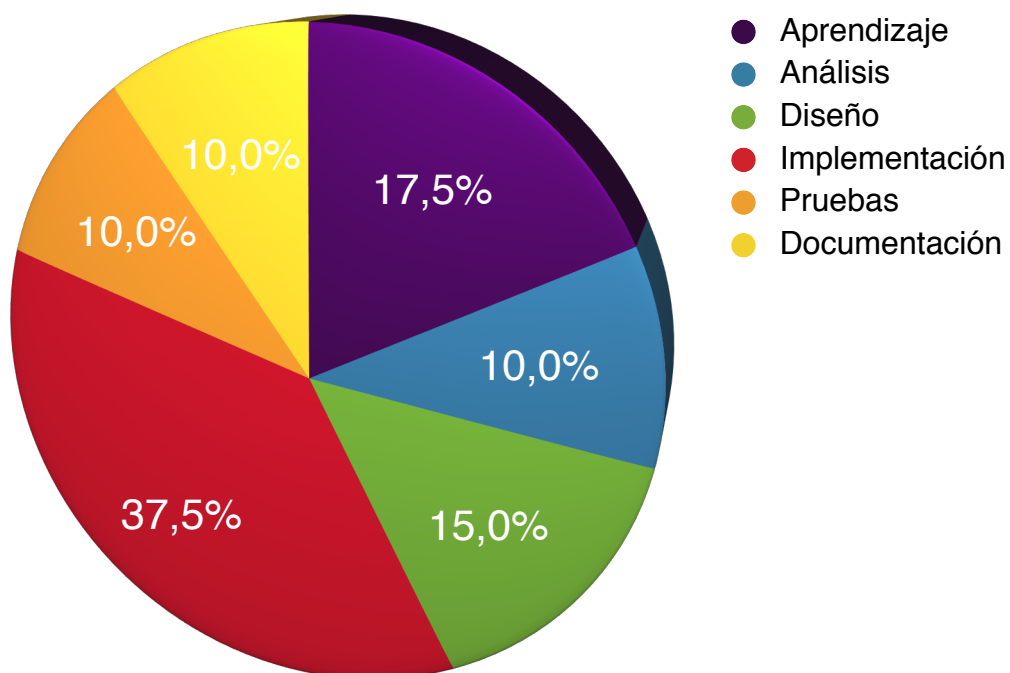


Figura F.2. Gráfica con el tiempo real en porcentaje

Como se puede observar en las tablas F.3 y F.4, el tiempo real difiere del estimado en el tiempo destinado al aprendizaje, análisis, diseño e implementación. Los motivos de estas diferencias han sido:

- ▶ Comienzo del proyecto con la coincidencia de la llegada de las Fiestas del Pilar que motivaron un ligero retraso en el inicio del proyecto.
- ▶ En cuanto al aumento de tiempo de aprendizaje, análisis, diseño e implementación, se debió a la detección de nuevas necesidades y mejoras que hubo que analizar, diseñar, implementar y probar. Por ejemplo, se modificó el diseño de la interfaz porque la original presentaba demasiadas opciones que no aportaban una gran mejora y sí dificultaban la sencillez y la adaptación a su uso.

Otro cambio que se realizó fue el del aspecto del curso, se cambió la paleta de colores empleada mejorando su visualización. También se aumentó el tamaño de letra para que el curso se adaptase a una presentación utilizando un cañón o directamente la visualización en un iPad.

Mejora de las aplicaciones interactivas.

Se crearon nuevos temas y se ampliaron algunos para explicar la creación del juego debido a un rediseño de escenarios, nuevos modelos, funcionalidades del entorno, y ampliación de las opciones del juego.

El trabajo se ha excedido también en horas debido a la novedad del proyecto para el autor del mismo en algunos aspectos como el proceso de modelado de objetos en 3D, creación y modificación de texturas, creación de animaciones, en resumen, prácticamente el proceso completo de creación de un videojuego en 3D desde cero.

- ▶ En el tiempo de implementación se ha incluido la creación de los textos (búsqueda de documentación, selección, adaptación al nivel del curso), la creación del visor, creación de los vídeos, creación de los ejemplos de los textos, creación de las aplicaciones interactivas y creación del videojuego completo.

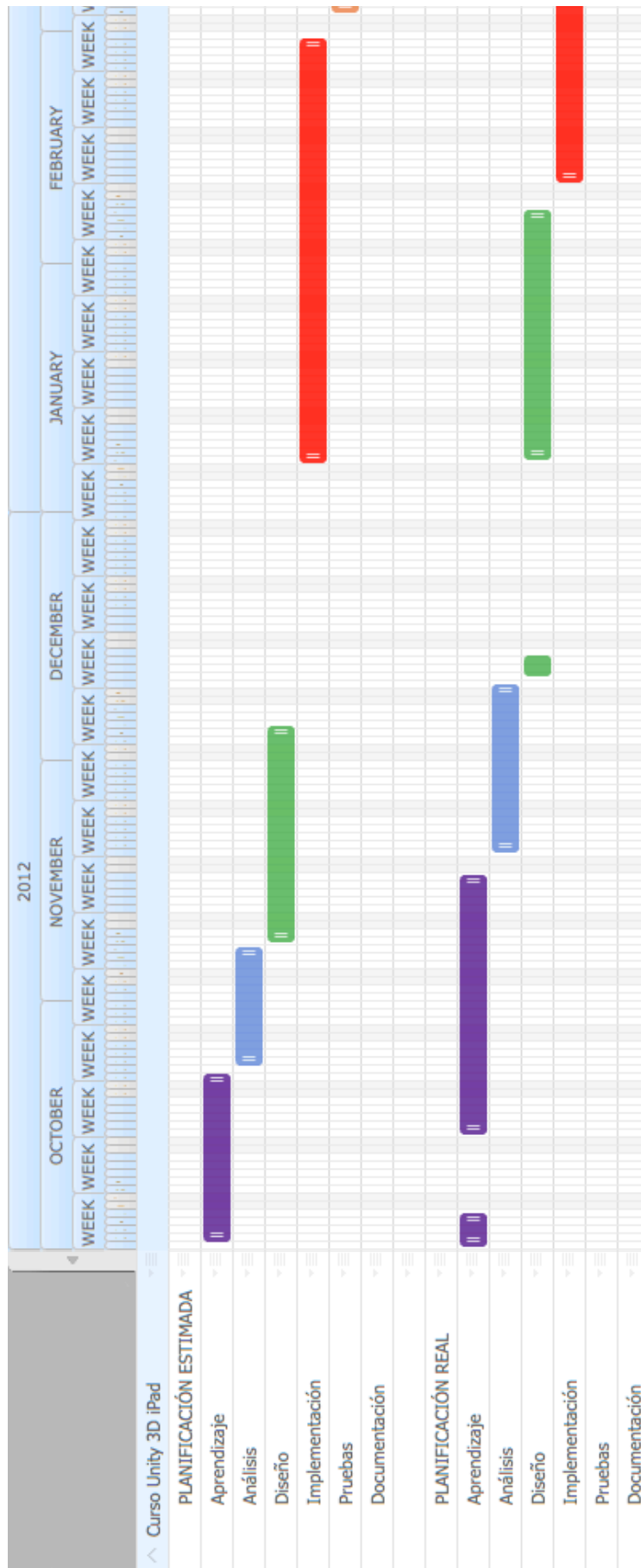


Figura F.3a. Diagrama de Gantt del tiempo estimado y real del proyecto

En las figuras F.3a y F.3b se muestra mediante un diagrama de Gantt la distribución del tiempo estimado y real del proyecto.

En el diagrama de Gantt con la planificación real se observa la parada durante las Fiestas del Pilar y de Navidad. También se observa que el proyecto comenzó en octubre de 2012 y finalizó en julio de 2013.

F.4. Análisis y diseño del videojuego

F.4.1. Casos de uso

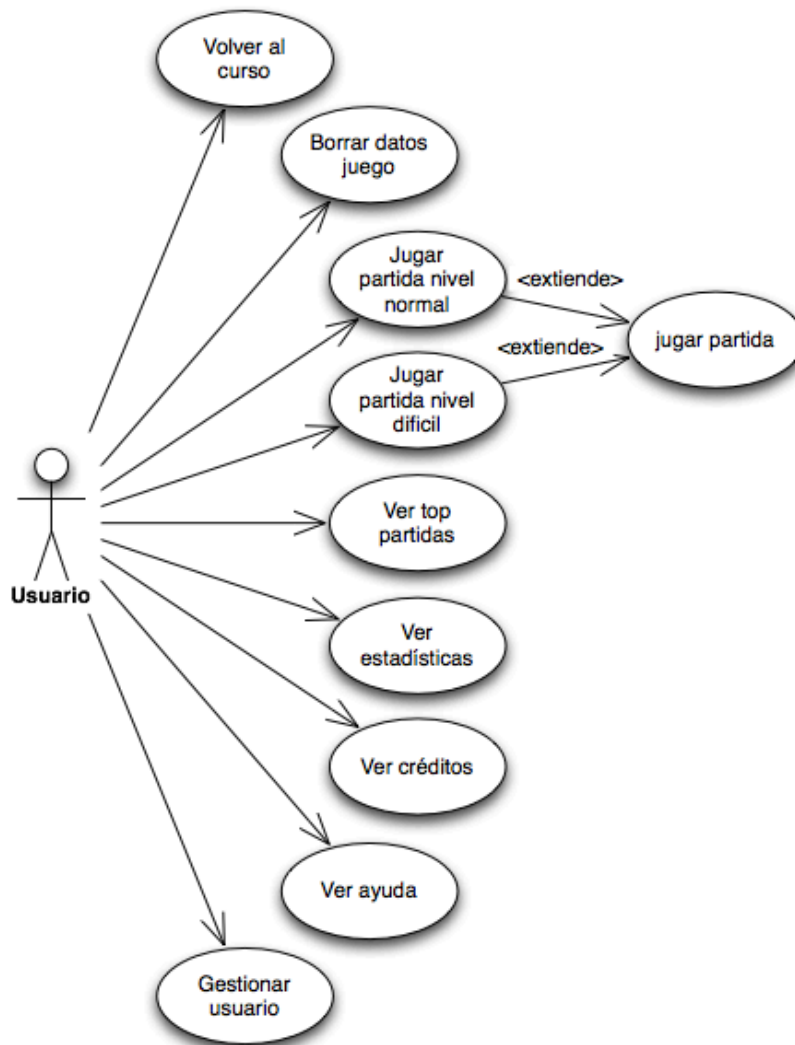


Figura F.4. Casos de uso del videojuego

En la figura F.4 se observan los casos de uso de un usuario del videojuego. Representan a un jugador interactuando con el videojuego de distintas formas:

- ▶ Volviendo a la posición del curso donde se termina de explicar la realización del videojuego.
- ▶ Borrando todos los datos que hay en el juego, algo que provocará la solicitud de un nuevo usuario por parte del videojuego.
- ▶ Jugar una partida, en nivel normal o difícil.
- ▶ Consultar el top 5 de las mejores partidas en nivel normal y difícil.
- ▶ Consultar las estadísticas de todas las partidas jugadas por nivel.
- ▶ Ver los créditos del juego.
- ▶ Consultar la ayuda del juego.
- ▶ Gestionar los datos del usuario y modificarlos si se desea.

F.4.2. Diagrama de flujo del videojuego

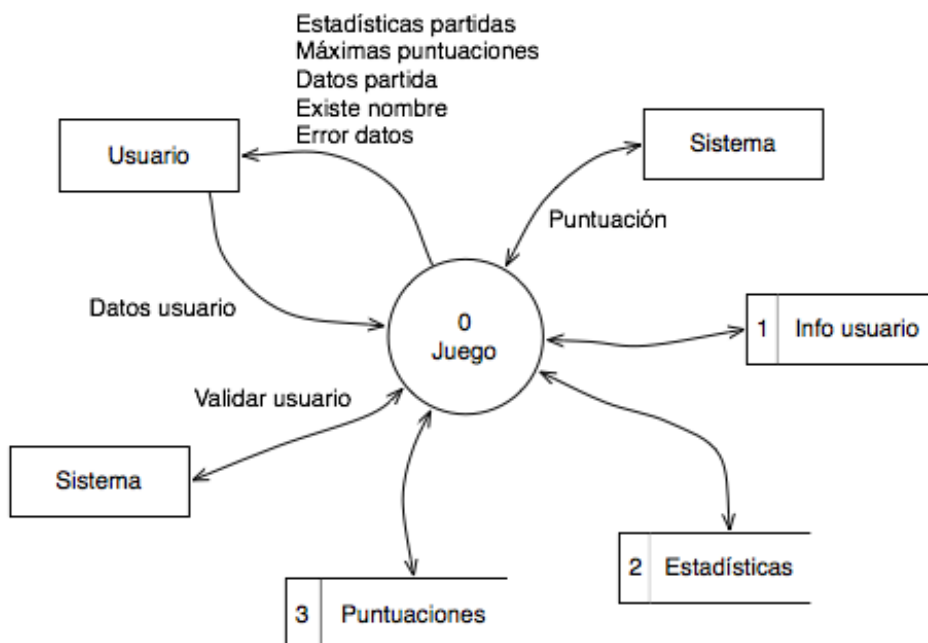


Figura F.5. Diagrama de flujo de datos de nivel 0 del juego

Como se comentó en la memoria el DFD de la figura F.5 muestra los siguientes movimientos de datos:

- ▶ El usuario proporciona al juego sus datos personales (Datos usuario).
- ▶ El juego muestra al usuario las estadísticas de todas las partidas jugadas (Estadísticas partidas). También se puede mostrar al usuario las máximas puntuaciones locales Máximas puntuaciones). Se ofrecen los datos de la partida actual (datos partida). Se pueden dar dos casos de error, uno debido a una incorrecta entrada de datos por parte del usuario (Error datos) y la otra porque el usuario ya exista en la base de datos al crear o modificar un usuario (Existe nombre).
- ▶ Validar usuario será lo que se envíe y retorne la base de datos al consultar la existencia de un usuario.
- ▶ Puntuación es la puntuación obtenida en la partida, se debe comprobar con la aplicación web si está dentro del top 100, si es un record se actualiza la base de datos de máximas puntuaciones.

En la figura F.6 se muestran todos los procesos que describen el proceso principal:

- ▶ 'Tratar usuario' es el proceso encargado de validar la entrada del usuario y de comprobar que este no existe en el sistema.
- ▶ 'Jugar partida' es el proceso que se encarga de gestionar todos los datos de la partida actual. Al terminar una partida se almacenan los datos estadísticos correspondientes y se comprueba que la puntuación sea un record local y del top 100, si es un record local se almacenan los datos localmente y si es un record del top 100, se almacenan los datos en la base de datos.
- ▶ El proceso 'Mostrar información' es el encargado de recoger los datos estadísticos y del top 5 de partidas y mostrarlos de forma adecuada al usuario.

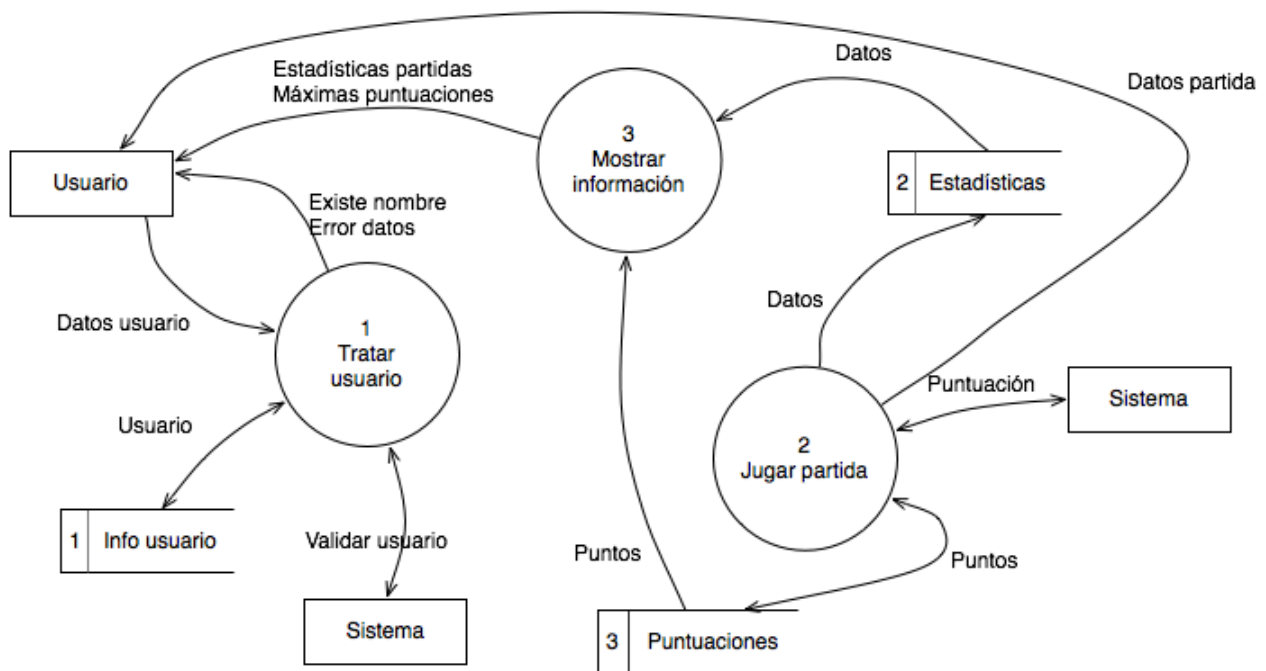


Figura F.6. Diagrama de flujo de datos de nivel 1 del juego

En la figura F.7 se muestra el diagrama de flujo de datos de nivel 2 para el proceso 'Tratar usuario', este proceso se descompone en dos:

- ▶ Procesar entrada: lo que hace es validar la entrada del usuario y eliminar entradas erróneas.
- ▶ Actualizar usuario: una vez validada una entrada se comprueba si el usuario introducido existe en la base de datos, si no existe se crea localmente y en la base de datos, si existe el nombre se indica al usuario.

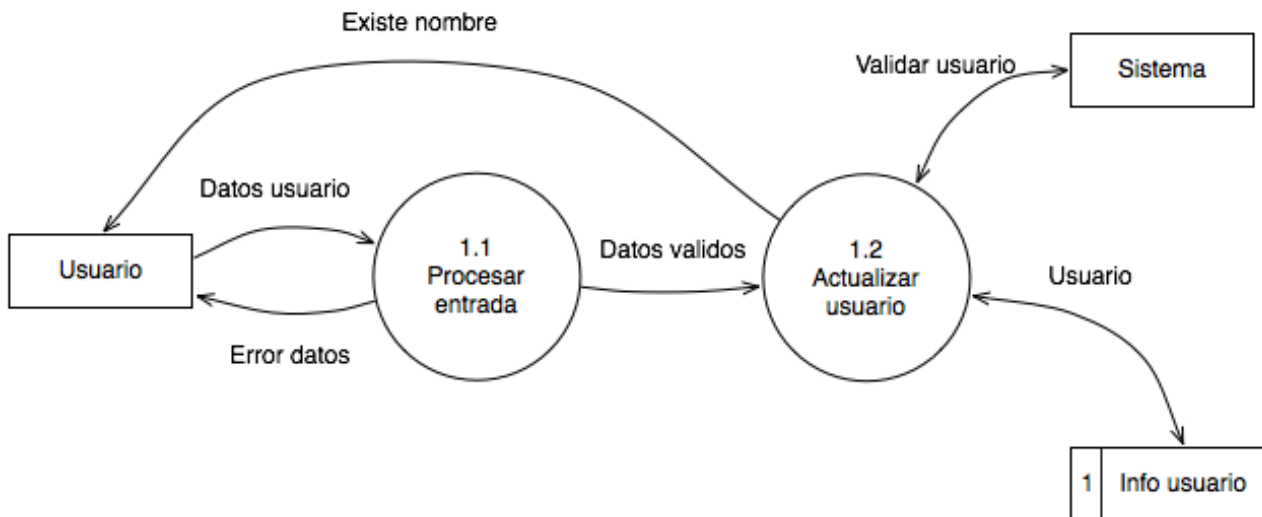


Figura F.7. Diagrama de flujo de datos de nivel 2. Especificación del proceso 1: Tratar usuario

En la figura F.8 se muestra el diagrama de flujo de datos de nivel 2 para el proceso 'Jugar partida', este proceso se descompone en dos:

- ▶ El proceso 'Tratar nivel' se encarga de guardar los datos estadísticos de la partida y de generar la puntuación.
- ▶ El proceso 'Comprobar record' con los datos de la partida comprueba si es una puntuación máxima local si es así almacena los datos. También comprueba si es un máximo del top 100 puntuaciones, si es así actualiza la base de datos de las puntuaciones y muestra la partida en la página web de puntuaciones.

