Universidad del País Vasco
Euskal Herriko Unibertsitatea

Facultad de Informática
Informatika Fakultatea

TITULACION: Ingeniería Técnica en Informática de Sistemas

Aplicación para dispositivos Android:

Parada en Boxes

Alumno: D. Borja Salazar Rey

Director: D. Germán Rigau Claramunt

Proyecto Fin de Carrera, febrero de 2013

Índice de Contenido

Índice de imágenes

1. INTRODUCCIÓN

Como muchos chicos de principios de la década de 1990 crecí pegado a mi Nintendo Game Boy. No sabría decir cuántas horas pase ayudando a Mario a rescatar a la princesa, intentando superar la puntuación máxima en el Tetris o peleando con mis amigos a través de una conexión por cable. Este pequeño hardware iba conmigo a todas partes. Mi pasión por los juegos me llevó a querer crear mis propios juegos, y hace dos años conseguí crear mi propia versión para PC del SuperPang[1], al que también tanto jugué.

A partir del 2010 los smartphones se habían convertido en el soporte perfecto para juegos pequeños, compitiendo con consolas portátiles como Nintendo DS o PlayStation Portable, esto me llevo a la idea de hacer una aplicación para estos dispositivos, un juego. Comencé investigar plataformas móviles para ver cuál sería la más indicada. El sistema iOS de Apple parecía un candidato excelente, sin embargo, caí en la cuenta de que es un sistema abierto donde solo podría compartir con los demás aquello que me permita Apple, además, no tenía un Mac, así que no podía desarrollar nada para iOS, entonces me encontré con Android.

Su entorno de desarrollo funciona con la mayoría de plataformas y cuenta con una comunidad de programadores cada vez más extensa. Puedo compartir mis aplicaciones sin tener que pagar nada a cambio y si quiero rentabilizar mi trabajo, puedo publicarlo y en cuestión de minutos, ponerlo a disposición de millones de clientes potenciales repartidos por todo el mundo.

Una vez llegado a este punto tenía que centrarme en las características que quería que tuviera mi juego. Lo primero que tenía que tener claro es el género, y opte por hacer un juego casual. Estos juegos tienen las características de que es fácil acceder a ellos, para ponérselos en bandeja a los usuarios que no sean jugadores habituales, así aumenta el número de jugadores potenciales. Las partidas no deben durar mucho tiempo, y su capacidad adictiva y sencillez hacen que los jugadores queden enganchados cuando lo prueban. Angry Birds (en la Ilustración 1) o Abduction son muy buenos ejemplos de lo que quería hacer, aunque cayendo demasiado en la sencillez.

[1] http://stargamer1138.files.wordpress.com/2010/08/super-pang-2.gif

Ilustración 1: Juego Angry Birds versión retro.

Quería que este juego me supusiera un reto, así que intente diseñarlo de tal modo que usara toda la tecnología que Android ponía en mis manos; pantalla táctil, multi-touch, acelerómetro, tratado de imágenes, efectos de sonido,… y todo, autoajustable a cualquier tamaño de pantalla y resolución posibles. Todo esto integrado en una temática de Formula 1 para hacerlo más atractivo a los ojos de los amantes de este deporte, y por qué no hemos visto ningún juego basado en la temática de una parada en boxes.

En definitiva, el juego tratara de una serie de pruebas a realizar en el menor tiempo posible. Para ello contamos con un cronometro ajustado hasta los milisegundos. Usamos eventos táctiles, como pulsaciones o movimientos de uno o varios dedos por la pantalla, inclinando el teléfono o resolviendo pequeñas pruebas de reflejos.

2. Documento de Objetivos del Proyecto (DOP)

2.1 Objetivos del Proyecto

Este proyecto consiste en la creación de un juego utilizando todas las herramientas posibles proporcionadas por Android. El juego se diseñará en 2 dimensiones, contará de un menú inicial donde se podrá elegir entre las opciones; Jugar, Puntuaciones y Ayuda. La parte de jugar llevará todo el peso de la aplicación y es donde se desarrollará toda la acción. Habrá que hacer transiciones entre distintas pantallas para cada uno de los distintos mini juegos. Además, llevará la cuenta del tiempo transcurrido con un preciso cronometro. El reloj de parará cuando se hayan terminado todas las pruebas. Si ha superado alguno de los mejores tiempos, guardaremos el record en el lugar que le corresponde. La parte de las puntuaciones mostrará los cinco mejores tiempos realizados en el juego. Estos tiempos seguirán guardados aún reiniciando la aplicación. La parte de ayuda mostrará las instrucciones de cada prueba; donde, cuándo, y de qué manera tienes que pulsar en cada momento.

Hay que tener en cuenta que esta aplicación de desarrollará para versiones de Android 2.1 o superiores, así que es necesario basarse en API's para estas versiones, y que no se debe necesitar nada más que el archivo de instalación (.apk) para poder disfrutar de el juego en cualquier terminal.

2.2 Método de Trabajo

En esta sección describiremos los métodos que vamos a usar para la realización del proyecto, divididos en procesos tácticos y procesos operativos, detallados como se indica a continuación:

Procesos tácticos:

Se especifica el orden que llevaremos en el desarrollo del trabajo. Se harán reuniones tanto para discutir los temas del proyecto como para tratar temas concretos en la mejora de las tareas que se deberán desarrollar (reuniones de progreso). Conseguir llegar al acuerdo mutuo.

El tipo de aplicación que usare para la gestión de archivos será DropBox para poder acceder a los archivos desde los distintos terminales disponibles y de ser necesario poder compartirlos con otras personas

Procesos Operativos:

Todas las tareas recaen en el alumno, dejando al director la parte de las reuniones y ayuda a la parte del alumno. Así, habrá que repartir en trabajo en partes fáciles de desarrollar en periodos de tiempo cortos (2-3 semanas) para hacer una reunión después de cada parte, para ajustar así las posibles mejoras y los siguientes pasos a tomar que el director crea convenientes.

Desarrollaremos la aplicación siguiendo el Proceso Unificado caracterizado por estar dirigido por casos de uso, centrado en la arquitectura y por ser iterativo e incremental.

Dividiremos el proyecto en tres iteraciones a las cuales se definirán unos objetivos y un plazo para poder ir controlando los avances de la memoria, ya que la memoria se irá redactando a medida que se desarrolla el proyecto.

La primera parte será la que abarque el análisis de mercado, la elección y prueba tecnológica y la realización del DOP. También se pondrá en marcha la primera versión del juego, una beta simple con el menú y algún detalle más, para comprobar la redimensión para distintas resoluciones. La duración de esta iteración durará, aproximadamente un mes, desde Septiembre de 2012 a Octubre.

La segunda iteración será la más larga y densa. Engloba la Captura de Requisitos, Análisis, Diseño e Implementación. En lo que respecta al software, se generará casi toda la estructura del juego, es decir, la estructura del juego y la mayor parte de los mini juegos, también la pantalla de finalización de la partida. Esta iteración durará desde Octubre hasta mediados de Diciembre, más o menos.

La tercera iteración es la más corta. Se realizara la parte de pruebas, gestión y Conclusiones. En la aplicación se desarrollarán los menús de ayuda, las puntuaciones, el sistema de entrada/salida en archivo. La iteración durará desde Diciembre a Enero de 2013.