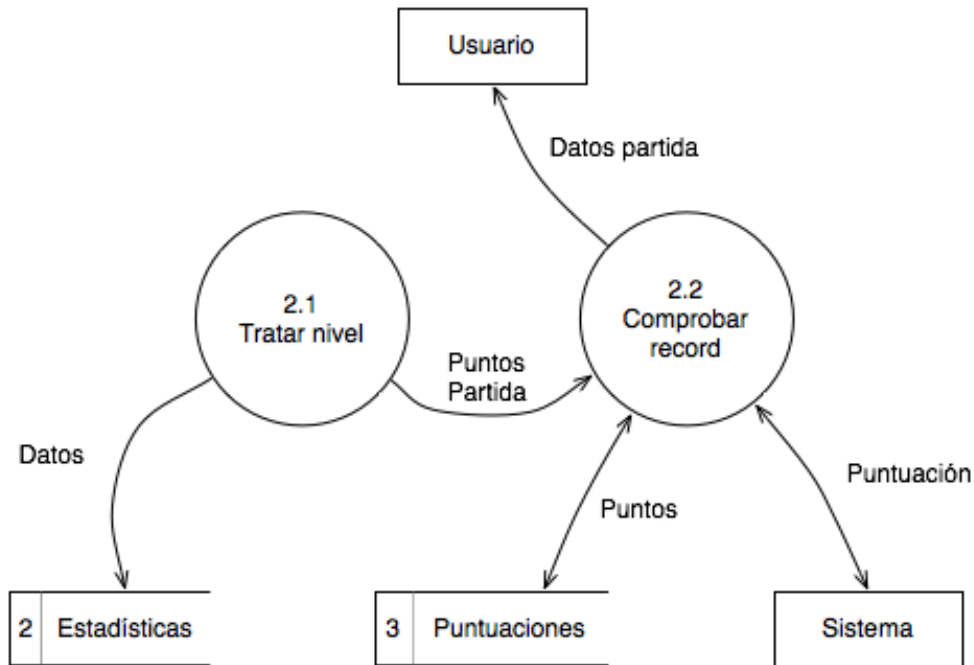


Figura F.8. Diagrama de flujo de datos de nivel 2. Especificación del proceso 2: Jugar partida

F.4.3. Diagrama de clases

Con el diagrama de clases se detalla la estructura estática del sistema, mostrando los objetos de la misma, sus atributos y métodos más relevantes y las relaciones entre objetos. En la figura F.9 se muestra el diagrama de clases empleado:

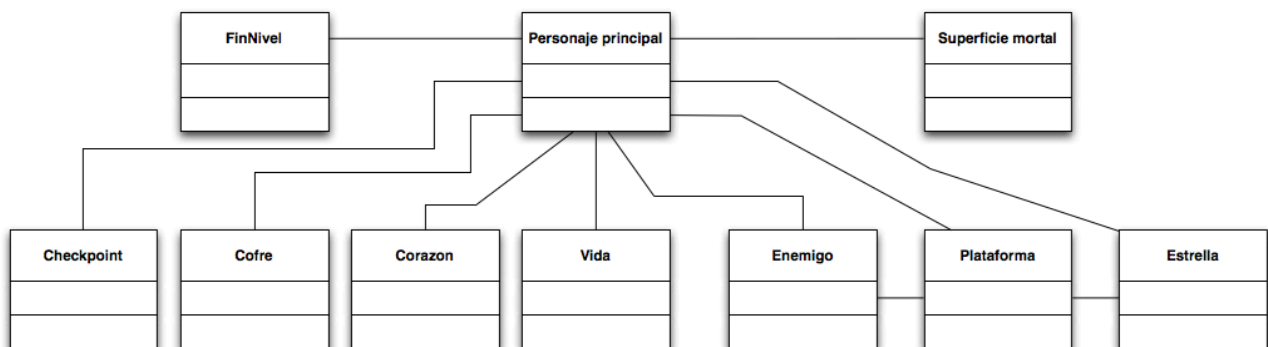


Figura F.9. Diagrama de clases principales del videojuego

A continuación se detallan las clases de la figura F.9.

Personaje principal:

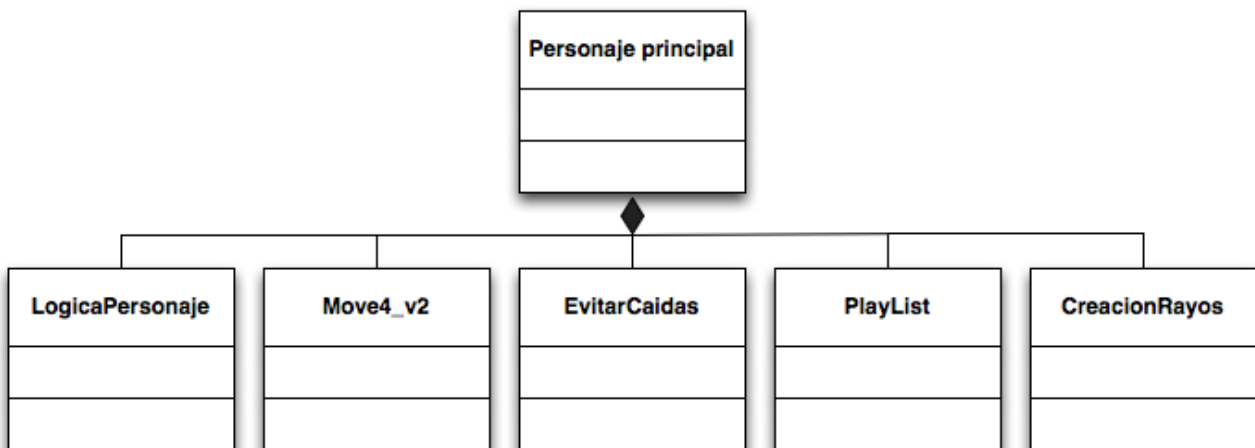


Figura F.10. Diagrama de clases del personaje principal

El personaje principal está formado por cinco clases que se muestran en la figura F.10.

► **LogicaPersonaje**: dispone de una serie de atributos que son utilizados para realizar el control de enemigos muertos, tiempo de juego, variables para efectos de sonido externos al personaje, estrellas y cofres recogidos, vidas restantes y última posición guardada del personaje entre otras.

Los métodos principales son:

- **perderVida()**: que hace que el personaje principal pierda una de las vidas que posee.
- **ganarVida()**: provoca que el jugador aumente en uno el total de vidas si no se ha alcanzado el máximo de vidas posibles (3).
- **reproducirSonido()**: método que ejecuta un sonido como efecto al producirse ciertos eventos.

► **Move4_v2**: es la clase que se encarga del movimiento lateral del personaje y de los saltos.

Los atributos principales son:

- speed: velocidad de movimiento del personaje.
- gravity: valor para la fuerza de la gravedad.
- moveDirection: aplica la velocidad y la dirección del movimiento para que el objeto se desplace.
- jumpSpeed: velocidad de salto del personaje.
- secondJump: permite saber si se ha realizado o no un segundo salto en el aire.
- movimiento: indica si el personaje se está moviendo o está parado.
- direccion: almacena el movimiento del personaje, es decir, si se desplaza hacia la derecha o hacia la izquierda.
- sonidoSalto: efecto de sonido cuando se produce un salto.
- sonidoSaltoDoble: efecto de sonido para el salto doble.
- timanChar: objeto en 3D del personaje.
- atras: se usa para saber si el personaje anda hacia atrás y es necesario girar el cuerpo.

Los métodos principales son:

- Update(): método que se ejecuta cada frame del juego. Cada frame se evalúa si el usuario ha pulsado la pantalla y se procesa la pulsación determinando si es movimiento, parada o salto y aplica la animación necesaria para cada situación. Si es un movimiento desplaza el personaje en la dirección indicada, si es salto aplica la fuerza del salto. La gravedad se aplica cada frame, que hace que el personaje caiga si ha realizado un salto.

En la figura F.14 se muestra un diagrama de estados que explica el movimiento del personaje.

- EvitarCaidas: se encarga de evitar que el personaje caiga de plataformas que poseen un movimiento vertical. Dispone de un atributo que indica si el personaje está en el interior o exterior de la plataforma.

Los métodos principales son:

- OnTriggerStay(): método que se ejecuta al entrar el personaje en contacto con la plataforma y quedarse en ella. Evita que caiga de la plataforma haciendo que el personaje se mueva con ella.

- OnTriggerExit(): método que se ejecuta al dejar de estar en contacto el personaje con la plataforma. Hace que el personaje deje de moverse con la plataforma.

- ▶ Playlist: contiene como atributos el conjunto de canciones a ser ejecutadas y una variable que indica la canción actual que está en reproducción.

El método principal es:

- LoopClips(): comienza de forma aleatoria la reproducción de una canción y al terminar continúa con la siguiente de la lista.

- ▶ CreacionRayos: contiene como atributos:

Atributos principales:

- prefab: referencia al objeto que será creado dinámicamente como rayo en el tercer nivel del juego.

- posiciones: vector que almacena las posiciones donde se generarán de forma aleatoria los rayos.

- timer: temporizador.

- intervaloCreacionRayos: almacena el tiempo cada cuanto se creará un nuevo rayo.

Método principal:

- Update(): crea un rayo cada intervaloCreacionRayos en una de las posiciones indicadas por el vector 'posiciones'.

Enemigos:

La clase que define los enemigos del personaje principal se muestra en la figura F.11. Esta clase dota de comportamiento a los enemigos del juego provocando por ejemplo que el personaje principal pierda una vida.

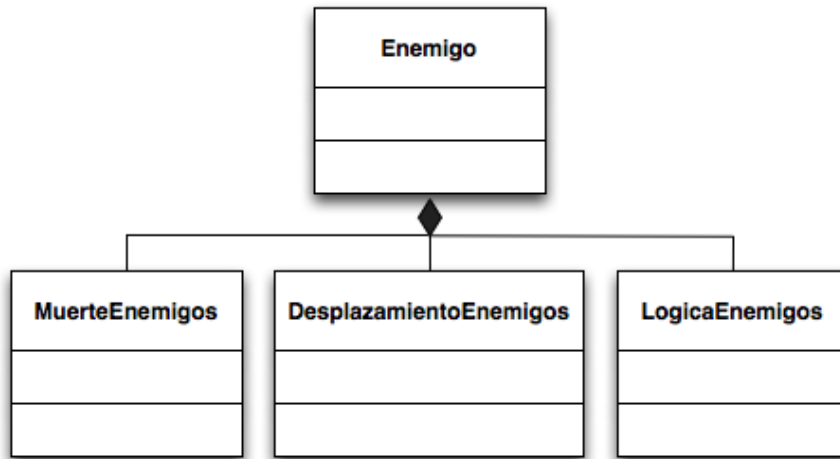


Figura F.11. Diagrama de clases de los enemigos

- ▶ **MuerteEnemigos**: se encarga de eliminar el objeto de la partida y de notificar de la muerte al personaje principal.

Método principal:

- **OnTriggerEnter()**: detecta colisión con un objeto, si el objeto es el jugador, le indica que el número de enemigos muertos debe aumentar en uno y que reproduzca el sonido de muerte de enemigo, a continuación destruye el objeto.

- ▶ **DesplazamientoEnemigos**: realiza el movimiento de los enemigos desde una posición origen a otra destino. Estas posiciones son variables que guardan una referencia a las posiciones. También posee una variable velocidad que determina la velocidad de desplazamiento del enemigo y una variable cambio que indica cuando debe de producirse el giro del enemigo (girando también el modelo).

Método principal:

- **FixedUpdate()**: realiza el movimiento del enemigo de forma independiente a los fotogramas por segundo.

- ▶ **LogicaEnemigos**: detecta la colisión lateral con el personaje y provoca la pérdida de una vida.

Método principal:

- OnTriggerEnter(): al producirse la colisión lateral, se le indica al personaje que pierde una vida y que emita el sonido de contacto con el enemigo.

Los diagramas de secuencia de las figura F.15 y F.16 describen la interacción entre el personaje principal y los enemigos.

Plataformas:

Como se observa en la figura F.12, las plataformas que se han usado en el videojuego pueden ser de varios tipos, las simples que son plataformas fijas, plataformas con desplazamiento lateral, plataformas giratorias y cajas de cartón.

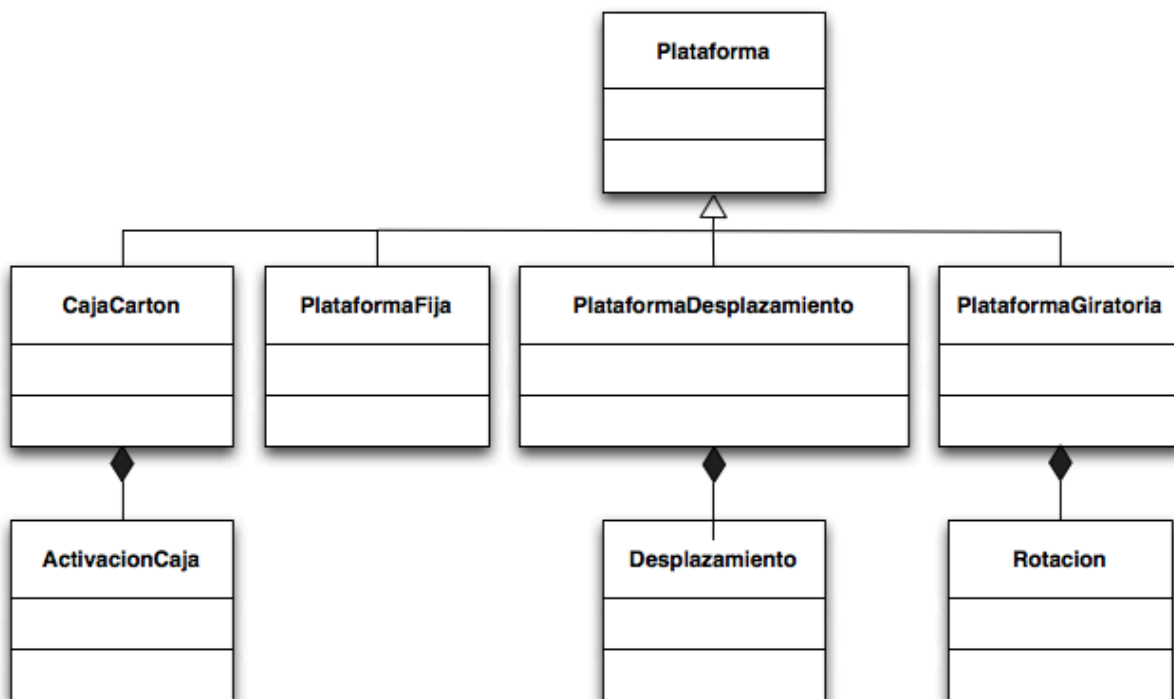


Figura F.12. Diagrama de clases de las plataformas

A continuación se van a comentar las características principales de la clase:

- **ActivacionCaja**: detecta la entrada del personaje principal y provoca su desaparición tras 2 segundos.

- ▶ Rotacion: realiza el giro a una determinada velocidad del objeto en 3D.
- ▶ Desplazamiento: realiza el movimiento de la plataforma desde una posición origen a otra destino. Estas posiciones son variables que guardan una referencia a las posiciones. También posee una variable velocidad que determina la velocidad de desplazamiento y una variable cambio que indica cuando debe de producirse el giro.

Superficies mortales:

Las superficies mortales se muestran en la figura F.13, se puede observar que existen dos tipos de superficies, fijas y con desplazamiento. Ambas comparten una clase LogicaEnemigos que ya se ha descrito en los enemigos. Las plataformas con desplazamiento poseen otras clases que son: Desplazamiento (también comentada), MuertePorLava y TexturaAnimada.

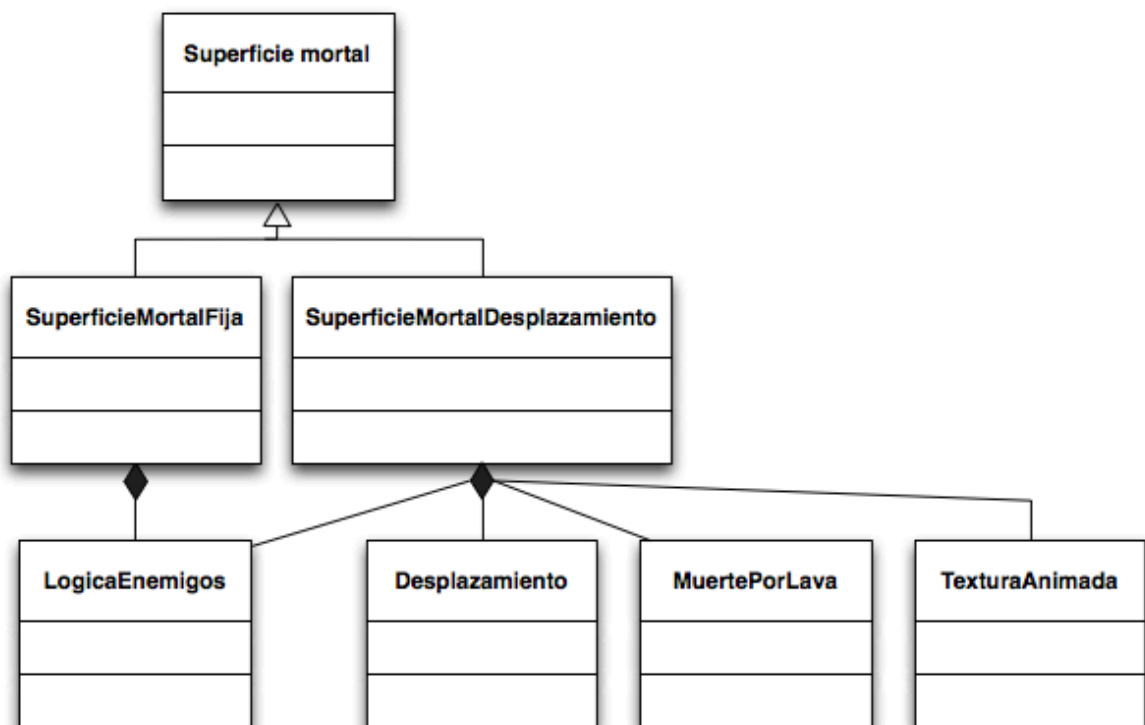


Figura F.13. Diagrama de clases de las superficies mortales

- ▶ MuertePorLava: provoca la pérdida de una vida al producirse el desplazamiento de la superficie mortal. Se muestra el diagrama de secuencia de esta clase en el siguiente apartado, figura F.17.

- ▶ Textura animada: desplaza la textura asignada al objeto para dar la sensación de movimiento.

PlayerPrefs:

Esta clase se comentó que se ha utilizado para almacenar datos entre sesiones o niveles del juego. La clase PlayerPrefs se utiliza para almacenar y recuperar las preferencias de usuario, tiene los siguientes métodos:

- Métodos de creación de claves:

PlayerPrefs.SetInt(): almacena una clave entera asociada a un nombre determinado.

PlayerPrefs.SetFloat(): almacena una clave real asociada a un nombre determinado.

PlayerPrefs.SetString(): almacena una clave de tipo cadena asociada a un nombre determinado.

- Métodos de lectura de claves:

PlayerPrefs.GetInt(): devuelve el entero guardado con el nombre indicado.

PlayerPrefs.GetFloat(): devuelve el real guardado con el nombre indicado.

PlayerPrefs.GetString(): devuelve una cadena guardada con el nombre indicado.

- Método de consulta de claves:

PlayerPrefs.HasKey(): devuelve true si existe la clave con el nombre proporcionado.

- Método de escritura en disco de todas las claves creadas:

PlayerPrefs.Save(): método que guarda en disco todas las preferencias de usuario creadas o modificadas.

- Método de borrado de claves:

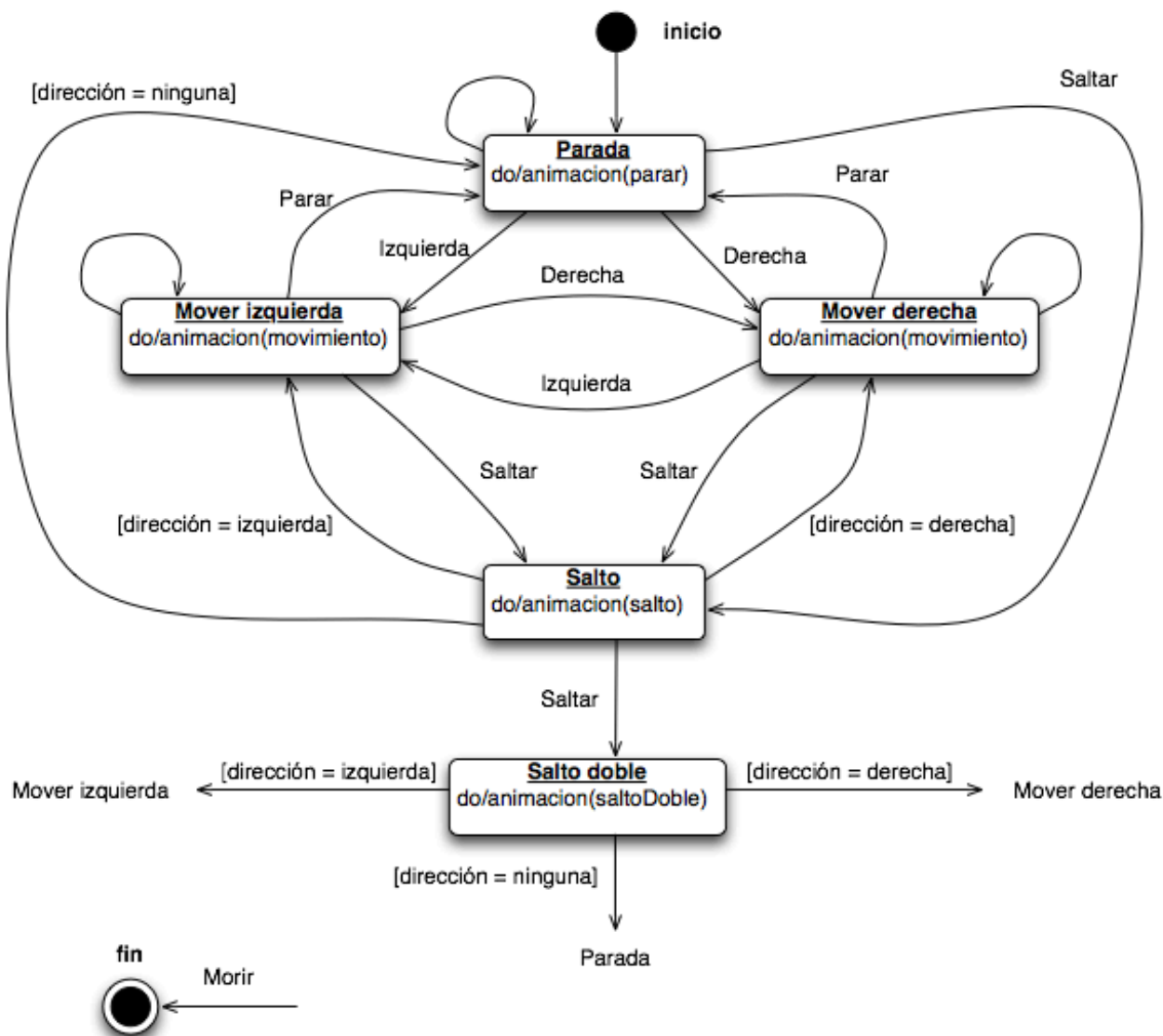
PlayerPrefs.DeleteKey(): borra la clave con el nombre proporcionado.

PlayerPrefs.DeleteAll(): borra todas las claves almacenadas en disco.

En el siguiente apartado mediante diagramas de secuencia se describen los objetos restantes y su comportamiento.