2.3 Alcance

2.3.1 Recursos disponibles

Para la realización del proyecto dispongo de; un ordenador con conexión a Internet, Eclipse como entorno de desarrollo integrado (IDE) al que instalaremos el complemento ADV, del que hablaremos más adelante. Un terminal smartphone Samsung Galaxy S II, con SO Android 4.1, y un tablet BQ Kepler, con Android 2.1, para realizar las pruebas de la aplicación.

2.3.2 Lista de secciones

Documentos de Objetivos del Proyecto: Desarrollo del documento en el que se recoge toda la información correspondiente a la gestión del proyecto y la manera en la que se abordará para conseguir los objetivos planteados al inicio de dicho proyecto.

Elección Tecnológica: Explicación de la selección de sistema operativo utilizado junto con todas las herramientas necesarias para la realización del juego.

Arquitectura: Estructura de las capas del programa.

Captura de Requisitos: Desarrollo del modelo de casos de uso y modelo de dominio del proyecto.

Análisis: Realización de los diagramas de secuencia del sistema y contratos de operaciones.

Diseño: Obtención de los diagramas de interacción y diagrama de clases.

Implementación: Forma de desarrollar la aplicación. Definición de las clases implementadas, la estructura utilizada, sus funcionalidades y demás.

Pruebas: Tras la implementación se realizaran distintas pruebas para comprobar el correcto funcionamiento de la aplicación.

Gestión: Distribución de las tareas y objetivos a lo largo de todo el proceso de creación del proyecto.

Conclusiones: Objetivos logrados, valoración personal y mejoras futras.

2.3.3 Lista de subtareas

ELEMENTO	NECESARIO
DOP	- Objetivos - Método de trabajo - Alcance - Planificación temporal - Plan de contingencia - Estimación de Costes - Factibilidad
Elección Tecnológica	- Android - Java - Eclipse - DropBox - Microsoft Office
Arquitectura	- Capas del sistema
Captura de Requisitos	- Captura de requisitos en el PUD - Modelo de casos de Uso - Modelo de Dominio
Análisis	- Diagrama de secuencia del sistema - Contratos - Modelo conceptual
Diseño	- Diagrama de interacción - Diagrama de clases
Implementación	- Organización del Codigo
Pruebas	- Fallos - Problemas
Gestión	- Análisis de lo ocurrido - Esfuerzo real vs planificado
Conclusiones	- Objetivos alcanzados - Valoración personal - Mejoras futuras

Ilustración 2: Tabla de Subtareas

2.3.4 Diagrama de Estructura de Descomposición del Trabajo (EDT)

Todos los procesos necesarios para este proyecto se recogen en el siguiente esquema, divididos entre procesos tácticos, operativos o elementos de formación.

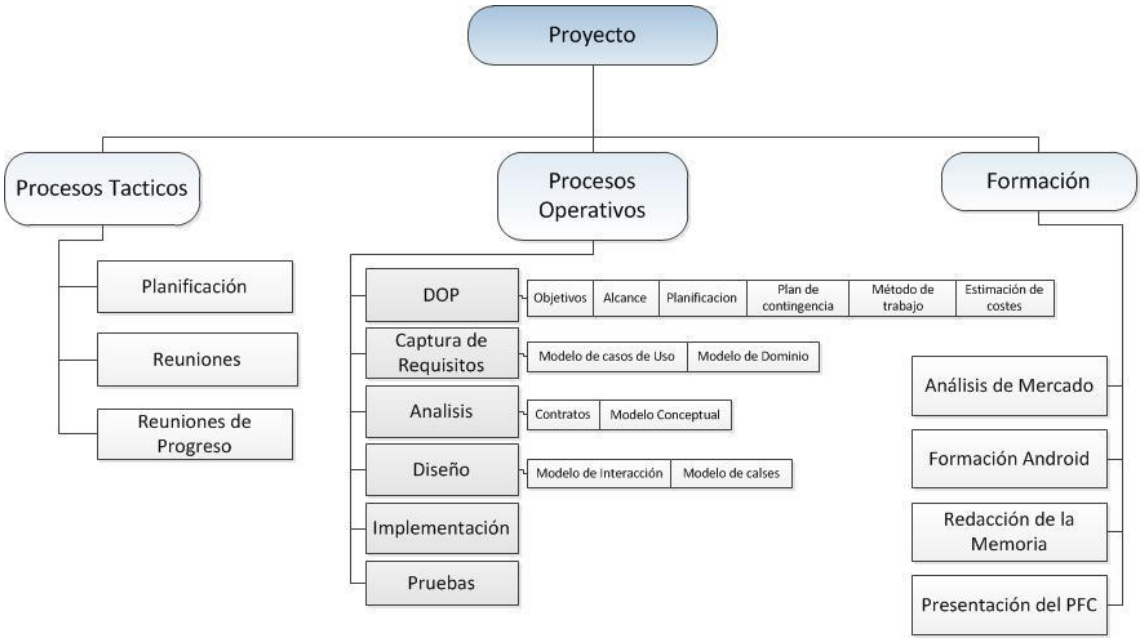


Ilustración 3: Diagrama EDT

2.4 Planificación Temporal

Para organizar la disposición temporal en el proyecto que nos ocupa, vamos a realizar un diagrama con el objetivo de poder presentar de una forma lo más visual posible dicha disposición, y para ello utilizaremos un tipo específico de gráfica: el diagrama de Gantt.

El diagrama de Gantt va a representar el esfuerzo empleado en cada apartado del proyecto a lo largo la duración de éste. Este esfuerzo es la suma de tiempo empleado para cada tarea. Se decide que el proyecto será entregado en la convocatoria de Enero 2013, ya que el alumno tiene el proyecto como única ocupación.

Tareas	Días
DOP	17
Captura de Requisitos	5
Análisis	8
Diseño	14
Implementación	50
Pruebas	10
Memoria	70
Formación	165

Ilustración 4: Duración de las tareas

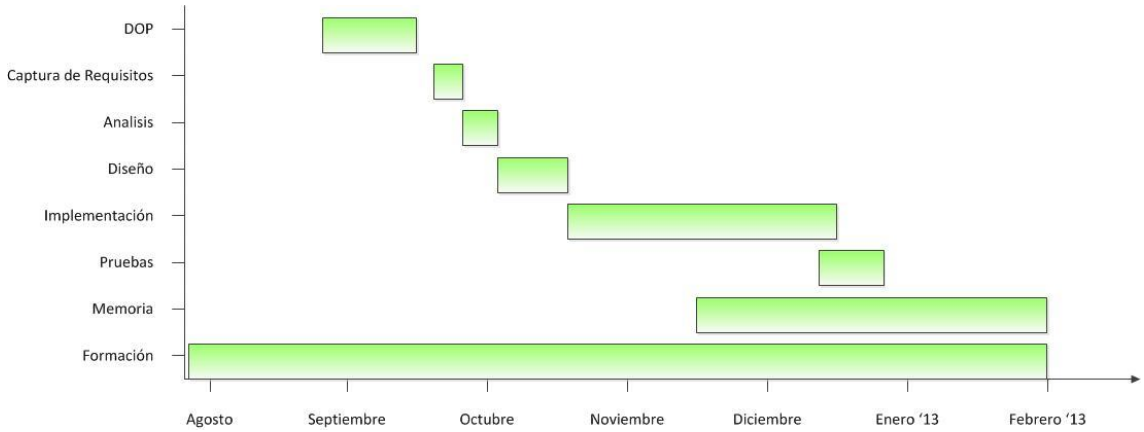


Ilustración 5: Diagrama de Gantt

Destaca la Formación por encima de las demás, ya que es una actividad que tiene lugar en todos los momentos del proyecto, porque en todo momento se necesitan aprender cosas nuevas.