F.4.4. Modelado dinámico

En la figura F.14 se muestra el diagrama de estados que explica el movimiento del personaje principal.



*Desde cualquier estado se puede producir el evento 'Morir' finalizando el diagrama de estados

**Desde 'Salto doble' se produce el mismo cambio de estado con las mismas condiciones que en 'Salto'

Figura F.14. Diagrama de estados del movimiento del personaje

En el diagrama se observa que el estado inicial del personaje es el de 'Parada'. En todos los estados se ejecuta una animación mientras se esté en ese estado. Si se pulsa a la derecha del personaje, se cambia de estado y se pasa al estado 'Mover derecha' que provoca que el personaje se desplace hacia la derecha hasta que se detecte una nueva interacción y se cambie de estado. Si se pulsa a la izquierda del personaje este cambiará de dirección si no se encontraba en el estado 'Mover izquierda' y desplazará al personaje hacia la izquierda. Desde cualquiera de los estados anteriores, si se pulsa con dos dedos sobre la pantalla, se cambiará al estado 'Salto', al terminar se volverá al estado del que se partía. Desde el estado 'Salto' se puede ir también al estado 'Salto doble' que provoca un nuevo salto del personaje en el aire, una vez terminado se volverá al estado previo a los saltos. Solo es

posible dar un salto doble si ya se estaba en el aire. Por último comentar que desde cualquier estado se puede producir la muerte del personaje que se muestra en el diagrama como la salida del mismo.

Al producirse la muerte del personaje, se sigue almacenando la dirección, por lo que si se estaba desplazando hacia la derecha o izquierda, al volver a aparecer el personaje este se seguirá desplazando en esa dirección. En el diagrama por simplicidad se muestra el movimiento del personaje al comienzo de la partida, pero al morir se empezará en el último estado en el que se encontraba.

El siguiente diagrama que se muestra es el de interacción entre el personaje principal y un enemigo provocando la muerte de este último.

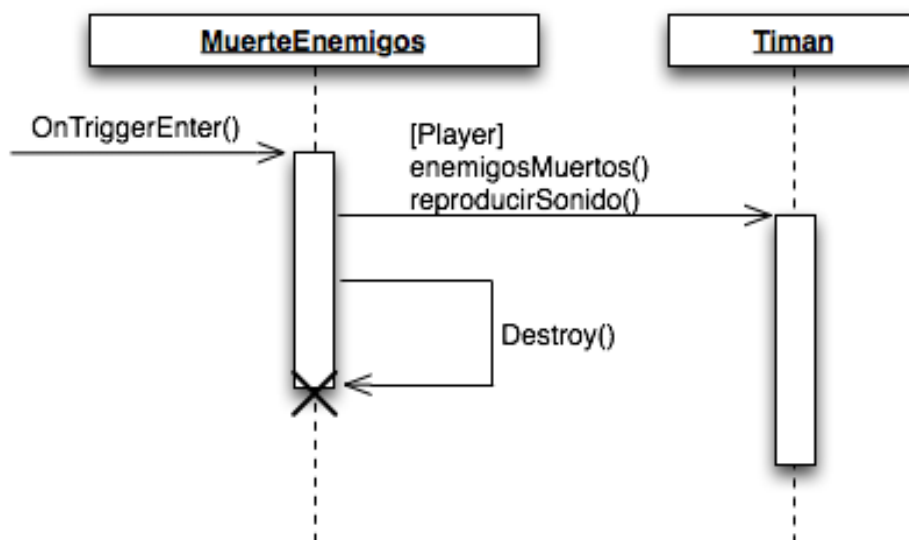


Figura F.15. Diagrama de secuencia de la interacción entre enemigos y personaje principal que provoca la muerte de un enemigo

Al alcanzar por la parte superior al enemigo, se activa la función OnTriggerEnter() que hace que el enemigo envíe al personaje dos mensajes, enemigosMuertos() que aumenta en uno el número de enemigos muertos y reproducirSonido() que provoca un efecto de sonido asociado a la muerte del enemigo. Tras estos dos mensajes se elimina al enemigo de la partida.

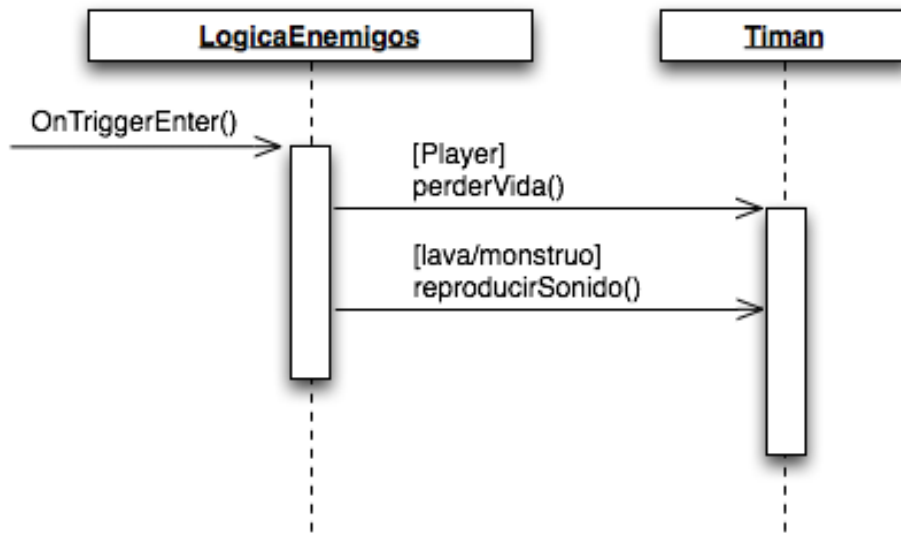


Figura F.16. Diagrama de secuencia de la interacción entre enemigos y personaje principal que provoca la pérdida de una vida del personaje

En la figura F.16 se muestra la otra interacción posible entre el personaje principal y un enemigo. Si se produce un contacto lateral, el enemigo envía el mensaje `perderVida()` al personaje principal que le provoca la pérdida de una vida, a continuación se indica al personaje que reproduzca el sonido relacionado con el evento de pérdida de una vida. Este mismo diagrama explica la situación por la que el personaje toca una superficie mortal provocando los mismos efectos que los que se acaban de describir para el enemigo.

En la figura F.17 se describe la situación por la que una superficie mortal alcanza en su desplazamiento al personaje principal, si se da esa situación, se envía el mensaje de perder una vida al personaje y la de la reproducción del sonido asociado al evento.

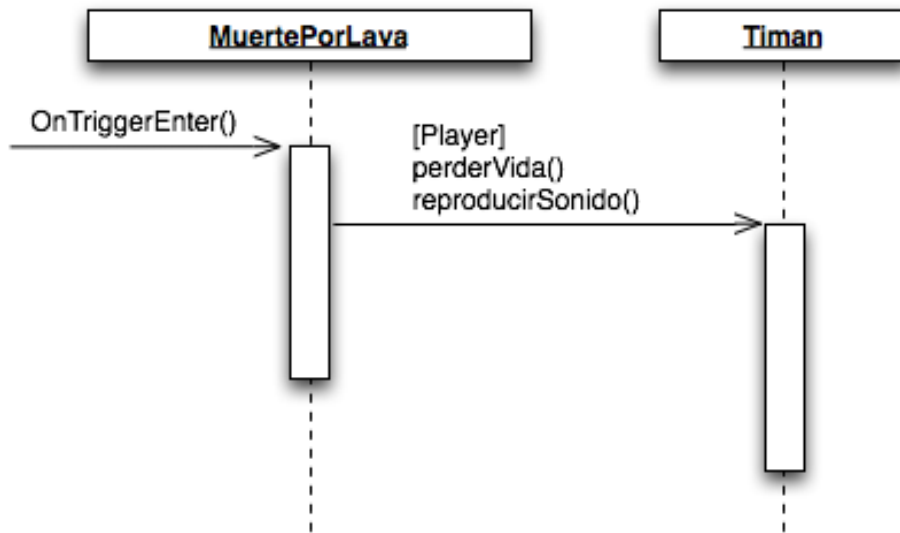


Figura F.17. Diagrama de secuencia que describe la situación provocada por la entrada en contacto de una superficie mortal en su desplazamiento con el personaje principal

Otro tipo de interacción con objetos del juego son los que se van a mostrar a continuación, son objetos que al pasar por encima de ellos se eliminan de la partida y producen el incremento de la puntuación (figura F.18, cofres y estrellas), recuperación de vidas (figura F.19, corazones) o punto de guardado (figura F.20, checkpoint).

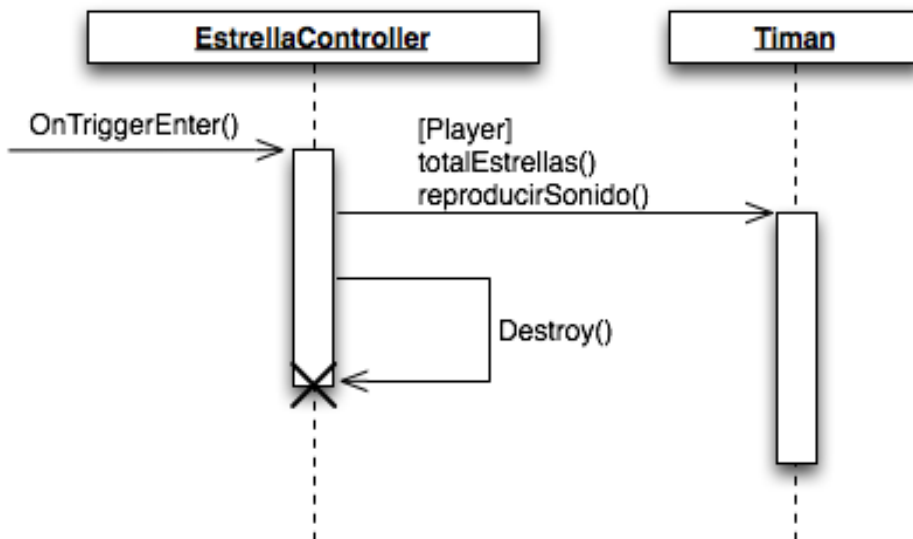


Figura F.18. Diagrama de secuencia que describe la interacción entre el personaje y una estrella

En el diagrama F.18 se muestra qué ocurre al pasar por encima de una estrella. Si el personaje entra en contacto con la estrella, se le envía el mensaje de aumentar el total de estrellas, a continuación el de reproducir el sonido que simula la recogida de la estrella y por último se destruye la estrella.

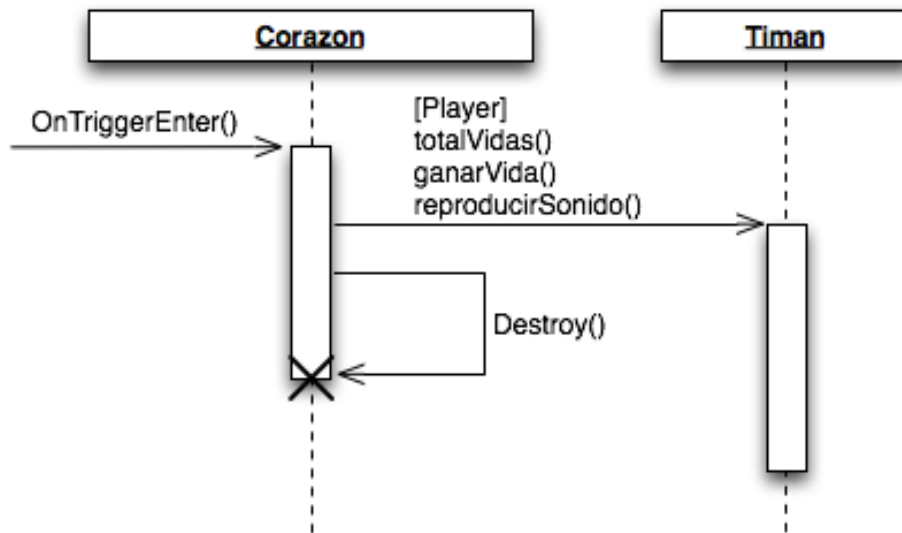


Figura F.19. Diagrama de secuencia que describe la interacción entre el personaje y corazón que recupera una vida del personaje

De forma similar, en la figura F.19, el personaje al entrar en contacto con un corazón provoca que este le envíe los mensajes de aumentar el número de vidas, mostrar una vida más en la interfaz del juego y reproducir el sonido de recogida de una vida. Después del envío de los mensajes, el corazón se destruye.

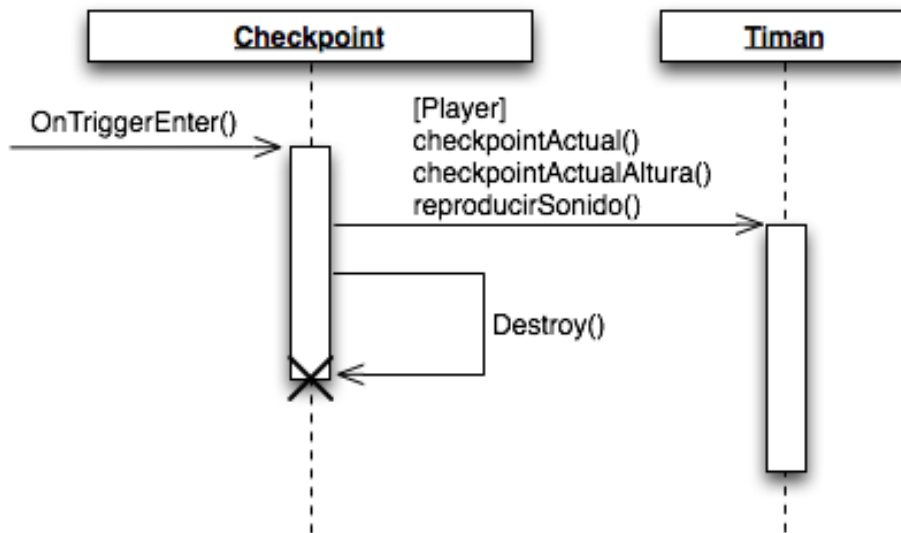


Figura F.20. Diagrama de secuencia que describe la interacción entre el personaje y un punto de guardado

En la figura F.20 se muestra la interacción entre el personaje y un punto de guardado (checkpoint). Al activar el checkpoint, este manda al personaje los mensajes de actualizar su posición de guardado a la del checkpoint, la altura del checkpoint (para que no atravesase el suelo) y la de reproducir el sonido de activación del checkpoint. Una vez enviados los mensajes, el checkpoint desaparece.

Por último y para terminar con este apartado se mostrará el diagrama de actividades que modela el comportamiento de la clase FinNivel:

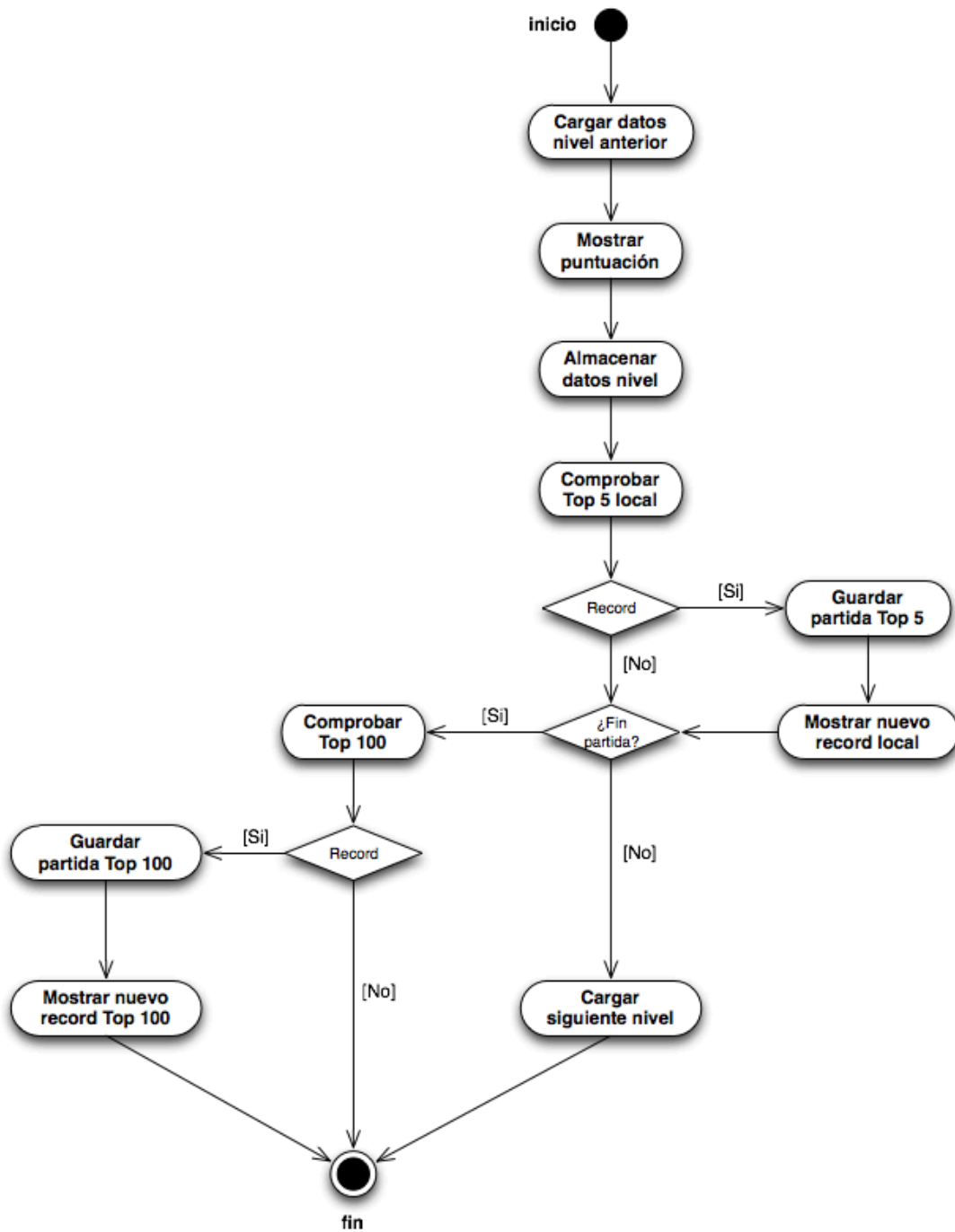


Figura F.21. Diagrama de actividades de FinNivel

Como se puede observar en la figura F.21 al finalizar un nivel se cargan los datos del nivel anterior que se sumarán a los del nivel actual generando los datos totales de la partida, si se está en el primer nivel, estos datos cargados serán 0. A continuación se muestran los datos del nivel actual junto a los totales de la partida y se guardan los datos de la partida actual.

Lo siguiente que se hace es comprobar si la puntuación total es un Top 5, si es un Top 5 se guarda la partida localmente y se muestra la nueva posición

al usuario, si no es Top 5, se mira si es fin de partida. Si no es fin de partida, se carga el siguiente nivel y se termina el actual. Si es fin de partida, se comprueba si es Top 100, si no es record Top 100 se sale de la partida sin guardar datos del Top 100, si es record Top 100 se guardarán los datos en la base de datos, se indicará la nueva posición dentro del Top 100 al jugador y se saldrá de la partida.

F.4.5. Interfaz del videojuego

Al igual que con la interfaz del visor, para crear la interfaz del juego se han analizado los requisitos y atendiendo a criterios de usabilidad se ha creado el diseño de la interfaz de usuario. Para el diseño de la interfaz se ha seguido el siguiente diagrama de navegación:

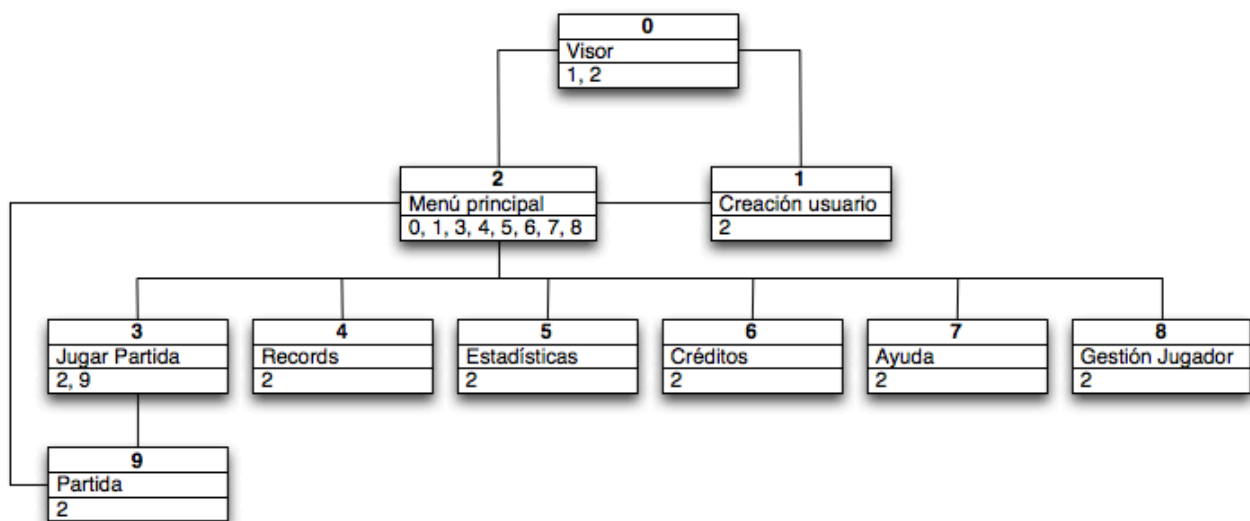


Figura F.22. Jerarquía de ventanas del videojuego

En la figura F.22 se observa la navegación que se puede seguir desde el videojuego. Se observa que para poder jugar al videojuego es necesario cargar el Visor, desde el Visor se lanza el videojuego y desde este se puede retornar al Visor. Una vez creado un usuario se accede al menú principal que es el menú que controla todos los aspectos del juego, borrar datos, jugar partida, ver información, cambiar datos y volver al curso.

En el anexo I se muestran todas las ventanas de la Figura 22 incluyendo la interfaz de una partida.