2.5 Estimación de Costes

El tiempo dedicado será muy variable de una semana a otra dependiendo de las circunstancias (puentes, fiestas,…). Se calcula que se dedicara un promedio de 13 ó 15 horas por semana, teniendo 12 semanas para el desarrollo equivale a, entre 150 y 180 horas de trabajo. En la siguiente tabla una estimación de las horas empleadas por fase del proyecto.

Tareas	Horas
DOP	7
Captura de Requisitos	8
Análisis	13
Diseño	14
Implementación	65
Pruebas	5
Memoria	20
Formación	45

Ilustración 6: Tabla de Estimación de Costes

2.6 Plan de Contingencia

Con el fin de lograr que el proyecto se sustente y prevalezca ante cualquier adversidad mínimamente predecible hemos desarrollado un plan de contingencia para superar dichos obstáculos.

Descripción: El problema primordial es el contacto entre el director y el alumno. Normalmente se utilizara Internet como medio de comunicación. Si esto falla, se pondrán en contacto mediante teléfono para concertar las próximas reuniones, que se realizaran antes de empezar cada fase, también servirán para reevaluar nuestra dinámica de grupo.

- Gravedad: Media.
- Probabilidad: Baja.

Descripción: Uno de los riesgos más obvios es la perdida de información tanto de software como de la documentación, para evitarlo tomaremos la costumbre de guardar backups de la información almacenada hasta el momento en el disco duro y al menos en un pen drive.

- Gravedad: Alta.
- Probabilidad: Media.

.**Descripción**: En caso de baja de alguno de los miembros, si fuera el director se intentara seguir adelante con el proyecto sin su ayuda. En caso de que fuera el alumno se esperara hasta la vuelta en activo del alumno. Se ajustarán los tiempos del proyecto según sea necesario.

- Gravedad: Media.
- Probabilidad: Baja

Descripción: Existe también el riesgo de que durante el desarrollo del código partiendo de una base funcional y tras implementar unos cambios éste deje de funcionar. Para corregir esta situación emplearemos un sistema de versiones para el software que guardaremos en los backups correspondientes.

- Gravedad: Alta.
- Probabilidad: Media.

Descripción: El control del tiempo y los plazos de entrega pueden ocasionar problemas ya que la mayor parte de las soluciones implican un gasto de tiempo extra. En caso de ver que se va a terminar fuera de plazo, pediremos una prorroga, y si es necesario entregar nuestros progresos hasta el momento y la razón del atraso.

- Gravedad: Baja.
- Probabilidad: Baja.

La seguridad de la información es vital, así que para evitar filtraciones restringiremos el acceso al código y a la documentación mediante contraseña.

Como una guía resumiremos los problemas y soluciones en la siguiente tabla:

PROBLEMA	SOLUCION
Contacto Roto	Llamadas telefónicas y reuniones periodicas
Perdida de Información	Backups en diferentes sistemas de almacenamiento.
Baja del director	Continuar solo.
Baja del alumno	Se esperara hasta su puesta en activo.
Seguridad de la información	Contraseña para la información.
Codigo no funcional	Sistema de versiones
Problema en los plazos	Pedir prorroga y enviar los progresos.

Ilustración 7: Tabla de Plan de Contingencia.

2.7 Factibilidad

La dificultad del proyecto radica en la inexperiencia del alumno en programación sobre Android. La extensa documentación sobre el tema y la posibilidad de poder realizar más aplicaciones para este SO en un futuro añade motivación para afrontar en proyecto con ganas.

Se deduce que es factible la realización del proyecto.

3. Arquitectura del Sistema

Se establecerá un sistema de tres capas. Mediante estas capas el sistema debería ser capaz de gestionar toda la información de la aplicación.

La primera capa se encarga de mostrar la información al usuario y de leer los eventos que este produce al tocar o inclinar la pantalla. Esta capa también se encarga de las transiciones entre las pantallas y la actuación de los distintos elementos que en ellas se encuentran.

La segunda capa de encarga de el control de los eventos registrados en la capa anterior. Lo más reseñable es el uso de los eventos multitáctiles presentes en los sistemas Android 2.1 y posteriores. En esta capa se establece el núcleo del programa y su estructura de ejecución.

La tercera capa es la de Datos. Consiste en un simple fichero de almacenamiento de datos donde se guardaran las puntuaciones y los ajustes del juego, que se guardará cuando termine cada sesión de juego.

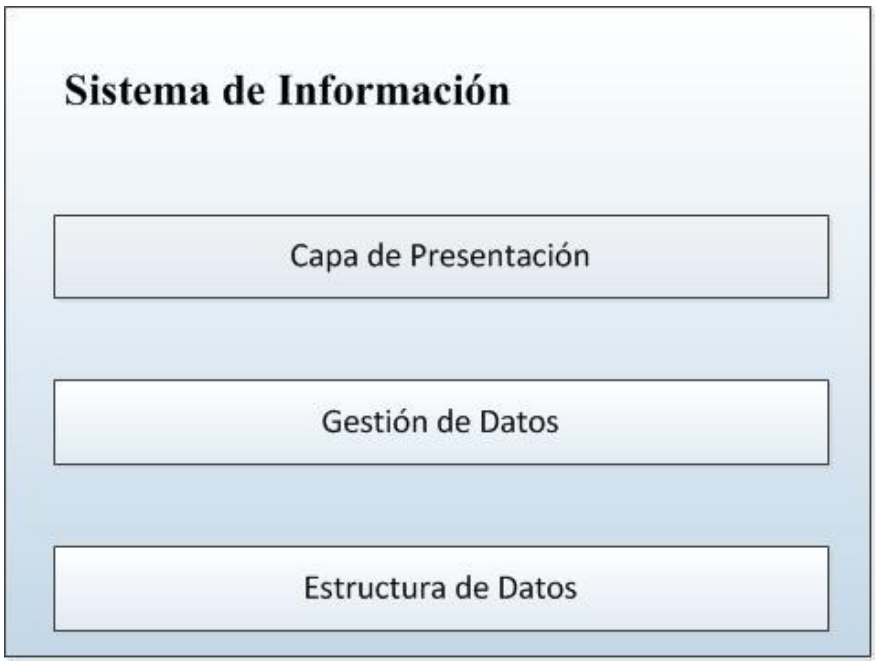


Ilustración 8: Sistema de Información.

4. Elección Tecnológica

4.1. Android

Una de las primeras decisiones que tuvimos que tomar fue para qué sistema iba a desarrollar la aplicación. Las opciones más claras eran iOS y Android, ya que estos dos sistemas están en alza y cada vez los usan más usuarios. De los dos, la primera opción fue iOS por que es un sistema más vistoso y fluido. Sin embargo, dado que no tenía ningún dispositivo Apple y ni si quiera un Mac (elemento indispensable para poder programar para iOS) este sistema se descartó rápidamente. La elección era clara, Android. Además este sistema se está abriendo mucho a distintas marcas de dispositivos y no se encierra en marcas propias de la casa. Eso fomenta a los usuarios a conocer mejor este sistema y a familiarizarse más con él, cosa que aumenta el número de clientes para aplicaciones de esa entidad.

4.1.1. ¿Qué es Android?

Android se ha convertido en uno de los sistemas operativos más usado para smartphones hoy en día. La primera vez que se publicó algo sobre Android fue en el año 2005, cuando Google compro una pequeña empresa llamada Android Inc. Era la apuesta que hacia este gigante para entrar en el mercado de los dispositivos móviles. En 2008, esta apuesta se convirtió en realidad y Google publicó la primera versión 1.0 de Android. Había nacido un nuevo competidor en el mercado de los dispositivos móviles. Desde entonces, continúa luchando con las plataformas asentadas como iOS (antes conocido como Phone OS) y BlackBerry.

Como Android en un sistema escrito en código abierto, los fabricantes de dispositivos móviles apenas se han encontrado trabas a la hora de recurrir a esta plataforma. Pueden fabricar dispositivos para todos los bolsillos y modificar Android para ajustarlo a la potencia del procesador. Es decir, Android no es un sistema exclusivo para los aparatos de alta gama, sino que se puede incorporar a los terminales más económicos. Esto le permite llegar a mucha más gente.

El propio Android es un sistema operativo y una plataforma móvil basada en la versión 2.6 del kernel de Linux y se puede usar gratuitamente tanto con fines comerciales como particulares. Muchos miembros de la OHA (Open Handset Alliance) desarrollan sus propias versiones de Android personalizadas para sus dispositivos, HTC Sense de HTC, MOTOBLUR de Motorola, etc… a estas adaptaciones de las conoce como mods, firmware o ROM

4.1.2. Android SDK

Para desarrollar aplicaciones para Android tendremos que emplear el kit para el desarrollo de software (SDK). El SDK se compone de varias herramientas, documentación, tutoriales y ejemplos que nos ayudaran en nuestro trabajo como desarrolladores. También se incluyen las librerías de Java necesarias para generar las aplicaciones para Android. Estas contienen las API del framework para aplicaciones. La mayoría de los sistemas operativos de escritorio son compatibles y se toman como entornos de desarrollo.

El kit SDK se puede integrar en Eclipse, un entorno de desarrollo integrado en Java (IDE) y realizado en código abierto que cuenta con muchas propiedades y una gran popularidad. La integración se establece a través de las herramientas para desarrolladores de Android (ADT) que añaden un nuevo conjunto de capacidades a Eclipse para originar proyectos Android; ejecutar, perfilar y depurar las aplicaciones en el emulador o en el dispositivo y empaquetar las aplicaciones para distribuirlas en Android Market. El kit SDK también se puede integrar en otros entornos IDE como NetBeans, aunque no hay ningún soporte oficial para ello.

4.1.3. API

Una API es una interfaz que nos proporciona comunicación entre dos partes del software. En este caso entre una aplicación desarrollada por nosotros y las bibliotecas del sistema operativo. Google ofrece una API por cada versión de Android disponible. Cada API permite desarrollar aplicaciones para esa versión y sus versiones posteriores. Son gratuitas y de código abierto. La API permite el desarrollo de aplicaciones para el lenguaje java y para el lenguaje C++.

4.1.4. Versiones de Android

Desde su aparición en 2008, Android ha recibido siete actualizaciones. Todas ellas tienen nombre de postres, a excepción de Android 1.1, que hoy en día es irrelevante. Cada versión a añadido una nueva funcionalidad a la plataforma Android que han representado importantes avances para los desarrolladores de juegos. Por ejemplo la versión 1.5 (Cupcake) admitía las librerías nativas de las aplicaciones Android, algo que hasta la fecha solo se podía hacer si se escribía en Java. La versión 1.6 (Donut) presento la capacidad de trabajar con diferentes resoluciones de pantalla. Con la versión 2.0 (Éclair) llego la posibilidad de trabajar con dispositivos multitáctiles. La versión 2.2 (Froyo) añadió compilación en tiempo real (JIT) a la maquina virtual Dalvik, que permitió utilizar todas las aplicaciones Java en Android.

4.1.5. Tipos de Dispositivos

Android no pertenece a un único ecosistema. Muchos fabricantes se han unido al tren de Android, como Samsung, HTC y Motorola y ofrecen una variedad de dispositivos que trabajan con él. Además de móviles, poco a poco están llegando unos nuevos dispositivos, las tabletas, que también trabajan con Android. Pero el hecho es que todos comparten los conceptos básicos de Android, con lo que nuestra vida como desarrolladores será más fácil.

4.2 Java

Para desarrollar aplicaciones Android existen dos lenguajes de programación: java y C++. Elegimos el primero porque es el más familiar para el desarrollador y porque es la más elegida por la mayoría de programadores Android. A parte de que la documentación utilizada para comprender Android esta basada en Java y los autores recomiendan Java antes que C++.

4.3 Eclipse

El SDK de Google está diseñado para este IDE. Y toda su documentación así como tutoriales y ejemplos están todos realizados bajo este software. No hacía falta dificultar el trabajo más de lo necesario y además el alumno había trabajo anteriormente con este IDE.

4.4 DropBox

Este sistema de almacenamiento fue escogido para poder probar sin problema mi aplicación en cualquiera de los dispositivos disponibles. Esta herramienta es de fácil uso, e instalación en ordenadores, smartphones o tabletas.

4.5 Microsoft Office

La herramienta de ofimática elegida fue Microsoft office, ya que esta herramienta es la más utilizada desde hace varios años por el alumno. Además de que este conjunto de programas ofrece la herramienta necesaria para crear los diagramas necesarios para esta memoria, Microsoft Visio.

4.6 PhotoShop CS4

Para crear todo el diseño grafico que utilizo PhotoShop. La razón fue que esta herramienta ya era conocida por el alumno y la tenía a su disposición. Además, es uno de los mejores software de edición de imagen que existen en el mercado.

5. Captura de Requisitos

Se desarrolla una aplicación que ponga a prueba los reflejos del jugador. El juego comenzará con un menú de inicio como muestra la Ilustración 9, en el que se pueden elegir entre varias opciones; Jugar, Puntuaciones, Ayuda y Ajustes, que será un pequeño botón en la esquina inferior derecha de la pantalla, para conseguir un menú menos cargado.

Ilustración 9: Menú del juego.

Cada una de estas opciones representara un caso de Uso concreto. Pero no serán los únicos, dado que dentro del juego se desarrollaran 5 "mini juegos" a los que se accederá a través de la pantalla de juego pulsando en las zonas de los mecánicos que estén resaltados de color naranja, como los de la Ilustración 10. Hay que decir, que el orden de los mini juegos no es el mismo en cada partida, se eligen el orden aleatorio cada vez que se empieza un nuevo juego.