F.5. Pruebas

Como se ha comentado en la memoria las pruebas realizadas han sido las siguientes:

- ▶ Pruebas de caja blanca. Como pruebas de caja blanca se han realizado:
 - Prueba de cobertura de sentencias para partes de código fundamentales y sobre aquellas que planteaban dudas sobre su ejecución (como alguna condición por ejemplo).
 - Prueba de bucles. Al no haber prácticamente bucles anidados se han realizado pruebas de bucles simples. Entre estas pruebas se ha comprobado: pasar por alto el bucle, comprobar que el número de iteraciones era el correcto.

- ▶ Pruebas de caja negra:
 - Se ha hecho un análisis de valores límite.
 - Valores típicos de error.
 - Valores imposibles.

- ▶ Entre las pruebas de inspección de código se ha realizado:
 - Errores en la declaración de datos.
 - Errores en la computación, asignando a variables algo que no se debía asignar.
 - Errores de tipos inconsistentes.
 - Errores de control de flujo.
 - Errores en la interfaz.

- ▶ Las pruebas unitarias se han realizado sobre los módulos del visor y en los elementos de interacción del videojuego, como por ejemplo el movimiento del personaje y los saltos. También se han probado elementos que tienen relación directa con el personaje como superficies mortales y enemigos, sobre los que se ha probado que los efectos eran los que debían ocurrir. En

el visor por ejemplo se ha comprobado que se realizasen las cargas de temas correctamente, el avance y retroceso de páginas, zoom, apertura y cierre de las opciones, su desaparición al pasar páginas.

► Las pruebas de integración y de sistema han consistido en analizar las interacciones entre las principales partes del sistema que han sido desarrolladas por separado. Se han preparado casos de prueba en los que se conoce de antemano el resultado y a continuación se han probado. Se ha probado la relación del visor con el lanzamiento de vídeos, aplicaciones y videojuego; la vuelta de cada uno de los elementos a la parte del curso que debía. Para el videojuego se ha creado una réplica del personaje controlado por teclado que ha permitido asegurar que el funcionamiento del videojuego era el esperado. Esta réplica del personaje ha permitido también una mayor rapidez en el desarrollo del videojuego.

► Para las pruebas de aceptación se han elegido personas que no han tenido relación con el desarrollo del proyecto, se les ha explicado el funcionamiento de la aplicación y se les ha dejado usarla libremente, a continuación se ha evaluado su respuesta comprobando la ausencia de errores y sugerencias en cuanto a su funcionamiento.

F.6. Vídeos del curso



Figura F.23. Índice de vídeos del curso

En la figura 23 se muestra el listado de vídeos del curso, estos vídeos son ejemplos que se comentan durante el tema en el que se muestran y representan ejemplos de aplicaciones realizadas pero que por las características de las mismas y por apreciarse mejor en vídeo no se han realizado en formato aplicación.

A continuación se indica qué es lo que se muestra en cada vídeo:

- ▶ **Tema 7 - Creación de terreno:** en este vídeo se muestra un ejemplo de creación de terreno usando las herramientas de Unity que permiten crearlos.
- ▶ **Tema 8 - Movimiento personaje:** en el vídeo se hace uso del controlador de caracteres que se dispone en Unity. De forma sencilla se crea este controlador que permite desplazar la cámara en primera persona mediante el

uso de un teclado. También sirve para mostrar otro terreno creado y cómo se ve afectada la vegetación por el viento.

► **Tema 9 - Distancia entre objetos:** en el vídeo del tema 9 se observa como mediante el movimiento del personaje del tema anterior este se acerca y se aleja de un cubo que está fijo en la escena. En todo momento se muestra en la pantalla la distancia del personaje a la caja.

► **Tema 10 - Colisiones selectivas:** se muestra mediante el lanzamiento de proyectiles cómo se destruyen ciertos objetos y otros no. También se muestra cómo al producirse una colisión se puede emitir sonidos.

► **Tema 11 - Apertura de puertas:** es un ejemplo en el que se puede observar cómo hacer que un objeto que simula una puerta se abra. Esta apertura de puertas no se realiza con cualquier objeto sino que solo se abren al acercarse el personaje.

► **Tema 11 - Bandera pirata:** en este vídeo se pretende mostrar cómo se crean objetos que simulan la tela. Para ello se ha creado una bandera pirata que está anclada a un mástil. Esta bandera se mueve simulando efecto del viento.

► **Tema 11 - Efectos de sonido:** vídeo en el que se muestra los distintos tipos de efectos de sonido, direccional y esférico y los modos de atenuación del sonido en función de la distancia desde la fuente del sonido al punto de escucha.

► **Tema 11 - Audio scripting:** en el vídeo se ha creado una zona en la que al entrar se reproduce un sonido. Es un ejemplo de programación donde se observa el efecto al acercarse a una caja.

► **Tema 11 - Ejemplo resumen de juego 3D:** se muestra en el ejemplo un vídeo de lo que podría ser un juego en primera persona. En el ejemplo se ha creado un personaje que se controla por teclado, lleva un arma que dispara proyectiles, hay objetos que se pueden destruir y la puerta que se ha creado se abre al acercarse el jugador.

► **Tema 13 - Ejemplo de construcciones:** el tema 13 es un capítulo en el que se termina de explicar la construcción del videojuego completo del curso. En este vídeo se muestran todas las construcciones creadas del nivel 1.

- ▶ **Tema 13 - Plataformas fijas y giratorias:** se muestran las primeras plataformas fijas y giratorias del nivel uno.
- ▶ **Tema 13 - Plataformas con desplazamiento:** se muestra la primera plataforma con desplazamiento del nivel uno.
- ▶ **Tema 13 - Plataformas con desplazamiento vertical:** se muestra el efecto de una plataforma que se desplaza verticalmente.
- ▶ **Tema 13 - Enemigos nivel 1:** en este vídeo se enseña la colocación de los enemigos del nivel uno.
- ▶ **Tema 13 - Cajas de madera y de cartón:** como el propio nombre indica se muestra la colocación de este tipo de plataformas.
- ▶ **Tema 13 - Estrellas, vidas, cofres y checkpoints:** en el vídeo se muestra la colocación de estos cuatro elementos en el nivel uno.
- ▶ **Tema 13 - Lava fija y desplazamiento:** en el vídeo se observa la colocación de estos dos elementos importantes, ya que son el límite inferior del nivel uno.
- ▶ **Tema 13 - Ejemplo nivel 1:** es un vídeo en el que se hace un recorrido por todo el nivel uno.
- ▶ **Tema 13 - Ejemplo nivel 2:** es un vídeo en el que se hace un recorrido por todo el nivel dos.
- ▶ **Tema 13 - Ejemplo nivel 3:** es un vídeo en el que se hace un recorrido por todo el nivel tres.
- ▶ **Tema 13 - Ejemplo completo de juego:** vídeo promocional del videojuego en el que se muestran las habilidades del personaje, se hace un recorrido por los tres niveles y se muestra un ejemplo de partida.

F.7. Aplicaciones interactivas

En este apartado se van a resumir las aplicaciones interactivas que pueden ser ejecutadas desde el visor.



Figura F.24. Índice de aplicaciones del curso

- ▶ **Hola mundo 3D:** aplicación típica de programación pero en este caso en 3D. Aparte de mostrar un mundo en 3D se puede interactuar con él desde un iPad, permite rotar el mundo al mover el iPad de izquierda a derecha y viceversa y acercar o alejar la cámara al inclinar el dispositivo.
- ▶ **Ejemplo de terreno en 3D con ejemplo de viento:** aplicación que muestra varias cosas. En primer lugar la animación de la cámara, que sigue un recorrido predefinido. Otro aspecto es el terreno creado con el editor de terrenos de Unity. Por último se muestra el efecto del viento en la hierba y hojas de los árboles.
- ▶ **Ejemplo de caída de objeto, cubo sobre cubo:** aplicación en la que se muestra el efecto de la física sobre un cuerpo. Son dos cubos, uno sobre una superficie y otro colocado a cierta altura que cae sobre el anterior. La aplicación se puede reiniciar desde ella misma todas las veces que se desee.

- ▶ **Ejemplo de Joints sin fuerzas sobre los objetos:** aplicación en la que se observa varios objetos unidos y cómo se ven afectados por la gravedad. Como objetos se tiene una bola unida con eslabones y un plano, ambos fijados en el espacio.
- ▶ **Ejemplo de Joints con fuerzas:** es el mismo ejemplo que el anterior pero en este se pueden aplicar fuerzas laterales positivas y negativas sobre la bola con cadenas y sobre el plano. También se pueden reiniciar los valores al pulsar en un botón.
- ▶ **La venganza de Timan:** videojuego completo de ejemplo del curso.

Anexo G

Manual de usuario

G.1. Perspectiva general de la aplicación

La aplicación ha sido diseñada para impartir o recibir un curso de programación de videojuegos en Unity 3D en un iPad. El usuario de la aplicación puede ver el curso, reproducir vídeos, lanzar aplicaciones interactivas o jugar una partida a un videojuego completo.

Para el uso de la aplicación es necesario disponer de un iPad.

Funcionalidades de la aplicación:

- ▶ Visor de textos del curso, se dispone de acceso a índices de temas, vídeos, aplicaciones interactivas y videojuego completo.
- ▶ Reproductor de vídeos del curso.
- ▶ Ejecución de aplicaciones interactivas de ejemplo.
- ▶ Videojuego “La venganza de Timan”.

Restricciones:

- ▶ Restricciones Hardware: dispositivo iPad y conexión a internet para poder ver los vídeos del curso y poder registrar en internet las puntuaciones máximas del jugador.
- ▶ Restricciones Software: versión iOS 5.0 o superior.

G.2. Visor

Avanzar y retroceder páginas

El visor se controla de forma táctil, en las imágenes G.1 y G.2 se muestra la navegación permitida desde la primera página del curso.

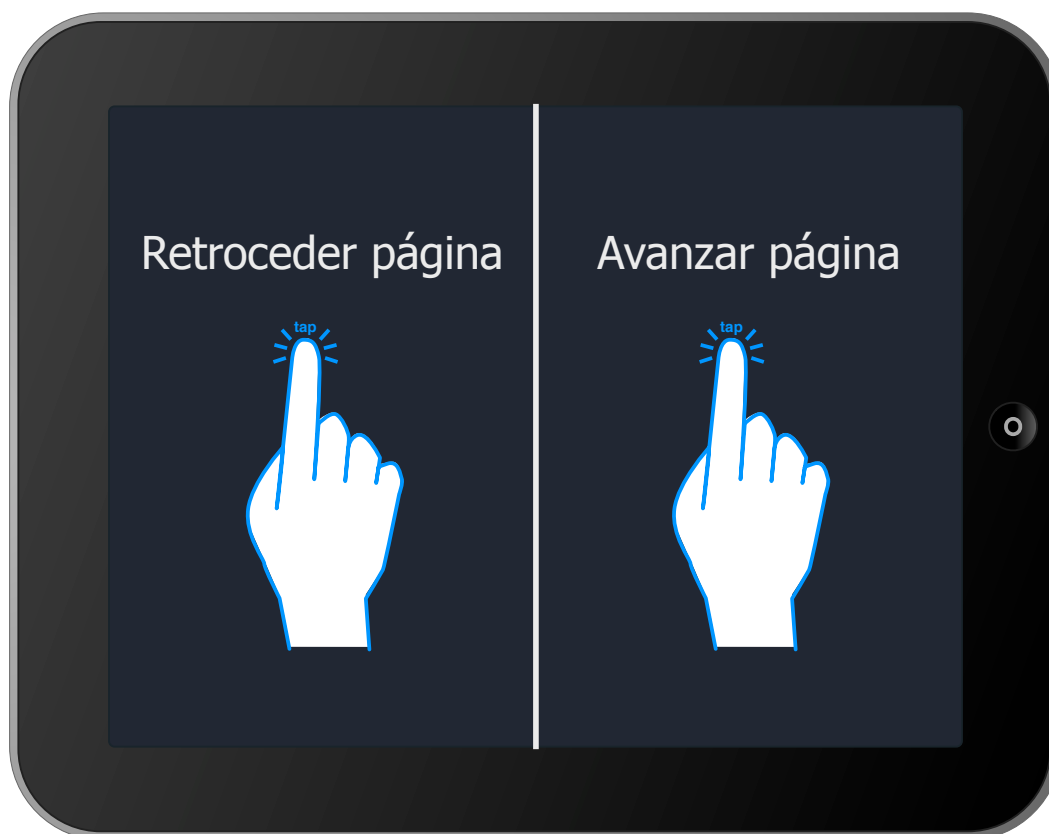


Figura G.1. Avance y retroceso de página mediante pulsaciones en la pantalla

En la figura G.1 se muestra que si consideramos la pantalla dividida en dos partes, para avanzar una página se debe pulsar en el lado derecho de la pantalla y para retroceder una página se debe pulsar en el lado izquierdo.

Otra forma de poder avanzar páginas como se ve en la figura G.2 es deslizando los dedos, por ejemplo si se desliza el dedo de derecha a izquierda en la parte derecha, se avanza página y si se desliza el dedo de izquierda a derecha en la parte izquierda, se retrocede página.



Figura G.2. Avance y retroceso de página por deslizamiento

Zoom

Para hacer zoom en el documento se debe pulsar con dos dedos sobre la pantalla, como se muestra en la figura G.3. Tras hacer zoom se podrá mover la pantalla de arriba a abajo y viceversa y de derecha a izquierda y viceversa como se indica en la figura G.4.

En modo zoom no se permite avanzar o retroceder páginas, será necesario salir del modo zoom para poder avanzar y/o retroceder páginas.

Para desactivar el zoom se deberá volver a pulsar con dos dedos en la pantalla.



Figura G.3. Activación y desactivación del zoom

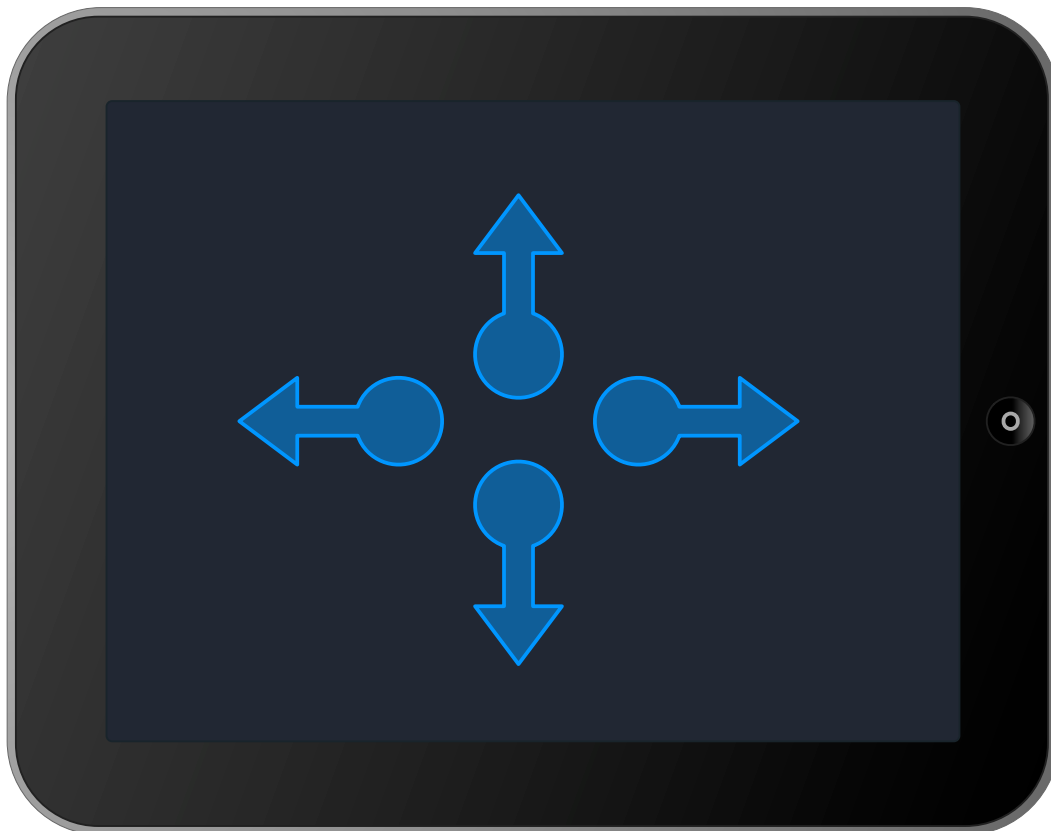


Figura G.4. Desplazamiento de pantalla con zoom activado

Acceso a las opciones

Para acceder a las opciones del visor se tendrá que pulsar sobre la cinta de color rojo que aparece en la figura G.5.

De forma normal el menú con las opciones aparece oculto y será necesario activarlo para que se muestren las opciones.



Figura G.5. Ejemplo de pantalla con las opciones sin desplegar

En la siguiente figura (G.6) se muestra el menú desplegado con las opciones.

Si se avanza o se retrocede de página el menú desaparecerá.



Figura G.6. Ejemplo de pantalla con las opciones desplegadas

Las opciones que se ofrecen desde el menú son:

- ▶ Ejemplos: acceso al índice de aplicaciones interactivas del curso (figura G.7).
- ▶ Vídeos: acceso al índice de vídeos del curso (figura G.8).
- ▶ Índice: acceso al índice de temas escritos del curso (figura G.9).
- ▶ Ayuda: acceso a la ayuda resumida del juego (figura G.10).

En la figura G.7 del curso se ve que se puede acceder a las aplicaciones interactivas del curso entre ellas al juego completo ('La venganza de Timan').

El botón 'Índice' vuelve al índice de temas escritos del curso.

Una vez lanzada una aplicación se muestra en la misma un botón que permite volver al curso.

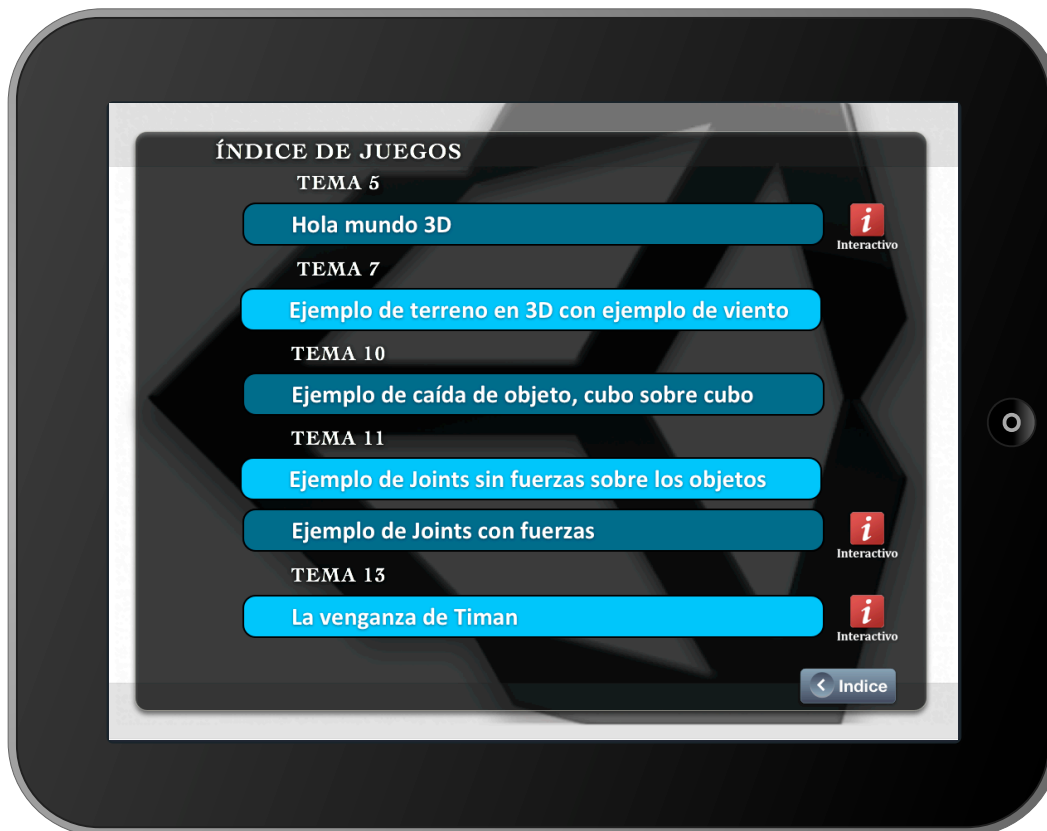


Figura G.7. Índice de aplicaciones interactivas

En la figura G.8 se muestra el índice de vídeos, al pulsar sobre uno de ellos comienza la reproducción del vídeo dentro de la aplicación.

El botón 'Índice' vuelve al índice de temas escritos del curso.

Una vez que se reproduce un vídeo si se pulsa sobre el botón 'Done' del reproductor aparecerá un botón que permite volver a la hoja del texto donde se hace referencia al curso.

En la figura G.9 se muestra el índice de temas escritos, desde el mismo índice se puede acceder al índice de aplicaciones, al índice de vídeos y a la ayuda.



Figura G.8. Índice de vídeos del curso



Figura G.9. Índice de temas escritos del curso

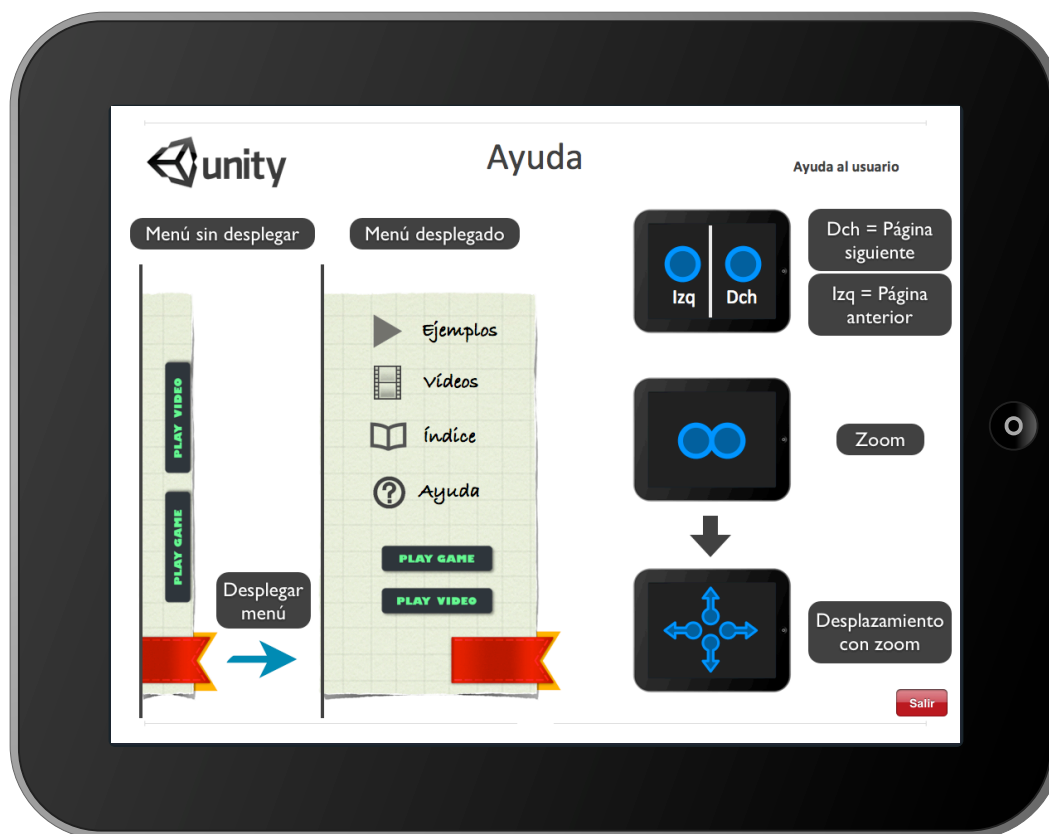


Figura G.10. Ayuda del curso cargada desde el menú de opciones

En la figura G10 se muestra la ayuda del curso que se accede tras pulsar en el menú de opciones sobre el texto 'Ayuda'.

Una vez cargada esta ayuda se puede volver a la página del curso en la que se encontraba pulsando sobre el botón 'Salir' de la esquina inferior derecha.

En algunas hojas en la parte izquierda del menú puede aparecer un botón de 'Play Video' o 'Play Game', esto indica que en esa hoja hay un vídeo y/o aplicación interactiva que está relacionada con el texto. Pulsando sobre estos botones se puede realizar la carga del vídeo o de la aplicación sin necesidad de ir al índice de vídeos o aplicaciones. En la figura G.11 se muestra un ejemplo.



Figura G.11. Botones 'Play Video' y 'Play Game' que permiten reproducir desde la misma hoja el vídeo o la aplicación interactiva relacionada

En la figura G.12 se muestra el reproductor y el botón 'Done' que muestra un botón que permite volver a la página del curso desde donde se puede lanzar el vídeo (similar a la de la figura G.11 con 'Play Video').

En la figura G.13 se muestra un ejemplo de aplicación interactiva y el botón de vuelta al curso en el centro de la pantalla (similar a la de la figura G.11 con 'Play Game').



Figura G.12. Ejemplo del reproductor de vídeos

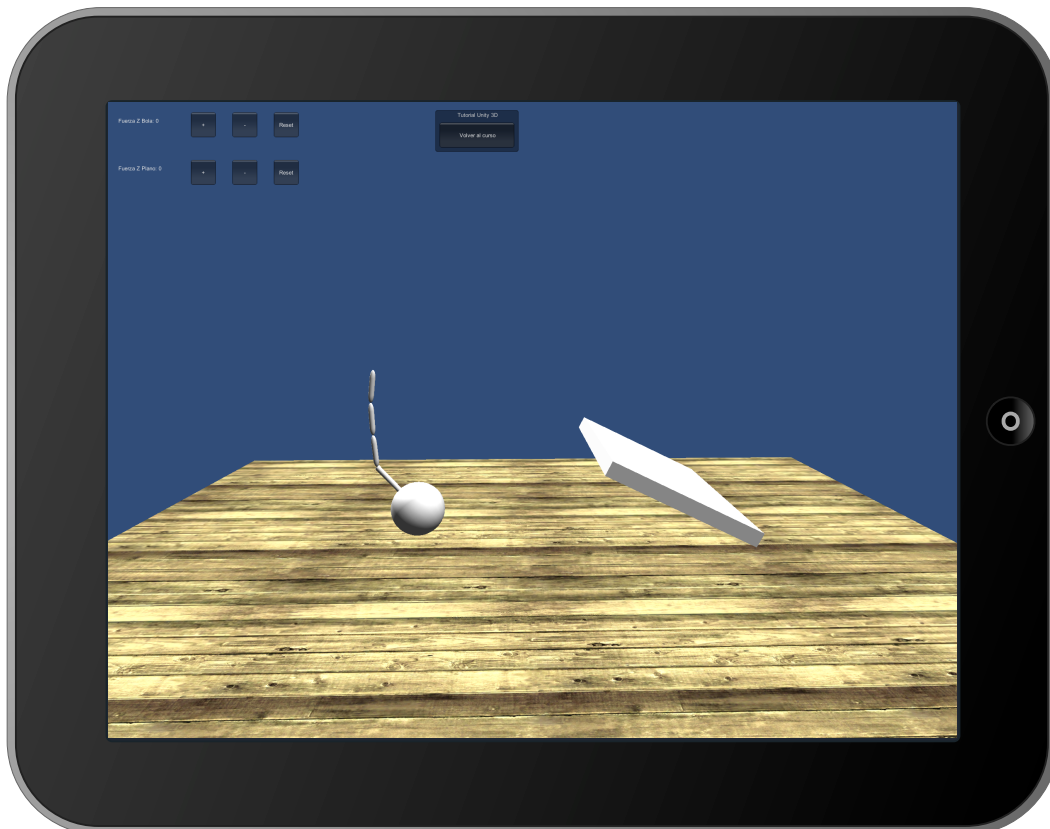


Figura G.13. Ejemplo de aplicación interactiva con el botón de vuelta al curso

G.3. Videojuego “La venganza de Timan”

Al jugar por primera vez al juego es necesario crear un nuevo usuario que será registrado en la base de datos de Top 100 máximas puntuaciones.

En la figura G.14 se muestra el formulario que es necesario rellenar para crear un usuario.



Figura G.14. Formulario de creación de nuevo usuario

Una vez creado el usuario se da acceso al menú que se muestra en la figura G.15.

La opción de borrar todos los datos hará que se elimine toda la información de la aplicación y se volverá al menú de la figura G.14.

La opción de volver al curso, volverá a la página del texto que explica el proceso de creación del videojuego.



Figura G.15. Menú principal del juego

El resto de opciones del menú se indican a continuación:

- ▶ **Jugar Partida:** tras seleccionar esta opción, se accederá al menú de selección de dificultad, existen dos opciones de dificultad, nivel normal y difícil. Después de elegir la dificultad, dará comienzo la partida. Se puede observar el menú de selección de dificultad en el anexo I (página 197). Tras comenzar una partida, se puede volver al menú principal directamente desde el juego.
- ▶ **Records:** esta opción dará acceso al menú de Top 5 mejores puntuaciones del usuario (anexo I página 198). Desde el menú de records se podrá acceder al Top 100 puntuaciones (vía web), borrar los datos de partidas guardadas y volver al menú principal.

- ▶ Estadísticas: muestra las estadísticas más importantes de las partidas jugadas (anexo I página 198). Las estadísticas se muestran según el nivel de dificultad. Desde este menú se pueden borrar todos los datos estadísticos y acceder al menú principal.
- ▶ Créditos: se muestran los créditos del juego y se permite volver al menú principal (anexo I página 199).
- ▶ Ayuda: si se selecciona esta opción se accederá a la ayuda del juego (anexo I página 199) donde se indican las acciones que se pueden llevar a cabo mediante pulsaciones en la pantalla. Como en el resto de opciones se permite volver al menú principal.
- ▶ Gestión Jugador: esta opción permite modificar los datos personales del usuario (anexo I página 200). Si se cambia el nombre, se verá el nuevo nombre elegido (si no existía previamente el usuario) en el menú principal del juego. Se pueden cambiar todos o algunos de los datos, si se cambia el nombre, se actualizarán las partidas del jugador en el servidor web pudiendo comprobar los cambios a través de la página web de puntuaciones. Desde este menú se puede volver al menú principal.

G.3.1. Objetivos del juego

El objetivo del juego es alcanzar el último nivel y permitir que Timan alcance 'El Cielo'.

Aparte de alcanzar 'El Cielo' y terminar la partida, existe la motivación para el jugador de conseguir la máxima puntuación en cada uno de los niveles.

Para facilitar el progreso en los niveles y por tanto en el juego, existen distribuidos puntos de guardado o checkpoints. Una vez recogido uno, si se muere se volverá a la posición del último checkpoint recogido.



Figura G.16. Checkpoint del juego

Se perderán vidas si se entra en contacto con superficies mortales o si hay contacto lateral con un enemigo. Se recuperarán vidas si se recogen corazones, una por cada corazón recogido. Cada nivel de una partida se comienza con 3 vidas.

G.3.2. Control del personaje

El jugador como se pone en el papel del héroe "Timan", a través de una vista en tercera persona moverá el personaje por los distintos niveles.

El jugador podrá mover el personaje hacia la derecha y hacia la izquierda además de poder saltar. El personaje estará sometido a las leyes de la física (esencialmente la fuerza de la gravedad). Los movimientos se indican en las figuras G.17 y G.18.

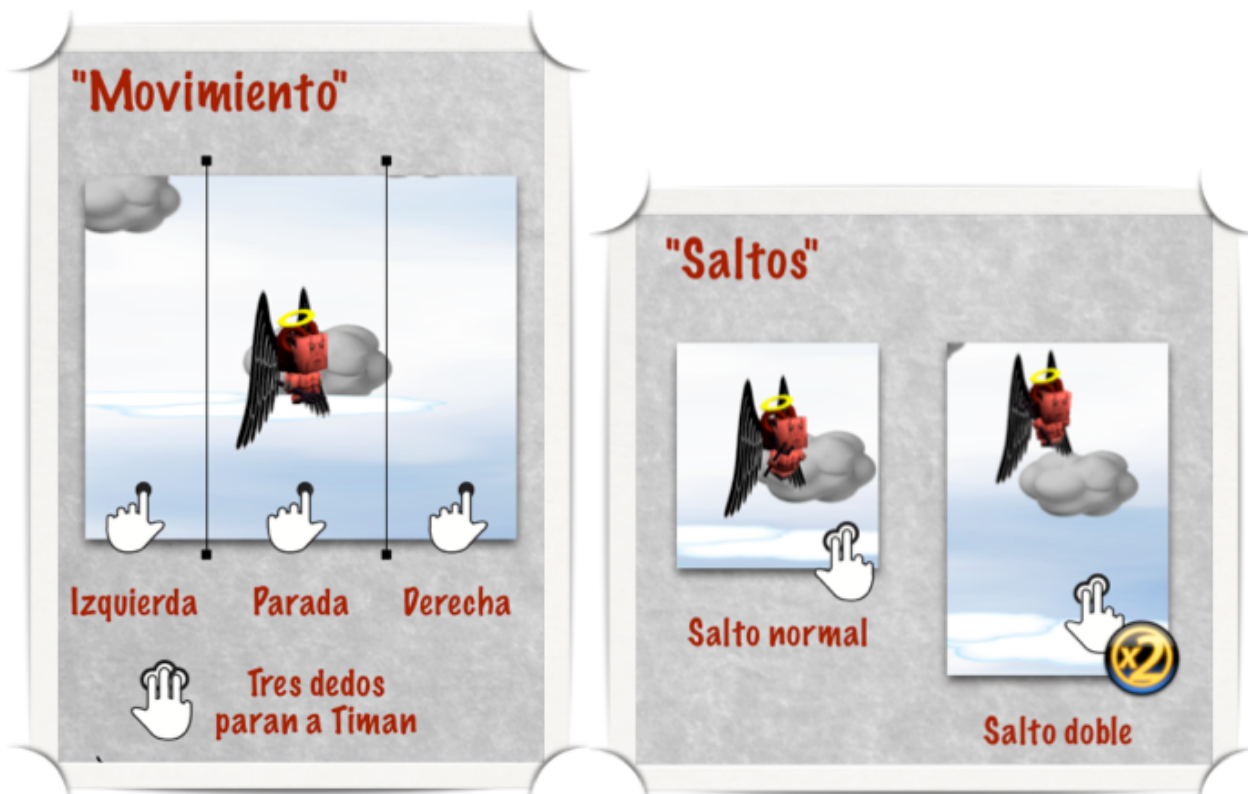


Figura G.17. Resumen de los movimientos del personaje principal
Figura G.18. Resumen de los saltos del personaje principal

En la figura G.17 se muestran los movimientos del personaje, si se pulsa a la derecha del personaje, se moverá hacia la derecha, si se pulsa a la izquierda del personaje, se moverá hacia la izquierda, si se pulsa sobre el personaje o se pulsa la pantalla con tres dedos, el personaje se parará.

En la figura G.18 se muestran los saltos que puede realizar el personaje, con dos dedos hará un salto y si se ha saltado una vez se podrá realizar un segundo salto con otra doble pulsación en la pantalla.

G.3.3. Puntuaciones

Elemento	Condición	Puntuación
Vida	Por cada vida	50 puntos
Tiempo (t) en minutos	$t < 5$	1000 puntos
	$5 < t < 7$	750 puntos
	$7 < t < 10$	300 puntos
	$10 < t < 15$	100 puntos
Estrella	Por cada estrella	10 puntos
Cofre	Por cada cofre	100 puntos
Enemigo	Por cada enemigo	10 puntos

Tabla G.1. Tabla de puntuaciones del juego

En la tabla G.1 se muestra la tabla resumen de las puntuaciones del juego y qué elementos las proporcionan.



Figura G.19. Elementos del juego

En la figura G.19 se muestran los elementos que proporcionan puntuaciones.

G.3.4. Interfaz de la partida



Figura G.20. Interfaz del juego

En la figura G.20 se muestra la interfaz del juego en ella se muestra en la parte izquierda al lado del símbolo de la estrella el número de estrellas conseguidas respecto del total disponible en el nivel. Justo debajo aparece un símbolo de un cofre que indica el número de cofres conseguidos respecto del total del nivel.

En la parte central superior aparece el tiempo de partida que ha discurrido desde el comienzo de la misma. En la parte central inferior aparece el botón que da acceso al menú principal.

En la parte derecha se muestra mediante corazones el número de vidas, en la imagen se observan 3 vidas que es el número inicial de vidas de cada nivel. Debajo de las vidas se muestra el número de enemigos muertos respecto del total del nivel.

Anexo H

Documento de Diseño del Juego (GDD)

Un GDD del inglés Game Design Document es una documentación que es recomendable crear para ser usada como guía para la creación de un videojuego.

Un GDD normalmente es creado y editado por el equipo de desarrollo y es una forma de organizar y documentar el desarrollo del videojuego. Se usa en la industria de los videojuegos para organizar los esfuerzos de un equipo de desarrollo.

El documento es creado por el equipo de desarrollo en colaboración con diseñadores, artistas y programadores para servir como guía visual durante el proceso de desarrollo del videojuego.

A grandes rasgos un GDD es una forma de expresar la visión que se tiene del juego, describe los contenidos y presenta un plan para su implementación.

Un GDD es un documento a través del cual el director del proyecto tiene unos objetivos fijados con los que poder controlar el desarrollo del videojuego. Es un documento que el diseñador usa para plasmar sus ideas, y es un documento con el que poder dar instrucciones a los artistas y programadores del videojuego.

El GDD depende de las personas que forman parte del proyecto y no existe una guía única para su elaboración. A continuación se indican una serie de puntos comunes a los GDD recomendables para todo desarrollo de un videojuego.

El anexo siguiente junto con este puede servir de guía para la elaboración de la documentación de un videojuego.

H.1. Modelo de GDD

Portada del documento:

- ▶ Lo primero es indicar el nombre del documento: 'Documento de Diseño del Juego' (GDD o su variante en castellano DDJ).
- ▶ Lo segundo es indicar el título del videojuego y entre paréntesis si tiene, el título mientras se trabaja con el.
- ▶ A continuación la versión.
- ▶ Fecha de creación.
- ▶ Última actualización, indicando la fecha.

Tabla de contenidos:

1. Visión general del proyecto

- 1.1. Resumen del videojuego
- 1.2. Conceptos de diseño del juego
- 1.3. Núcleo del juego
- 1.4. Género
- 1.5. Audiencia destino
- 1.6. Competidores
- 1.7. Miembros del equipo / Trabajos / Información de contacto

2. Diseño del juego

- 2.1. Visión de conjunto del juego
- 2.2. Historia
- 2.3. Descripción del mundo
- 2.4. Personajes
- 2.5. Niveles de dificultad
- 2.6. Entorno
- 2.7. Vista general de los niveles
- 2.8. Storyboards

3. Juego

- 3.1. Objetivos y metas del jugador
- 3.2. Reglas del juego
- 3.3. Habilidades
- 3.4. Recursos del jugador
- 3.5. Límites y restricciones
- 3.6. Combate
- 3.7. Puntuaciones
- 3.8. Enemigos

4. Diseño de los menús del juego

- 4.1. Esquema de colores
- 4.2. Pantalla de entrada al juego
- 4.3. Información legal
- 4.4. Menú principal del juego
- 4.5. Tutorial / instrucciones
- 4.6. Pantalla de créditos
- 4.7. Pantalla de victoria
- 4.8. Pantalla fin de nivel
- 4.9. Pantalla de fin de juego
- 4.10. Pantalla de puntuaciones máximas
- 4.11. Pantalla de selección de nivel

5. Diseño de la interfaz del juego

- 5.1. Posición de la cámara
- 5.2. Controles del juego
- 5.3. Modos de juego
- 5.4. Puntuación de la partida
- 5.5. Tiempo de juego
- 5.6. Menú de opciones
- 5.7. Vuelta al menú principal

Acompañando al GDD sería interesante realizar un Documento de Diseño Técnico (TDD - Technical Design Document). Una guía de lo que debería incluir el documento se muestra a continuación.

H.2. Modelo de TDD

Portada del documento:

- ▶ Lo primero es indicar el nombre del documento: 'Documento de Diseño Técnico' (TDD o su variante en castellano DDT).
- ▶ Lo segundo es indicar el título del videojuego y entre paréntesis si tiene, el título mientras se trabaja con el.
- ▶ A continuación la versión.
- ▶ Fecha de creación.
- ▶ Última actualización, indicando la fecha.

Tabla de contenidos:

1. Visión general del proyecto

- 1.1. Resumen del proyecto
- 1.2. Visión general de la arquitectura del juego

2. Hardware y Software necesario

- 2.1. Software 2D necesario
- 2.2. Software 3D necesario
- 2.3. Software de sonido
- 2.4. Programación
- 2.5. Hardware requerido
 - 2.5.1. Mínimo
 - 2.5.2. Recomendado para desarrollo de contenido

3. Evaluación

- 3.1. Motor
- 3.2. Plataformas destino

4. Plan de desarrollo

- 4.1. Temporización
- 4.2. Hitos
- 4.3. Objetivos del proyecto
- 4.4. Objetivos de productos terminados

5. Datos compartidos

- 5.1. Información del servidor

6. Formatos de los ficheros

- 6.1. Formatos 2D
- 6.2. Formatos 3D
- 6.3. Entorno de desarrollo
- 6.4. Ficheros de sonido

7. Diseño de los niveles

- 7.1. Lista de niveles y mapas del mundo

8. Lista de recursos

8.1. Tipos comunes

- 8.1.1. Arte 2D
- 8.1.2. Modelos 3D
- 8.1.3. Sonido

8.2. Nivel 1 (nombre del nivel)

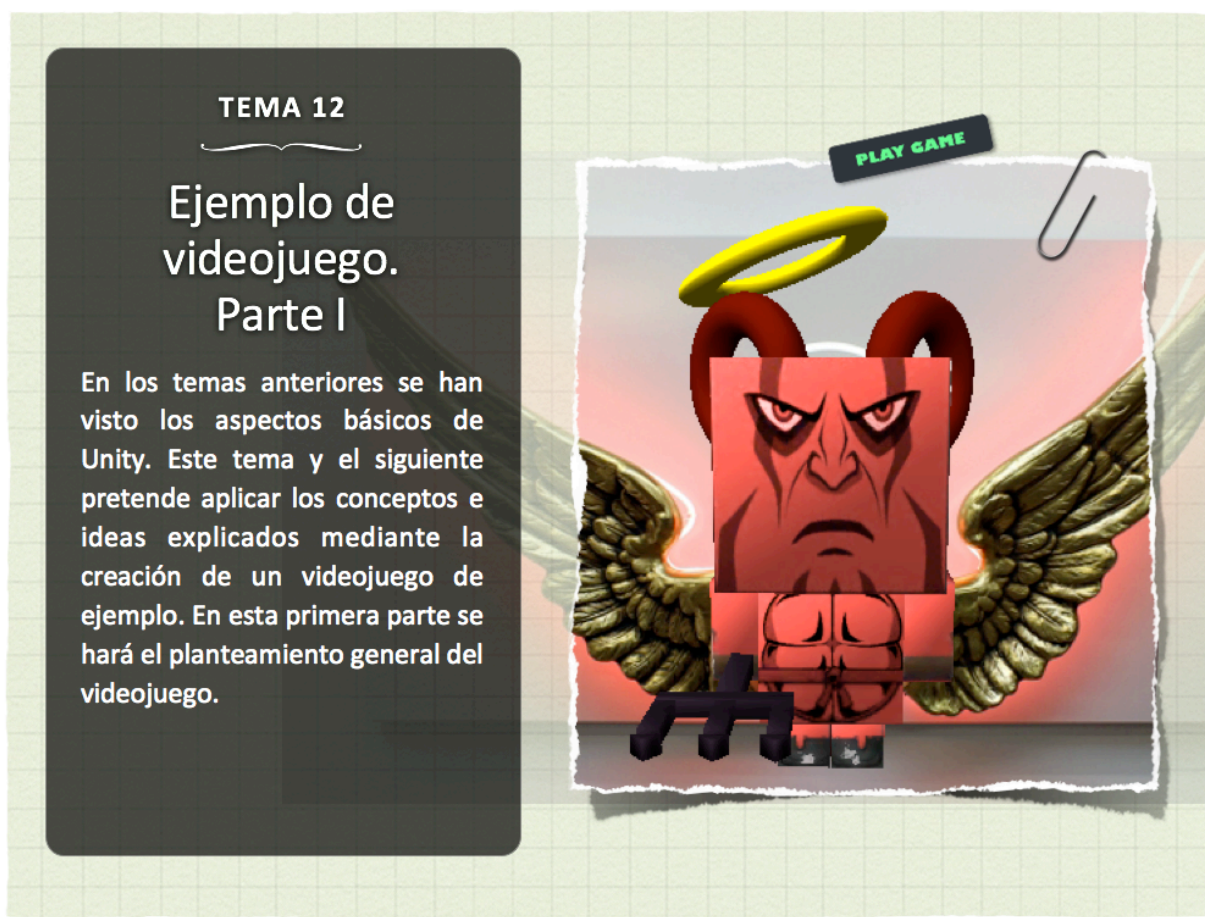
- 8.2.1. Arte 2D
- 8.2.2. Modelos 3D
- 8.2.3. Ambientación del nivel
- 8.2.4. Sonido

8.3. Nivel 2 (nombre del nivel). Copiar y pegar punto 8.2 para cada nivel

Anexo I

Ejemplo de capítulo del curso

A continuación se muestra un capítulo de ejemplo del curso. El capítulo mostrado es el del tema 12 que muestra la primera parte del proceso de creación del videojuego completo. Este capítulo también sirve como ejemplo a grandes rasgos de GDD (Documento de Diseño del Juego).