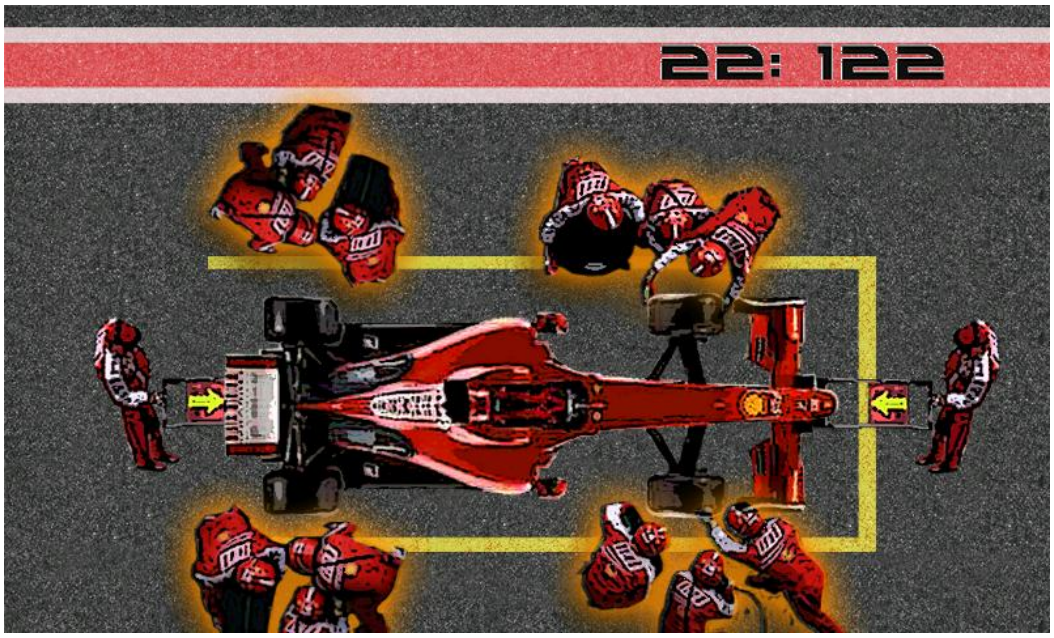


Ilustración 10: Elección de mini juego.

Con la aplicación solo jugara una persona a la vez, así que solo habrá un actor, al que a partir de ahora llamaremos Jugador.

5.1 Casos de Uso del Jugador

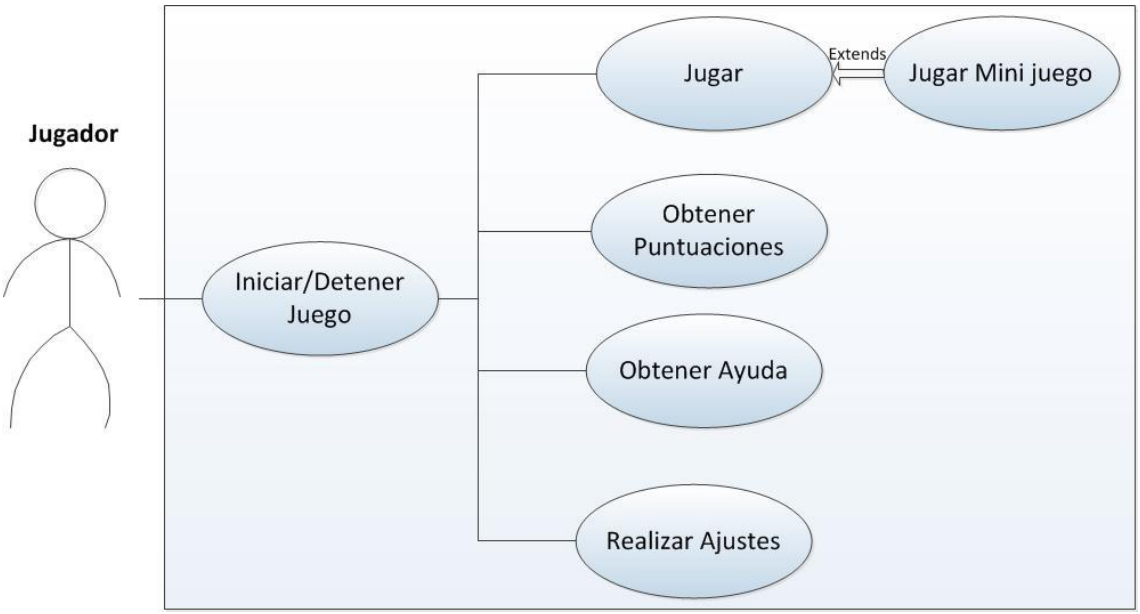


Ilustración 11: Casos de Uso del Jugador

5.1.1. Caso de Uso Jugar

Actores: Usuario

Descripción: El usuario puede elegir la opción de Jugar en el menú principal para inmediatamente empezar una partida. Al hacerlo comenzara automáticamente un primer mini juego obligatorio para poder seguir adelante, después se podrá elegir entre distintas zonas de la pantalla (representadas por mecánicos) para poner en marcha alguna de los demás mini juegos que queden por jugar.

Cuando el usuario acabe todos los mini juegos, se le guardara la puntuación si ha quedado entre los cinco primeros, y de mostrara un menú con las opciones de volver a jugar, o regresar al menú principal.

5.1.2. Caso de Uso Obtener Puntuaciones

Actores: Usuario

Descripción: El usuario pulsa en la opción de Puntuaciones de la pantalla principal para hacer aparecer una pantalla con los cinco mejores tiempos logrados en el juego y un botón para poder volver al menú principal.

5.1.3. Caso de Uso Obtener Ayuda

Actores: Usuario

Descripción: El usuario toca la opción de Ayuda del menú principal para entrar en este caso de uso. En él se explicara el objetivo del juego, así como las reglas y objetivos de los otros cuatro mini juegos. El usuario podrá avanzar en las explicaciones o volver atrás por si acaso no ha entendido cualquier cosa que se le explico anteriormente.

5.1.4 Caso de Uso Realizar Ajustes

Actores: Usuario

Descripción: Pulsando en el boton de ajustes se desplegara el menú de ajustes, en el que el usuario podrá elegir el idioma de toda la aplicación, eligiendo entre español e inglés. También tendrá la oportunidad de desactivar o activar el sonido

5.1.5 Caso de Uso Jugar Mini Juego

Actores: Usuario

Descripción: Después de seleccionar a alguno de los mecánicos comenzara inmediatamente alguna de las pruebas que restan por superar. Cuando los objetivos de la prueba se realicen esta se dará por terminada y de volverá a la pantalla de juego.

5.1.6 Caso de Uso Iniciar/Detener Juego

Actores: Usuario

Descripción: Cuando el usuario pulse el icono de juego, este se iniciara, cargando todas las imágenes que se utilizaran más adelante. También se leerán desde archivo todas las puntuaciones máximas y los datos de los ajustes, idioma (En o Sp) y sonido. Cuando el usuario decida salir del juego los nuevos datos de las puntuaciones y los ajustes se escribirán en un archivo creado en la tarjeta del dispositivo.

5.2 Modelo de Dominio

El juego se divide en cuatro partes destacables.

La primera es el juego en sí mismo, en él se guardará el tiempo transcurrido y el número de pruebas que están resueltas hasta el momento. Este juego consta de cinco mini juegos asociados a él. Cada uno de estos mini juegos guarda el tiempo transcurrido en el mini juego, y si la prueba ha terminado o no. Cuando la prueba haya terminado, se volverá al juego para seguir la partida.

La segunda parte son las puntuaciones. En esta pantalla se mostrarán las mejores puntuaciones realizadas en el juego durante cualquier momento desde su instalación. Cada puntuación guardará un tiempo, que será el utilizado para establecer el orden de las puntuaciones.

La tercera parte es la Ayuda. Mostrará los objetivos del juego o como se realizan cada uno de los mini juegos. Esta clase guardara un índice que será utilizado para saber en qué pantalla de la ayuda nos encontramos. Si avanzamos el índice aumentará y si retrocedemos disminuirá. Cambiando con cada índice, la explicación de cada mini juego.

La cuarta y última parte es la de los ajustes. Esta mostrara los ajustes que están seleccionados en el momento, el idioma elegido y si el sonido esta activado o no, y

dará opciones para cambiarlo. Esta clase guarda un booleano para saber el estado del sonido, y el nombre del idioma seleccionado.

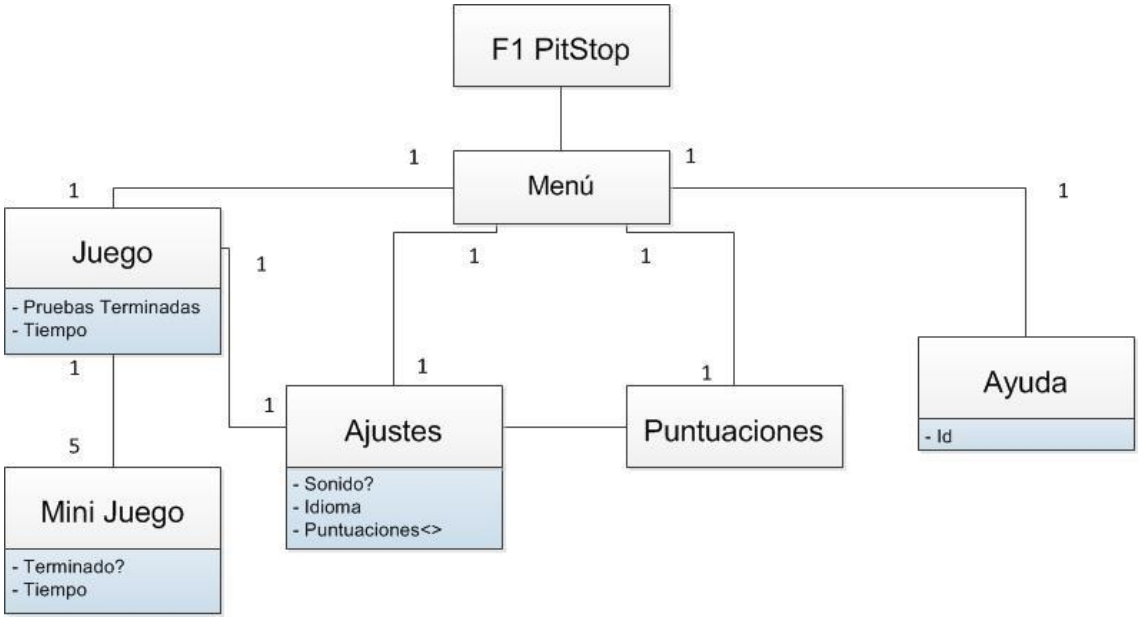


Ilustración 12: Modelo de Dominio.

6. Análisis

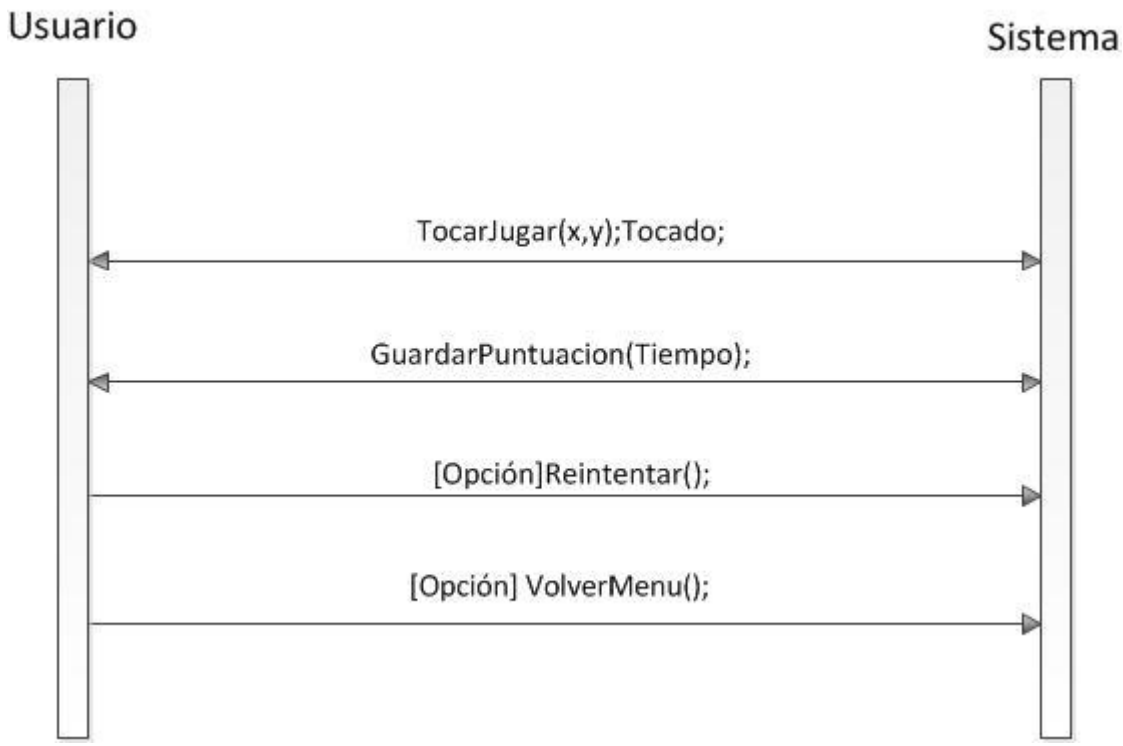
En este capítulo realizaremos un análisis más a fondo de los casos de uso descritos en el Capitulo 5.

Mediante diagramas de secuencia mostraremos los eventos generados por el actor, su orden, y las respuestas generadas por el sistema.

Una vez determinados todos los eventos que tendrán lugar en los distintos casos de uso se explicaran brevemente mediante los contratos. Especificaremos los cambios realizados por el sistema al igual que la información que este muestra.

6.1. Caso de uso Jugar

6.1.1. Diagrama de secuencia del sistema



Ilustracion 13: Diagrama de secuencia del caso de uso Jugar

6.1.2. Contratos

Nombre: TocarJugar

Responsabilidades: Se encarga de controlar si el usuario a pulsado en el botón de Jugar, y de haberlo hecho, empezará la partida.

Precondición: El evento del toque del dedo debe de ser del tipo UP (cuando el dedo se despegue de la pantalla)

Postcondicion:

Salida: - Tocado: Es un boolean que indica si la pulsación del jugador está entre las coordenadas (x,y) que se han pasado como parámetro.

Nombre: GuardarPuntuacion

Responsabilidades: Se encarga de que, una vez terminado el juego, guardar la puntuación en el lugar correspondiente en la lista de puntuaciones.

Precondición: El juego ha terminado.

Postcondicion: Las puntuaciones quedarán actualizadas en el orden adecuado.

Salida:

Nombre: Reintentar

Responsabilidades: Se encarga de que, una vez terminado el juego, puedas volver a jugar sin necesidad de volver al menú principal

Precondición: El juego ha terminado.

Postcondicion: El tiempo se reinicia, y el contador de las pruebas realizadas vuelve a 0.

Salida:

Nombre: VolverMenu

Responsabilidades: SE encarga de que, una vez terminado el juego, puedas vovler al menú principal para elegir entre las distintas opciones que ofrece el juego.

Precondición: El juego ha terminado.

Postcondicion: El tiempo se reinicia, y el contador de las pruebas realizadas vuelve a 0.