Contenidos

- Elementos básicos de un videojuego
- La idea
- La estructura del juego
- La interfaz
- Elementos del juego
- Niveles
- Música y sonido
- Creación de los elementos del juego
- Organización de las escenas
- Programación
- Organización de los elementos del juego



TEMA 12

Ejemplo de videojuego



Introducción

Tema 12
Ejemplo de videojuego

- ¿Qué es un videojuego?. Un videojuego es algo a lo que se juega. Desde un punto de vista más científico una definición podría ser:

“Un juego es un ejercicio de control voluntario, en el que existen objetivos, con unas determinadas reglas en el que se puede ganar y perder”.

- A la hora de crear un videojuego por primera vez surgen un gran número de preguntas:

- ¿Qué tipo de juego crear?.
- ¿A qué audiencia irá destinado el juego?.
- ¿Qué experiencia debe tener el jugador?.
- ¿Debe haber algún tipo de sorpresa en el juego?.
- ¿Deben existir notas de humor en el juego?.
- ¿Qué retos debe plantear el juego?.
- ¿Qué tipo de preguntas debe hacerse el jugador?.
- ...

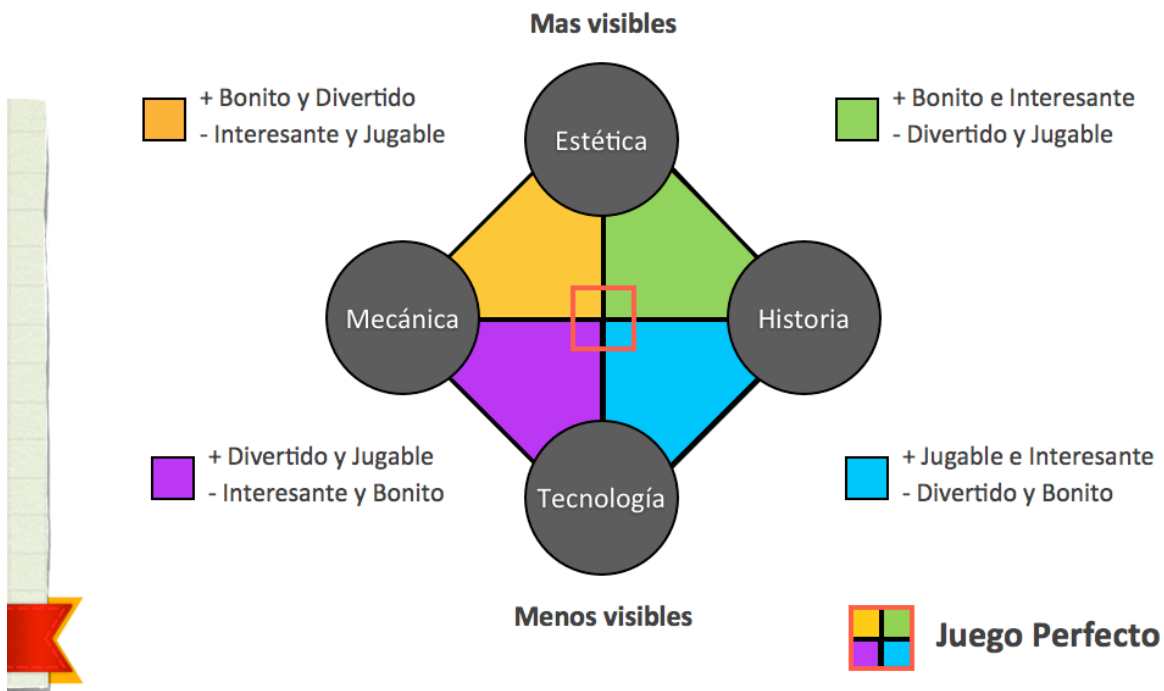
- Pese a plantearse un gran número de preguntas y posibilidades de realización en los primeros momentos de la creación de un videojuego, existen cuatro elementos básicos de todo videojuego:

- 1.- Mecánica: es el conjunto de procedimientos y reglas del juego. La mecánica describe el objetivo principal del juego, cómo pueden y cómo no lograrlo los jugadores, y qué ocurre cuando lo consiguen.
- 2.- Historia: es la secuencia de eventos que se desarrollan en el juego. Pueden ser lineales y predefinidos, o pueden variar y emerger. La historia debe ser contada a través del juego, se debe elegir aquellas mecánicas que harán que la historia cobre fuerza.
- 3.- Estética: es cómo se muestra el juego, suena, “huele”, se “saborea” y se siente. Es un punto muy importante dentro del diseño de un juego ya que es responsable directo de transmitir experiencias al jugador.

4.- Tecnología: hace referencia a todos los elementos e interacciones que hace posible el juego. La tecnología elegida permitirá hacer determinadas cosas e imposibilitará hacer otras.

- Todos los elementos están relacionados entre sí de forma que para que la historia sea interesante y cautive al jugador, el juego estéticamente debe ser atractivo. Pero no basta con esto, se puede tener una buena historia y que estéticamente sea bonito un juego, pero si la mecánica falla, el juego no será divertido. De igual forma si a estos tres elementos no les acompaña la tecnología, el resultado final puede distar mucho del esperado.

- La relación entre estos elementos se puede resumir en la siguiente figura:



- Todo videojuego nace de una idea. A partir de esta idea es por donde comienza a construirse toda la estructura del juego, la historia le dará sentido y la estética será la responsable de que se transmitan y se refuercen las emociones deseadas. La mecánica del juego hará que la idea se plasme en el videojuego y la tecnología será fundamental para transmitir la experiencia deseada.

- Para generar ideas es útil hacer una lluvia de ideas a partir de los cuatro elementos comentados anteriormente:

- Ideas basadas en la tecnología: plataforma móvil, juego para móviles, PC, integración con aplicaciones de mensajes instantáneos, consola, ...
- Ideas basadas en la mecánica: simuladores, juegos de ficción interactivos, juegos tipo tetris, plataformas, RPG's (Role Playing-Game), MMORPG (Massively Multiplayer Online Role-Playing Game), ...
- Ideas basadas en la historia: vampiros, invasiones alienígenas, de amor, temas musicales, ...
- Ideas basadas en la estética: estilo anime, todos los personajes son animales, todos los personajes son cubos, el estilo de música que defina los sentimientos o incluso el juego ...

La historia

- Lo primero que se hizo fue pensar en el tipo de juego que se quería realizar como ejemplo y se decidió crear un juego de tipo "plataformas".
- A partir del tipo de juego empezaron a surgir ideas sobre el personaje principal y cómo debería ambientarse la historia del juego.
- Así surgió el personaje y a partir de ahí se desarrolló la historia:

"El personaje es un demonio cuyo nombre es Timan y su hogar como no podía ser otro es 'El Infierno'. Tras miles de años, Timan sufre una transformación, afloran recuerdos en su mente y se da cuenta de que su sitio no es 'El Infierno'. Acusado de traición se ve obligado a luchar por alcanzar su objetivo que no es otro que ocupar su lugar en 'El Cielo'."

611

La mecánica

- El objetivo principal del juego es alcanzar el último nivel con la mayor puntuación. Matar enemigos y recoger recompensas otorgará un número determinado de puntos, así como finalizar cada nivel en un tiempo determinado o pasar de nivel con el máximo de vidas (o alguna de ellas).
- Existen dos niveles de juego, un nivel normal en el que el número de vidas actúa como bonificador al final de cada nivel y un nivel difícil en el que las vidas fijan la posibilidad o no de seguir jugando. En ambos niveles de dificultad se parte con tres vidas, éstas se pueden perder al chocar con un elemento o al entrar en contacto con ciertos ambientes "mortales" como lava, rayos ... En el nivel difícil, al perder la tercera vida finalizará la partida tomando como puntuación de la misma los puntos adquiridos hasta ese momento.
- Se pueden perder vidas al entrar en contacto con superficies mortales (como lava, trampas, ...) o al tocar a un enemigo por una zona que no sea la superior (caer encima de un enemigo provocará la muerte de este).

612



Si al saltar se cae encima de un enemigo, este morirá, consiguiendo 10 puntos.

Muerte del enemigo

613



Por el contrario, si se alcanza a un enemigo lateralmente, se perderá una vida.

Muerte de Timan

614

La estética

- Una vez definido el personaje y la historia, un paso natural fue pensar en la estética que iba a tener el juego. Antes de describir la estética de cada nivel, se pensó en crear una estética tipo comic, es decir desenfadada y “graciosa”.
- Al ser un juego de ejemplo en el que se iban a plasmar los conceptos de Unity vistos en temas anteriores se pensó en crear tres niveles, cada uno con su propia “personalidad” con la que se transmitiesen situaciones diferentes, así cada nivel tiene su propia estética.
- En el primer nivel, Timan siendo todavía un demonio, lo natural era en pensar en un ambiente que simulase “El Infierno” del cual intenta escapar.
- El segundo nivel, nivel de transición entre “El Cielo” y “El Infierno”, se pensó en una especie de “Purgatorio”, por lo que se optó por introducir elementos típicos de un cementerio y de halloween.

615

- Por último el tercer nivel sería el de la llegada al cielo por lo que se pensó en simular el ascenso al mismo junto con un recibimiento de un coro de ángeles a su llegada. En este nivel Timan adquiriría sus primeras alas como “Ángel Vengador”.
- Por último comentar que el juego es en 3D pero con la cámara en tercera persona típico de los juegos de plataformas.

La Tecnología

- Este aspecto fue sencillo de definir, el juego va a ser ejecutado en un iPad por lo tanto estaba establecida desde el principio, así que más que una selección fue una constante predefinida con la que trabajar y tener en cuenta en el resto del diseño del juego.

616

Control

- El jugador como se ha explicado anteriormente se pone en el papel del héroe "Timan", a través de una vista en tercera persona moverá el personaje por los distintos niveles.
- El jugador podrá mover el personaje hacia la derecha y hacia la izquierda además de poder saltar. El personaje estará sometido a las leyes de la física (esencialmente la fuerza de la gravedad).
- El juego como se ha comentado, está organizado por niveles, los cuales corresponden a los distintos ambientes por los que el personaje irá avanzando. El jugador progresará a través de ellos acumulando puntos que podrán provenir de diferentes fuentes.
- El jugador al morir aparecerá en el último punto de guardado (checkpoint).

617

Puntuación

A) Vidas

- Al comienzo de cada nivel se disponen de tres vidas. En el nivel normal las vidas actúan como bonificadores al final de cada nivel. Por cada una que se disponga al finalizar el mismo, se recibirá una determinada cantidad de puntos.
- En el nivel difícil aparte de ser bonificadores al final de cada nivel, las vidas permiten o no continuar una partida. Mientras quede alguna vida, se podrá seguir con el juego, si queda una vida y se muere, se terminará la partida.
- A lo largo de todos los niveles existen vidas que permiten reemplazar las perdidas. No se podrán acumular más de tres vidas en todo momento.
- Por cada vida conservada al final del nivel se obtendrán 50 puntos.



Vida del juego

618

B) Tiempo

- Otro de los aspectos que permite obtener puntuación al finalizar un nivel es el tiempo empleado en finalizarlo.

- Existen una serie de intervalos de tiempo que otorgan una bonificación de puntos:

Tiempo		Puntuación
	< 5 minutos	1000 puntos
> 5 minutos	< 7 minutos	750 puntos
> 7 minutos	< 10 minutos	300 puntos
> 10 minutos	< 15 minutos	100 puntos

C) Estrellas

- Las estrellas son elementos del juego que se pueden recolectar para obtener puntos. Se encuentran distribuidas por el mapa.

- Al ser elementos que proporcionan puntuación y por lo tanto influyen en obtener un buen resultado, no todas están visibles al pasar por las zonas principales del juego. Hay muchas estrellas que se conseguirán explorando diferentes zonas del nivel.

- Por cada estrella conseguida se obtendrán 10 puntos.

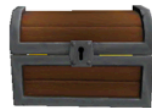


Estrella

D) Cofres

- Los cofres al igual que las estrellas otorgan puntos al ser recogidos. Por unidad otorgan 100 puntos.

- Debido a que tienen un alto valor en puntuación comparado con otros elementos, existen pocos y en zonas especiales que será necesario encontrar.



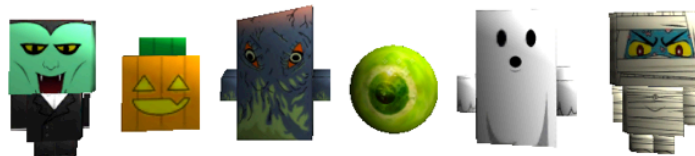
Cofre de recompensas

621

E) Enemigos

- Matar enemigos también proporciona puntos. Por cada enemigo muerto se obtendrán 10 puntos extras.

- Para matar a un enemigo habrá que caer encima, cualquier otro tipo de contacto con un enemigo supondrá la pérdida de una vida.



Enemigos

622

F) Resumen de puntuaciones

Elemento	Condición	Puntuación
Vida	Por cada vida	50 puntos
Tiempo (t) en minutos	$t < 5$	1000 puntos
	$5 < t < 7$	750 puntos
	$7 < t < 10$	300 puntos
	$10 < t < 15$	100 puntos
Estrella	Por cada estrella	10 puntos
Cofre	Por cada cofre	100 puntos
Enemigo	Por cada enemigo	10 puntos

623

Diseño

- El diseño de la interfaz se puede dividir en dos partes, la primera que muestra el menú principal del juego y la segunda que es la interfaz de una partida en marcha.
- El primer menú que se mostrará será el de creación del usuario. Una vez creado se mostrará el menú principal.
- El menú principal da acceso a jugar una partida pudiendo elegir el nivel y comenzar la partida. También permite ver el top 5 de mejores puntuaciones, estadísticas de juego, los créditos, la ayuda, cambiar los datos del jugador y borrar todos los datos almacenados (top 5 puntuaciones y estadísticas de juego).
- La interfaz que se muestra en una partida muestra el número de vidas, enemigos muertos / total enemigos, estrellas recogidas / estrellas totales, cofres recogidos / total cofres.

624

Menú principal

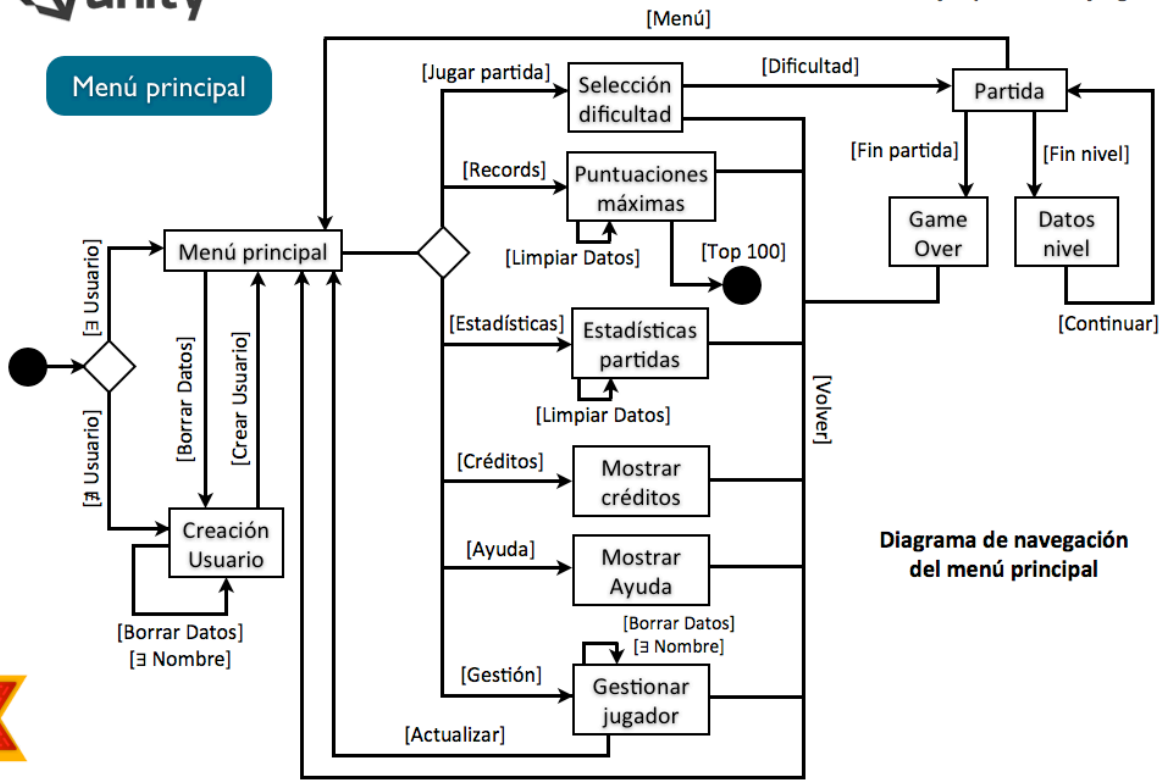
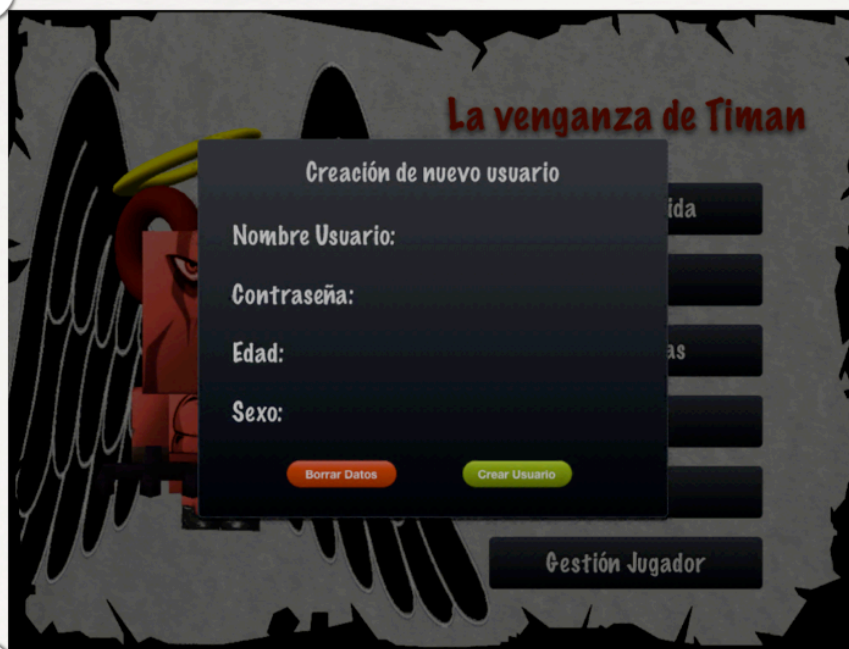


Diagrama de navegación del menú principal

Creación Usuario



Menú principal



627

Selección dificultad



628

Puntuaciones máximas

La venganza de Timan
Top 5 (Local)

Volver | Limpiar datos | Top 100

Nivel fácil			
Posición	Puntos	Tiempo	Puntuación final
1	1000	1000	2000
2	1000	750	1750
3	800	750	1550
4	750	750	1500
5	500	500	1000

Nivel difícil			
Posición	Puntos	Tiempo	Puntuación final
1	1000	1000	2000
2	1000	750	1750
3	800	750	1550
4	750	750	1500
5	500	500	1000

629

Estadísticas partidas

La venganza de Timan
Estadísticas

Volver | Limpiar datos

Nivel fácil		Nivel difícil	
Total partidas jugadas:	5	Total partidas jugadas:	5
Total enemigos muertos:	100	Total enemigos muertos:	100
Total estrellas conseguidas:	400	Total estrellas conseguidas:	400
Total cofres conseguidos:	20	Total cofres conseguidos:	20
Total muerto por enemigos:	5	Total muerto por enemigos:	5
Total muerto por entorno:	10	Total muerto por entorno:	10
Enemigos muertos/partida:	20	Enemigos muertos/partida:	20
Estrellas/partida:	80	Estrellas/partida:	80
Cofres/partida:	4	Cofres/partida:	4
Muerto por enemigo/partida:	1	Muerto por enemigo/partida:	1
Muerto por entorno/partida:	2	Muerto por entorno/partida:	2

630

Mostrar créditos



631

Mostrar ayuda



632

Gestión Jugador



633

Partida



634

Actualizar estadísticas

¡ Nivel superado !