Salida:

6.2. Caso de uso Jugar mini juego

6.2.1. Diagrama de secuencia del sistema

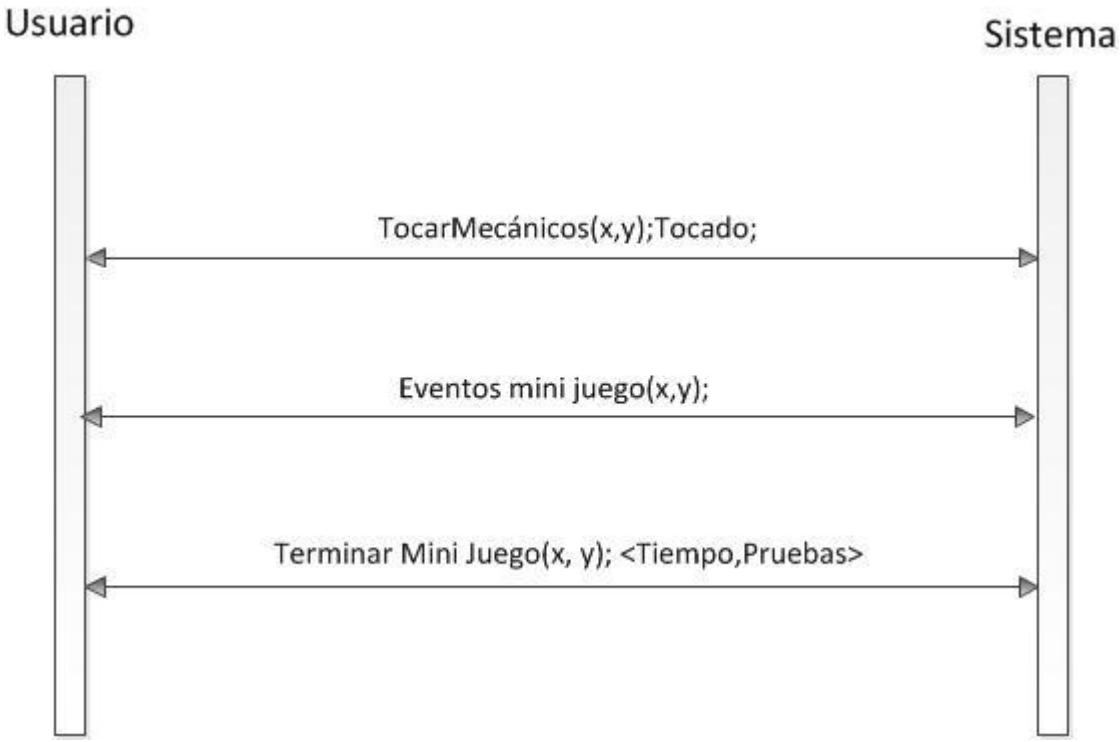


Ilustración 14: Diagrama de secuencia del caso de uso Jugar mini juego

6.2.2. Contratos

Nombre: TocarMecanicos

Responsabilidades: Se encarga se controlar que el evento de toque en la pantalla, toque al menos a uno de los mecánicos en activo.

Precondición: El movimiento debe de ser del tipo UP.

Postcondición:

Salida: Tocados: Es un boolean que será cierto si el jugador a tocado dentro de las zonas deseadas.

Nombre: Eventos mini juegos

Responsabilidades: Se encarga de controlar que los toques en la pantalla sean los oportunos para cada uno de los distintos mini juegos.

Precondición: Hay un mini juego iniciado.

Postcondición: Actualiza el estado de la variable "Terminado" acorde con los eventos producidos.

Salida:

Nombre: Terminar mini juego

Responsabilidades: Cuando se complete el último de los eventos de toque del mini juego, este será responsable de volver al juego, pasando el tiempo tardado en completarlo y aumentando el número de pruebas completadas.

Precondición: Debe ser el último movimiento del mini juego

Postcondición:

Salida:

- Tiempo: Es el tiempo tardado en realizar el mini juego.
- Pruebas: Es el número total de pruebas realizadas hasta la fecha.

6.3. Caso de uso Obtener Puntuaciones

6.3.1. Diagrama de secuencia del sistema

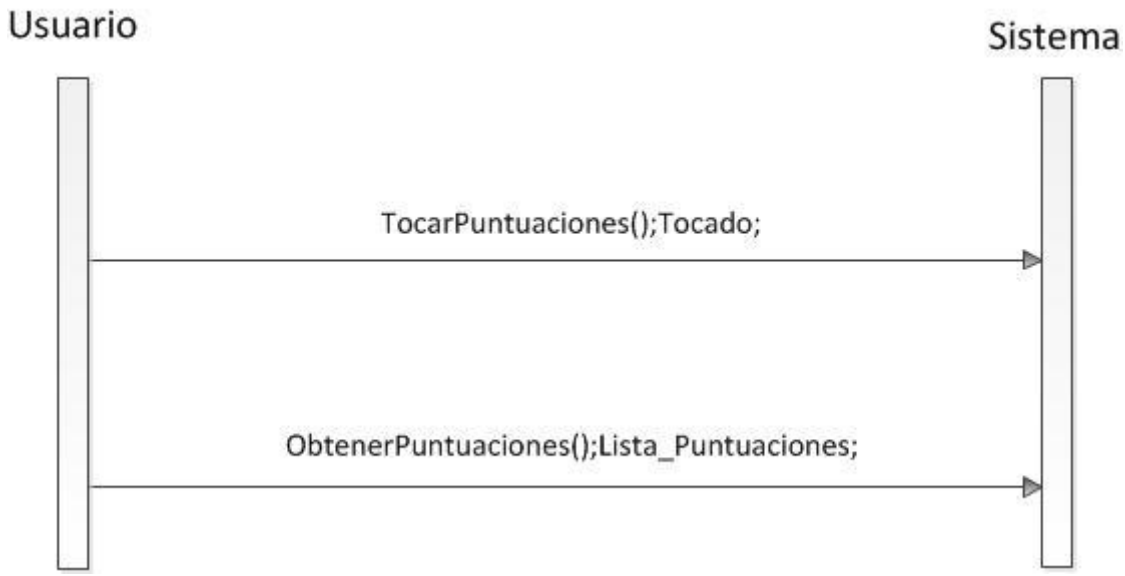


Ilustración 15: Diagrama de secuencia del caso de uso Obtener Puntuaciones

6.3.2. Contratos

Nombre: Tocar Puntuaciones

Responsabilidades: Se encarga de saber si el usuario a tocado en la palabra puntuaciones, para enviarle a la pantalla de puntuaciones.

Precondición: EL movimiento del dedo debe de ser del tipo UP

Postcondición:

Salida: Tocados: Es un boolean que será cierto si se han tocado en la palabra Puntuaciones.

Nombre: Obtener Puntuaciones

Responsabilidades: Se encarga de cargar las puntuaciones para mostrarlas más tarde en pantalla.

Precondición: Hay cinco puntuaciones cargadas de antemano al inicio del juego.

Postcondición: Se cargan las cinco mayores puntuaciones.