Puntuación del nivel 1 **Continuar**

Estrellas conseguidas:	45 x 100	-----	450 puntos
Cofres conseguidos:	2 x 100	-----	200 puntos
Vidas restantes:	2 x 50	-----	100 puntos
Enemigos muertos:	15 x 10	-----	150 puntos
Tiempo empleado:	7:41	-----	300 puntos

			Total: 1250 puntos
			Puntuación acumulada: 1250 puntos
¡Nuevo Record nivel normal! Posición: 4			

635

Game Over

GAME OVER

< Volver

Estrellas conseguidas:	0 x 10	-----	0 puntos
Cofres conseguidos:	0 x 100	-----	0 puntos
Vidas restantes:	5 x 50	-----	150 puntos
Enemigos muertos:	0 x 10	-----	0 puntos
Tiempo empleado:	05 : 55	-----	750 puntos

			Total: 900 puntos
			Puntuación acumulada: 4750 puntos
¡Nuevo Record local! - Nivel normal. Posición: 1			
Tu posición (Chichu) en el Top 100 es: 1 con: 4750 puntos			

636

Interfaz partida

"Movimiento"



- La interfaz de una partida viene mostrada parcialmente en la ayuda incorporada dentro del juego. En ella se indica cómo se realiza el movimiento.

- Al ser un juego para iPad, la interfaz es táctil, aprovechando esto el movimiento del personaje se realiza mediante toques en la pantalla.

- Como se muestra en la ayuda, un toque a la derecha del personaje hará que se desplace hacia la derecha. Un toque a la izquierda del personaje hará que se desplace hacia la izquierda. Un toque sobre el personaje o con tres dedos en cualquier parte de la pantalla, parará el personaje.

"Saltos"



- Otra interacción que se permite en el juego son los saltos.

- El personaje realiza dos tipos de saltos, un salto normal, para ello se pulsa la pantalla con dos dedos.

- El otro tipo de salto es un salto doble que se realiza pulsando con dos dedos en la pantalla dos veces.

- En cuanto al resto de elementos que aparecen en la interfaz, la mayor parte se encuentran situados en la parte superior de la pantalla para facilitar su lectura y para que no interfieran en el movimiento del personaje.
- En la parte inferior de forma centrada está el botón de acceso al menú principal que permite salir de la partida.
- Para pausar el juego es suficiente con pulsar el botón "Home" del iPad. Para reanudar el juego bastará con pulsar sobre el icono de la aplicación.
- En las siguientes páginas se muestra la información que aparece en la pantalla.

639



640



641



642



643



644



645

Superficies

- Las superficies son un conjunto de elementos con los que "Timan" puede tener algún tipo de contacto. La mayor parte de estas superficies permiten al personaje progresar en los niveles o alcanzar determinados objetos.

A) Plataformas

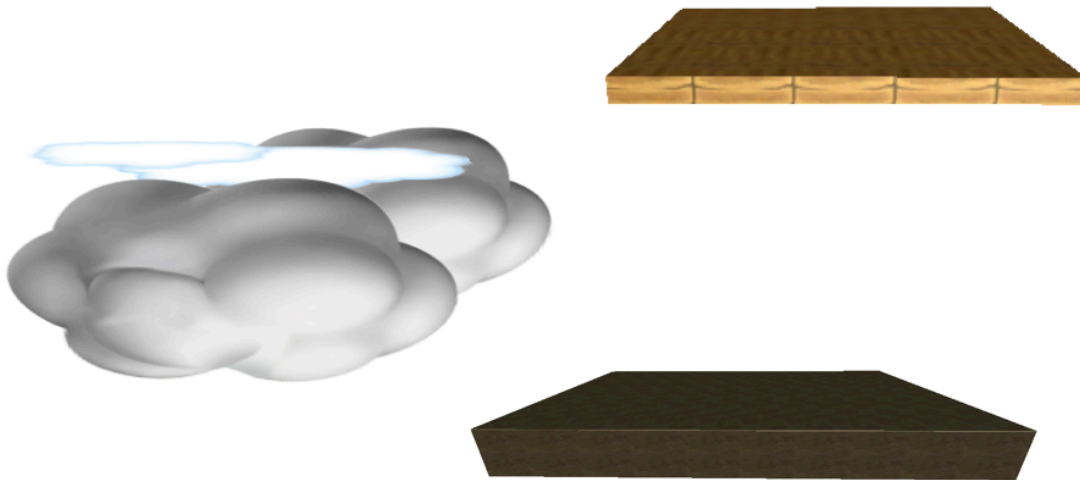
- Las plataformas son el elemento fundamental del juego. Permiten diseñar los niveles y dar ambiente al juego.

- En función del nivel, la apariencia de la plataforma varia, pero en esencia todas son iguales, bloques que sustentan al personaje y le permiten dar saltos para alcanzar otros elementos o plataformas.

- Aparte de diferir en la apariencia, también existen baldosas de distintos tamaños dentro del juego.

646

- Ejemplos de plataformas del juego:



647

B) Superficies mortales

- Entre las superficies mortales para Timan se tiene la lava. Pese a que originalmente Timan es un demonio, con su transformación ha perdido su capacidad de tolerar temperaturas extremas.

Caer sobre la lava o ser alcanzado por ella supondrá la pérdida de una vida.

La lava se encuentra en los niveles 1 y 3.



Ejemplo de lava

648

- Otra superficie mortal es el foso de pinchos. Como ocurría con la lava, si se cae sobre el foso de pinchos se perderá una vida.

Para evitar caer sobre ellos habrá que calcular bien los saltos y en muchos casos habrá que utilizar la habilidad especial de Timan del doble salto.

Estos fosos se encuentran distribuidas a lo largo del nivel 2.

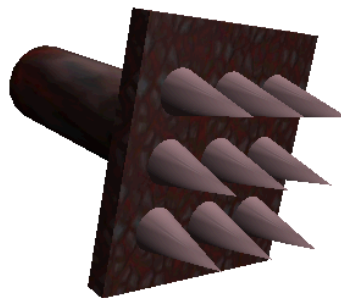


Foso de pinchos
mortales

649

- La siguiente trampa que existe en el juego son las trampas de pinchos. Estas trampas son superficies mortales que se desplazan en determinadas direcciones. Si en uno de esos desplazamientos alcanzan al jugador, se perderá una vida.

Estas trampas se encuentran en el nivel 2, en una zona especial del mapa que se llama "House of Horrors", una casa en la que será una pesadilla encontrar la salida.



Trampa de pinchos

650

- Otro elemento relacionado con el anterior es la sierra. La sierra al igual que la trampa de pinchos se encuentra dentro de "House of Horrors" en el nivel 2. Como toda sierra, el problema principal reside en situarse en la trayectoria en la que avanzan las cuchillas.

Si las cuchillas impactan con Timan, se perderá una vida.



Sierra

651

C) Otras superficies

- Aparte de las baldosas existen otro tipo de superficies sobre las que se puede pasar sin perder vidas. Estas superficies son las cajas, elementos que aparecen en puntos estratégicos del mapa que permiten alcanzar recompensas extras que no forman parte de la misión principal del juego (que es llevar a Timan al cielo).

- Hay cajas que están dispuestas en zonas que a simple vista no se ven y que requieren explorar el nivel y arriesgarse a encontrarlas. Pueden tener encima de ellas elementos valiosos como vidas que son fundamentales en el nivel difícil del juego.

- Existen dos tipos de cajas en el juego, las de madera y las de cartón. La diferencia entre ellas no es meramente estética sino que las de madera son permanentes mientras que las de cartón duran 2 segundos una vez que se pasa por encima de ellas.



Caja de madera



Caja de cartón

652

Enemigos

- Como enemigos se pueden distinguir de dos tipos, aquellos que pueden ser eliminados y los que no.
- Dentro de los primeros se tienen el conjunto de criaturas o “monstruos” que tratan de impedir que Timan alcance su objetivo. Estos enemigos como se comentó se pueden eliminar cayendo encima de ellos, pero pueden provocar la pérdida de una vida si alcanzan lateralmente al personaje.
- Los enemigos que se pueden eliminar generan una cantidad de puntos al morir.



Conjunto de enemigos que se pueden eliminar

653

- Como enemigos que no pueden ser eliminados dentro del juego se encuentran algunos fenómenos naturales, como los rayos. Los rayos son elementos que forman parte del juego en el nivel 3, son objetos que caen mientras Timan asciende al cielo entre las nubes.
- Si un rayo impacta sobre Timan se considerará que muere y en consecuencia perderá una vida.
- Los rayos no podrán ser eliminados aunque desaparecerán tras cierto tiempo. Los rayos caerán de forma aleatoria durante todo el nivel.



Rayo (enemigo que no se puede eliminar)

654

Otros elementos principales

A) Puntos de guardado (checkpoints)

- Uno de los elementos que se hace imprescindible en un juego de plataformas es el “punto de guardado” de la partida. Estos puntos almacenan una posición fija determinada del mapa y sirven para guardar lo que es el avance de una partida en un determinado nivel.
- Si por alguna casualidad se pierde una vida, la posición desde la que se continuará no será la misma en la que se perdió la vida sino que será el último punto de guardado activado.
- Al activarse un punto de guardado desaparecerá de la partida. Existen varios puntos de guardado por cada nivel.



Punto de guardado

655

B) Estrellas

- Las estrellas como se comentó al explicar el sistema de puntuación del juego son elementos que proporcionan puntos al ser recogidas. La mayor parte de las estrellas no se encuentran en el camino natural del personaje. Algunas se observan al realizar saltos en determinadas zonas de los niveles y otras se descubrirán por exploración del nivel o por errores normales derivados de una partida.
- Una vez que una estrella es recogida, desaparece del nivel y produce un aumento de puntos en el marcador del jugador (10 puntos por estrella). A más estrellas recogidas mayor probabilidad de obtener una de las cinco mejores puntuaciones correspondientes a cada nivel de dificultad.



Estrella

656

C) Cofres

- Los cofres como las estrellas suponen puntos en el juego. El número de cofres por nivel es muy inferior al de estrellas y por lo tanto su valor es superior (100 puntos).
- Los cofres están situados por todo el mapa y al igual que las estrellas suponen un reto el recogerlos en la mayoría de los casos.
- Los cofres al ser recogidos desaparecerán del nivel e incrementarán la puntuación del jugador.



Cofre

657

D) Vidas

- Las vidas en el juego son un elemento fundamental. Desde el punto de vista de la dificultad elegida, si se ha elegido el nivel difícil, las vidas supondrán la posibilidad de seguir o no la partida. Como se ha comentado en el tema, al empezar cada nivel el número de vidas es 3, al morir, se pierde una vida, cuando queda una vida si se muere y se está en el nivel difícil la partida terminará.
- Si el nivel elegido es el normal, las vidas actúan como bonificadores de nivel, es decir, si se acaba el nivel, por cada vida que se mantenga se ganarán 50 puntos adicionales. Por el contrario si no quedan vidas y se muere, la partida no terminará, pero la posibilidad de obtener una bonificación terminará.
- Las vidas como el resto de elementos que se recogen, una vez recogidas desaparecen. Estas vidas se encuentran distribuidas por el mapa, algunas se recogen fácilmente, otras no, algunas se ven fácilmente, otras no.



Vida

658

Elementos secundarios (decorativos)

- Existen multitud de elementos secundarios o decorativos a lo largo de cada nivel. Algunos son comunes y compartidos entre los niveles, mientras que otros son específicos de un nivel. Hay ciertos elementos que aparecen en todos o varios niveles aunque tienen una mayor importancia en alguno de ellos.

- Estos elementos no influyen en el juego, es decir, no darán puntos, no quitarán vidas y no se podrá interaccionar con ellos ya que no se encuentran en el camino permitido del jugador.

- A continuación se muestran ejemplos de elementos decorativos que se encuentran en el juego.

659

Ejemplo de elementos decorativos



Antorcha



Ataúd



Calavera



Puerta



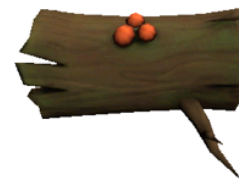
Pilar



Lápida



Valla



Tronco

660

Nivel I - "Escapar del Infierno"

A) Descripción

- Es el primer nivel del juego y lugar donde se desarrolla la introducción de la historia del héroe. Este nivel como el propio nombre lo indica se desarrolla en el infierno, antiguo hogar de Timan.

- El nivel comienza en la entrada a la sala del trono de Timan que queda a su espalda, desde ahí intentará alcanzar su objetivo que es la salida del infierno. Para ello tendrá que luchar contra los enemigos que se encuentre a su paso.

- Conforme asciende se irá encontrando con recompensas y con puntos de guardado del nivel, pero el camino no será sencillo, se irá encontrando con obstáculos naturales propios del lugar en el que se encuentra, alguno de ellos mortales.

661

- Deberá usar sus poderes especiales para superar muchos de los obstáculos y sobre todo para no caer en la lava que corre por el fondo del nivel.

- Existe una zona en la que la lava no solo circula sino que sube y baja, así que se deberá tener cuidado y calcular el momento oportuno para realizar los saltos si no se quiere acabar abrasado por el fuego.

- Los enemigos se encuentran por todo el camino natural de salida del nivel, pero no son los únicos caminos ni los más lucrativos. Existen alternativas para generar mayor puntuación y recuperar aquellas vidas que se hayan perdido por errores de cálculo.

- El nivel representa la revelación que sufre Timan y por tanto tendrá que realizar "saltos de fe", donde el camino parece que no tiene salida, un salto puede revelar la nueva ruta de escape.

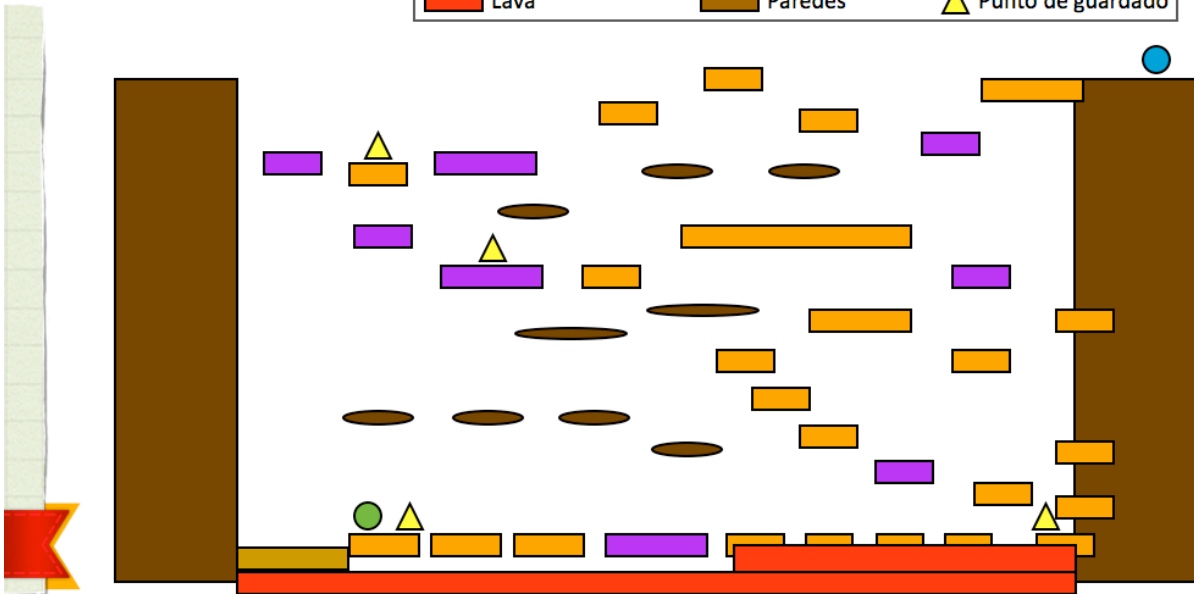
- Después de ascender a la superficie de la tierra se accederá al siguiente nivel.

662

B) Diseño del nivel

Leyenda

Plataforma	Zona de cajas	Comienzo nivel
Plataforma móvil	Trono	Fin nivel
Lava	Paredes	Punto de guardado



663



Nivel 1 - "Escapar del Infierno"

664

Nivel 2 - "Festín de muertos"

A) Descripción

- Timan ha logrado escapar del infierno y ya se encuentra en la superficie. Este nivel es una especie de purgatorio o entrada al infierno por lo que "caminará" entre criaturas de la noche, momias y seres malignos como calabazas de halloween diabólicas.

- A diferencia del nivel anterior, el nivel 2 es lineal, no se trata de ascender a ninguna parte para alcanzar el fin del nivel. En este nivel se irá saltando de plataforma en plataforma intentando evitar la caída en los fosos mortales, todo ello esquivando o evitando entrar en contacto con los enemigos que pueden provocar la muerte mientras entorpecen el camino.

- Al igual que en el nivel 1, habrá zonas especiales con estrellas para aumentar la puntuación, de nuevo, algunas más o menos visibles, otras no, algunas sencillas de recoger, otras, no tanto.

665

- También estarán repartidas varias vidas en puntos estratégicos para recuperar las que se hayan perdido, así como puntos de guardado para evitar tener que empezar desde muy atrás en el nivel cuando se pierde una vida.

- Algo especial y que caracteriza a este nivel es el ambiente, al llamarse "Festín de muertos", el nivel está lleno de tumbas, lápidas, y otros adornos típicos de escenas de cementerios.

- Por último pero no por eso menos importante, el nivel posee una zona especial llamada "House of Horrors" y pretende recrear en la mente del jugador la típica atracción de una casa "encantada" repleta de trampas mortales. Es una zona complicada y en la que se requiere habilidad y sobre todo paciencia, calcular mal cuándo se debe pasar de una parte a la siguiente puede suponer el perder una vida y tener que repetir aquella parte de la casa que tanto ha costado superar.

666



Nivel 2 - "Festín de muertos"

669

Nivel 3 - "El Ángel Redentor"

A) Descripción

- El nivel 3 es el último del videojuego de ejemplo. En él, Timan se propone llegar a su nuevo hogar, "El Cielo". Tras alcanzar en el nivel 2 el camino de ascenso al cielo, ahora tiene que ascender hasta las puertas del cielo, donde le esperarán para darle la bienvenida a su nuevo hogar.

- Este nivel es un nivel más parecido al primero en el sentido de que se trata de ir subiendo por distintas plataformas en las que hay enemigos que tendrá que esquivar o eliminar. Pero también incorpora un elemento especial e importante del nivel que son unos rayos que caen y que actúan de forma parecida a como lo hacían las trampas del nivel 2, por lo que se puede considerar este último nivel como una mezcla de los dos anteriores pese a tener un mayor parecido con el nivel 1.

670

- Como ocurre en los otros dos niveles, en este último nivel sigue habiendo estrellas, cofres y vidas distribuidas por todo el mapa, mas o menos accesibles y con mayor o menor dificultad para ser alcanzadas.

- También se dispone de puntos de guardado que permitirán recuperar una posición mas o menos avanzada si accidentalmente se cae, muere por culpa de un enemigo o se es alcanzado por un rayo.

- Los rayos que se han mencionado y que aparecen como novedad en este nivel, pueden caer por la derecha, izquierda o en la posición en la que se encuentra Timan. Caen de forma aleatoria en una de estas tres zonas y el ser alcanzados o tocar alguno de ellos supondrá la pérdida de una vida.

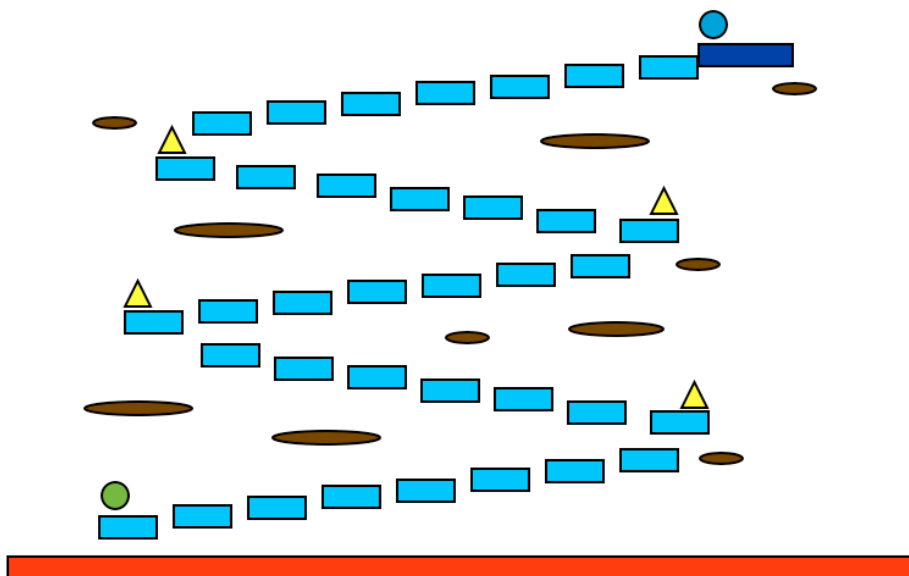
- En la parte inferior del mapa existe una zona alargada de lava y que supondrá el límite inferior del mapa por si Timan no logra alcanzar una nube en una posible caída. Esta lava como en el primer nivel será mortal y asemeja a lo que sería una caída desde el cielo y una vuelta al infierno.

671

B) Diseño del nivel

Leyenda

Nube	Comienzo nivel	Zona de cajas
Puertas del cielo	Fin nivel	
Lava	Punto de guardado	



672



Nivel 3 - "El Ángel Redentor"

673

Introducción

- El sonido es uno de los elementos de un juego que normalmente se suele dejar para el último momento. Muchas veces no se le da mucha importancia aunque el sonido sea un aspecto fundamental en el desarrollo de un videojuego.
- El sonido y en especial la música va a servir de unión entre la historia del juego, la ambientación y la partida. El sonido transmite emociones, puede sumergir al jugador en un determinado ambiente y aumentar la tensión de determinadas situaciones.
- Una buena elección del sonido en un videojuego puede hacer que este pase de ser un juego normal a un gran juego.

674

- Al igual que por ejemplo no se concibe una película de terror sin sonidos que creen miedo o una banda sonora que genere tensión, en un videojuego el sonido juega el mismo papel. Muchas de las emociones que queremos transmitir a un jugador no se lograrán sin una buena elección del sonido.

- Debido a que el ejemplo de videojuego del curso tiene un motivo más didáctico y pretende ser muestra de juego que se puede realizar en Unity la elección del sonido se ha basado en seleccionar música bajo licencia de libre uso no comercial.

- Dentro de lo que aquí se ha denominado sonido encontraremos los típicos sonidos al realizar una determinada acción o sufrir una consecuencia de alguna de ellas con el personaje, como por ejemplo saltar, eliminar a un enemigo o morir por culpa de uno de ellos, caer a la lava, morir por impacto de un rayo, ...

- Por otro lado dentro del sonido se tendrá la música que estará de fondo durante toda la partida.

675

Estilo de sonido

- Para las acciones y sus consecuencias, se han elegido sonidos de tipo comic que van acorde al tono de humor en general del videojuego.

- Para la música de fondo como estilo se ha elegido el rock muy usual en este tipo de juegos de plataformas ya que confiere dinamismo durante su uso.

676

Fuentes de sonido

- Las fuentes de sonido se han obtenido de internet. Se han realizado numerosas búsquedas y pruebas en diferentes páginas. Al final las que se han seleccionado han sido:

- Efectos de sonido:
<http://soundbible.com/>
<http://www.grsites.com/archive/sounds/>
- Sonidos:
<http://www.jamendo.com/es>
<http://www.newgrounds.com/>
<http://www.nosoapradio.us/>

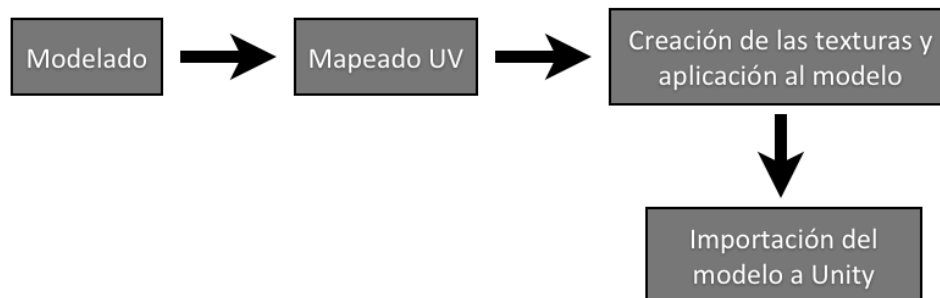
677

Creación de los elementos del juego

Introducción

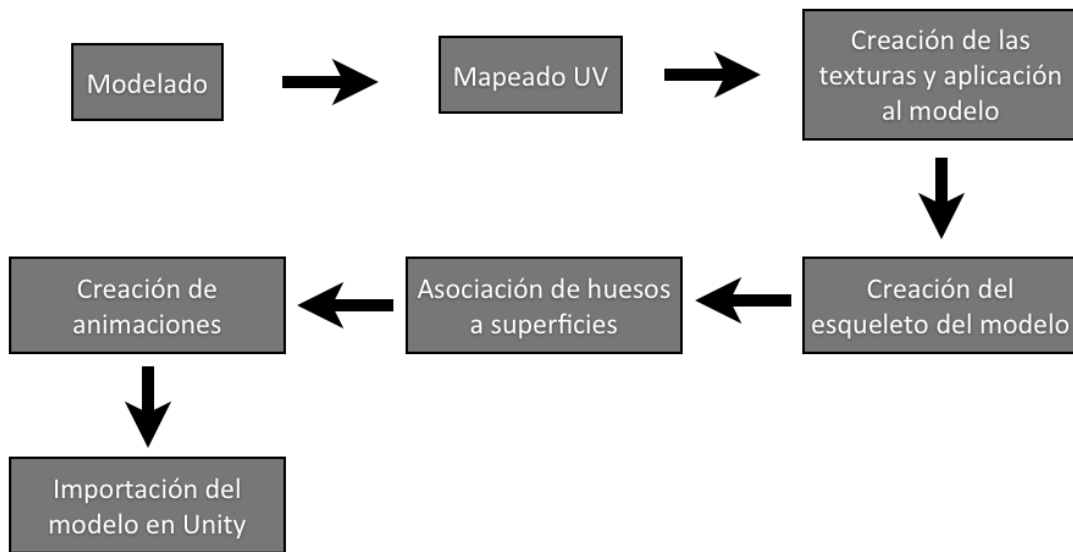
- La creación de cualquier elemento del juego es un proceso que engloba una serie de fases. En función del tipo de elemento creado, las fases varían ligeramente.

- En el ejemplo del videojuego, se han seguido dos procesos en función del tipo de objeto creado. Para los objetos simples que no requieren animación como las superficies el proceso ha sido el siguiente:



678

- Otros elementos del juego como el personaje principal o los enemigos tienen un proceso diferente que es el que se indica a continuación:



Modelado

- El modelado es un paso fundamental en la creación de un videojuego. Los personajes, objetos y entorno que se crean es lo que el jugador percibirá y es donde en gran parte se transmitirán las sensaciones deseadas.

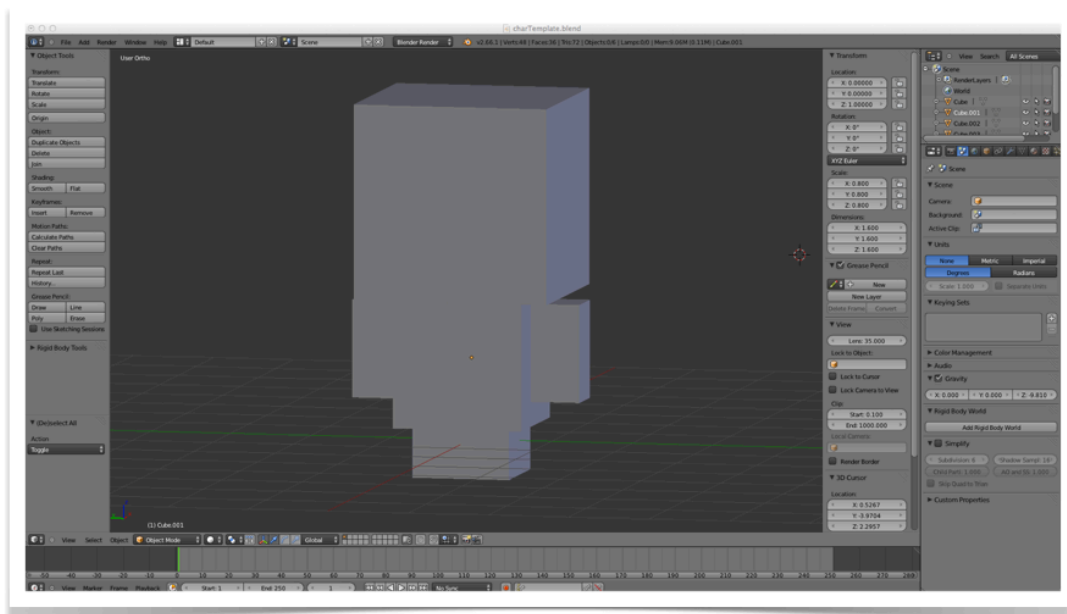
- Por un lado influirá el modelado de los objetos de forma individual pero por otro la composición que se haga de un conjunto de ellos en cada uno de los niveles.

- Para modelar todos los elementos de un juego existen gran cantidad de aplicaciones en el mercado, algunas disponibles para todos los sistemas operativos y otras específicas para alguno de ellos.

- Entre las habituales se encuentran 3ds Max, Maya, ZBrush, Mudbox, Blender. Pero existen multitud de alternativas como Sculptris, K-3D, Art Of Illusion, Zmodeler, Wings, Cheetah 3D, Shade 3D for Unity, etc ...

- Para el ejemplo del videojuego se ha elegido como herramienta de modelado Blender. Blender es un software libre multiplataforma que permite el modelado y animación en 3D.

- Se ha elegido esta herramienta por ser software libre, porque dispone de bastante documentación en internet y por su calidad como herramienta. Es un programa que para los usuarios con poca experiencia puede impresionar (sobre todo la interfaz) la primera vez que se usa, pero es una aplicación que ha evolucionado mucho en cuanto a su aspecto y con la que con un poco de tiempo y esfuerzo se pueden realizar trabajos de calidad.

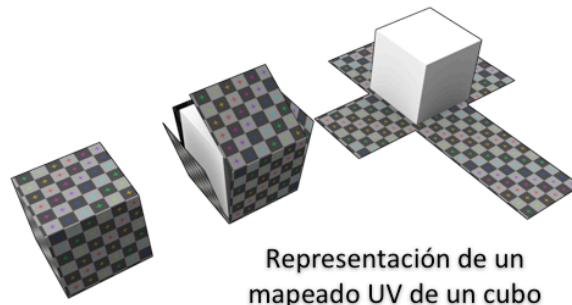


Ejemplo de modelado de personaje con Blender

Mapeado UV, creación y aplicación de texturas

- El mapeado UV es un proceso de modelado en 3D que consiste en crear una imagen en 2D que sea la representación de un modelo en 3D.

- Este proceso se realiza con Blender de forma sencilla. Una vez creado el modelo en 3D, se crea el mapa UV, se exporta a .png y mediante un programa de edición de imágenes se puede crear la textura deseada en ese .png. Una vez creada la textura, se importa a Blender que la aplicará a la superficie visible del modelo.



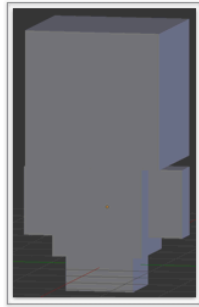
683

- Para la creación de las texturas se pueden utilizar programas para la manipulación de imágenes como por ejemplo Gimp, Inkscape, Photoshop, Fireworks, etc.. Para el ejemplo de videojuego se han elegido Gimp e Inkscape como programas de manipulación de imágenes ya que son software libre que permiten realizar sobradamente todo el trabajo requerido para el videojuego. Comentar que para la creación de las texturas a aplicar a los modelos se ha utilizado solamente Gimp.

- Una vez generado el .png con el mapa UV desde Blender, se puede modificar desde Gimp para crear la textura que se quiera para el modelo creado.



684



Modelado en 3D



Mapa UV a partir del modelo

(Creado con Blender)



Modelo con la textura aplicada

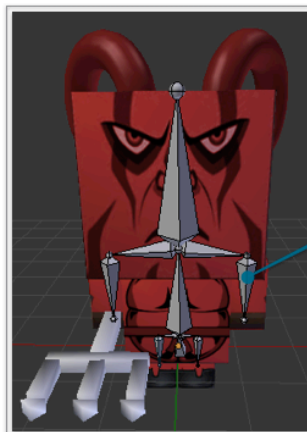


Mapa UV con textura creada

(Modificado con Gimp)

Creación del esqueleto del modelo

- Para poder animar un modelo es necesario crear un sistema de huesos que formen el esqueleto. Blender permite crear huesos que se irán enlazando por todo el modelo hasta formar el esqueleto.



Timan con el esqueleto creado

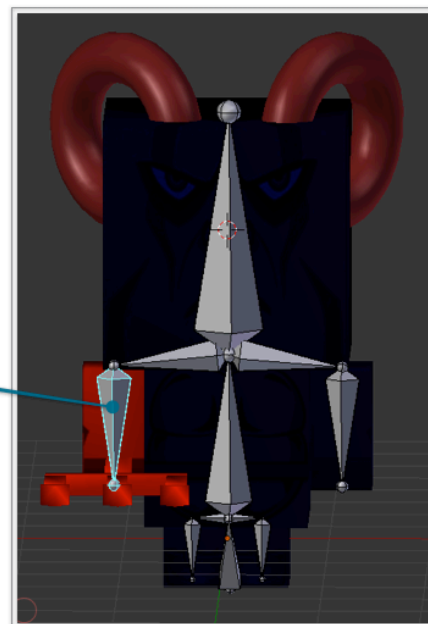
Los diferentes huesos del modelo se unen hasta formar el esqueleto.

Asociación de huesos a superficies

- Cada hueso deberá asignarse a las superficies que se desee que se muevan cada vez que el hueso cambie de posición.
- Esta parte también se realiza dentro de Blender y consiste en colorear aquellas zonas (rojo implica que las zonas se asignan al hueso seleccionado, azul implica que no se asignan al hueso seleccionado) que se desee se muevan con un determinado hueso.
- Una vez asociadas las superficies, Blender se encarga de deformar el modelo de forma automática cada vez que el hueso cambie de posición. Los huesos cambiarán de posición durante la animación, de esta forma al animar un modelo, lo que se hace es animar los huesos, como las superficies están asociadas a los huesos al moverse este se moverán las superficies asociadas.
- Blender también permite asignar de forma automática las superficies a un determinado esqueleto, pudiendo luego modificar como se ha indicado (por colores) aquellas asociaciones incorrectas.

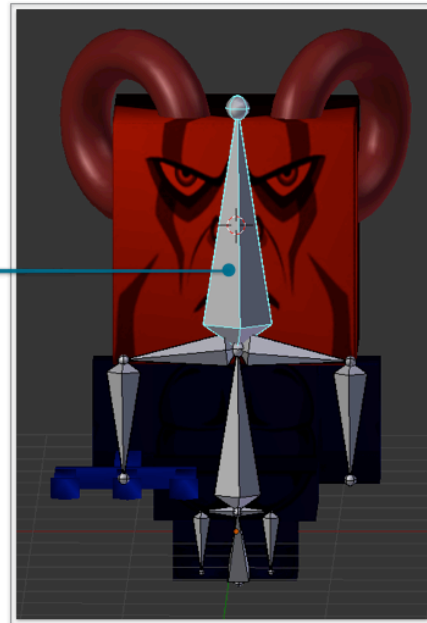
687

En este ejemplo, el hueso del brazo derecho está seleccionado y a él se han asociado el brazo derecho del personaje y el tridente, que como se ve aparecen coloreados de rojo. Así al moverse el hueso del brazo derecho se moverá el tridente y el brazo derecho.

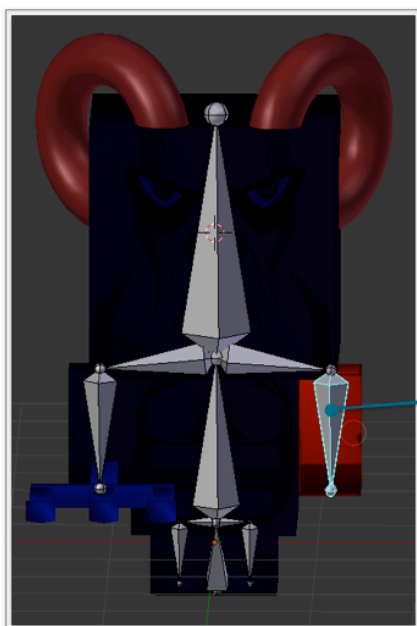


688

En este caso, el hueso de la cabeza es el hueso seleccionado y en este caso sólo se le ha asociado la cabeza (de color rojo) para moverse con él.



689



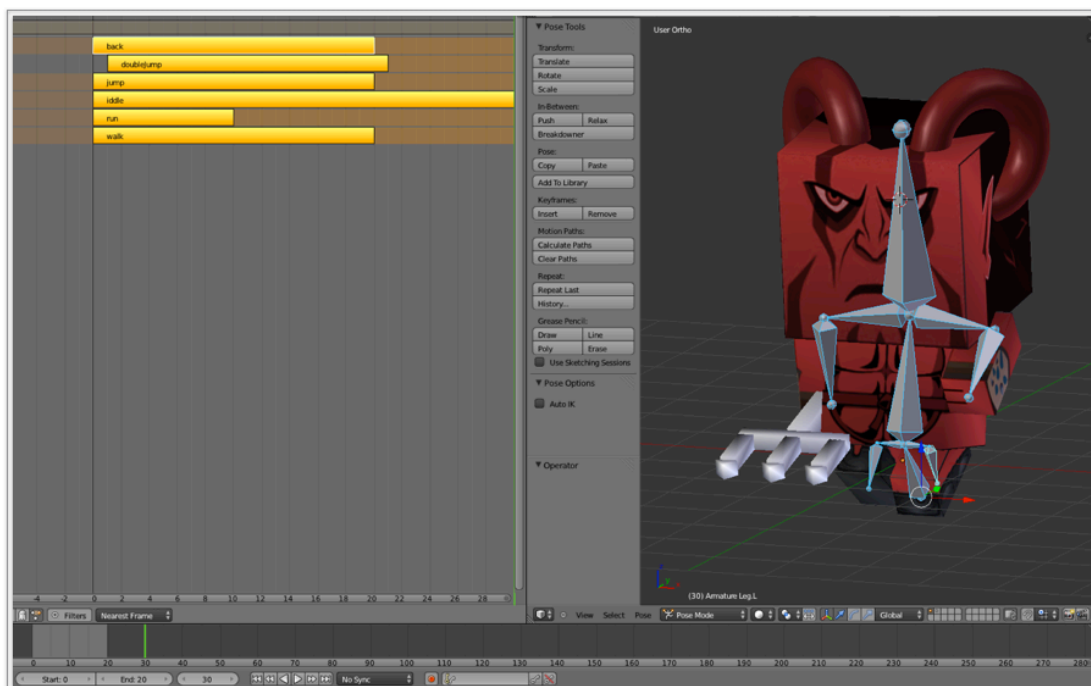
Otro ejemplo en el que ahora es el hueso del brazo izquierdo seleccionado y sólo el brazo izquierdo se asocia.

690

Creación de animaciones

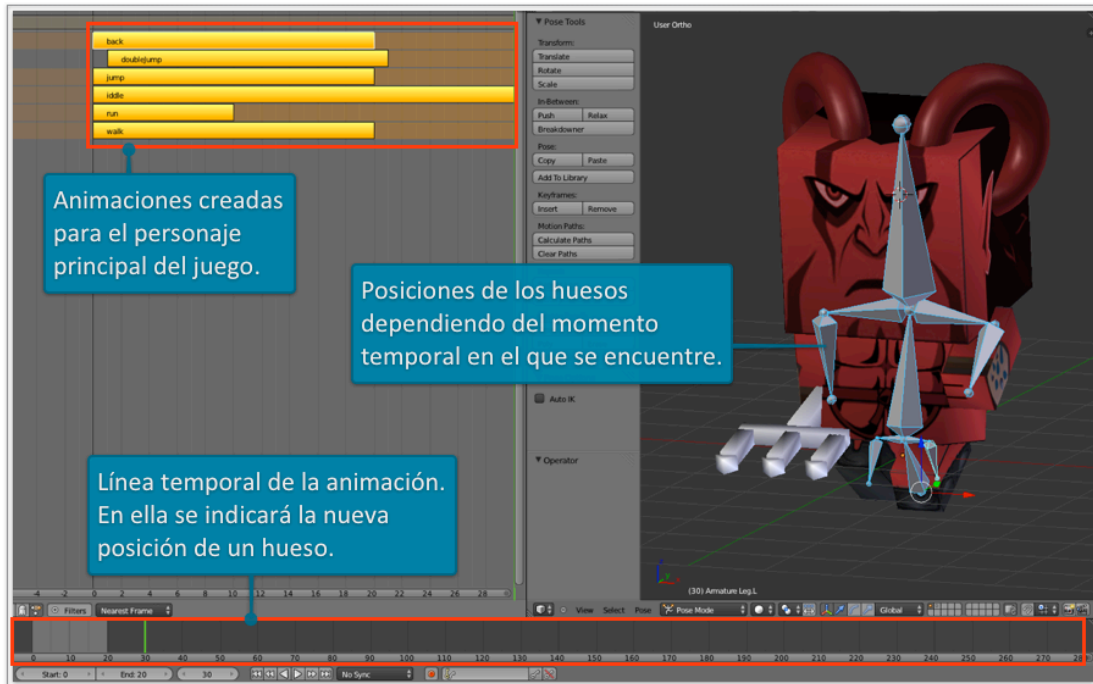
- Las animaciones como se acaba de comentar se crearán en Blender (casi todo el proceso de modelado se ha realizado en Blender, a excepción de la creación de la textura para el modelo. Incluso desde Blender se pueden crear materiales y asignarlos al modelo).
- Las animaciones dependiendo del tipo de modelo que se disponga y del juego que se quiera crear son muy importantes ya que si un modelo se pretende que sea realista, si la animación (que no deja de ser que una simulación de acciones, como el andar, correr, saltar, etc...) no lo es, el modelo no lo parecerá.
- Animaciones simples son muy sencillas de crear en Blender, basta con indicar en el tiempo la posición inicial y final de un hueso, la aplicación creará automáticamente el movimiento.

691



Ejemplo de animación de Timan

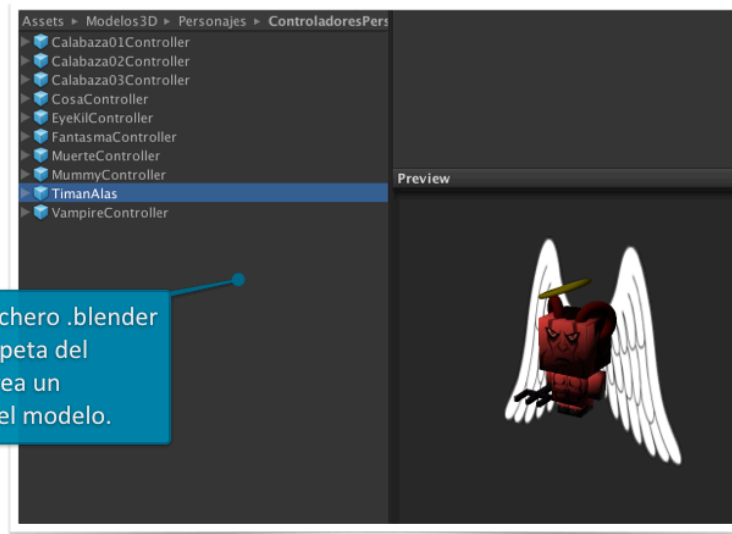
692



Ejemplo de animación de Timan

Importación del modelo en Unity

- Otra de las razones por las que se ha elegido Blender como aplicación para el modelado de objetos es que Unity es capaz de importar directamente los ficheros con extensión .blender creados.
- Así para importar un modelo creado con Blender bastará con arrastrar el fichero .blender dentro de la carpeta del proyecto de Unity que se desee.



Tras arrastrar el fichero .blender dentro de una carpeta del proyecto, Unity crea un prefabricado con el modelo.

- Las escenas del juego están organizadas como se indica en la imagen.

- Las escenas 00_aMenu_Inicial, 00_aMenu_Principal, 00_bJugar_Partida, 00_cRecords, 00_dEstadísticas, 00_eCreditos, 00_fAyuda y 00_iGestion_Jugador corresponden a la interfaz del menú principal.

- La escena 00_gGame_Over será la escena que se mostrará al terminar una partida en nivel difícil.

- La escena 00_fFin_nivel se mostrará al finalizar un nivel tanto en nivel normal como difícil.