Salida:

6.4. Caso de uso Obtener Ayuda

6.4.1. Diagrama de secuencia del sistema

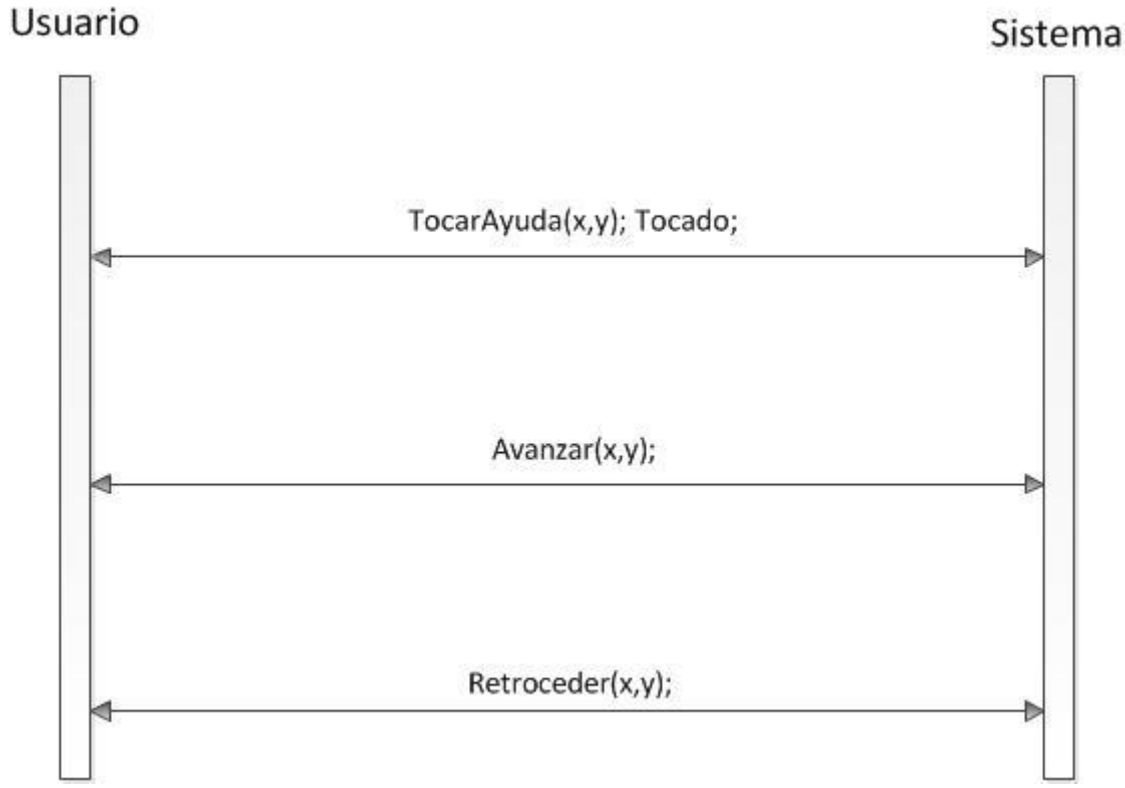


Ilustración 16: Diagrama de secuencia del caso de uso Obtener Ayuda

6.4.2. Contratos

Nombre: Tocar Ayuda

Responsabilidades: Se encarga de saber si el usuario a tocado en la palabra ayuda del menú principal

Precondición: El movimiento del dedo tiene que ser del tipo UP

Postcondición:

Salida: Tocados: Es un boolean que será cierto si se han tocado en la palabra Ayuda

Nombre: Avanzar

Responsabilidades: Se encarga de avanzar en las pantallas de ayuda.

Precondición: Debes de estar dentro de la pantalla de ayuda.

Postcondición: Muestra la siguiente pantalla de ayuda.

Salida:

Nombre: Retroceder

Responsabilidades: Se encarga de retroceder en las pantallas de ayuda.

Precondición: Debes de estar dentro de la pantalla de ayuda.

Postcondición: Muestra la anterior pantalla de ayuda.

Salida:

6.5. Caso de uso Realizar Ajustes

6.5.1. Diagrama de secuencia del sistema

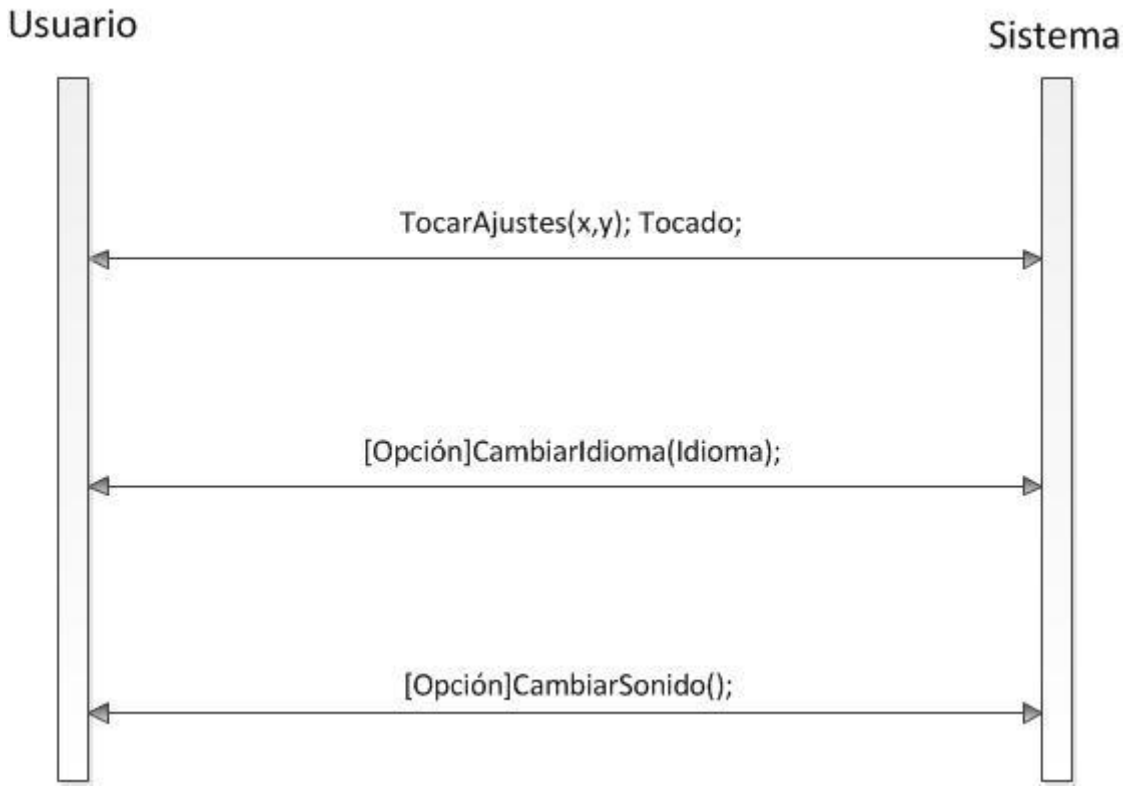


Ilustración 17: Diagrama de secuencia del caso de uso Realizar Ajustes

6.5.2. Contratos

Nombre: Tocar Ajustes

Responsabilidades: Se encarga de saber si el usuario ha pulsado en el botón de ajustes del menú principal.

Precondición: El movimiento del dedo debe de ser del tipo UP.

Postcondición:

Salida: Tocados: Es un boolean que será cierto si se han tocado en el botón de Ajustes

Nombre: Cambiar Idioma

Responsabilidades: Se encarga de cambiar el idioma del juego.

Precondición: Tienes que pulsar dentro del campo de uno de los idiomas del juego

Postcondición: Se cambia el idioma de todo el juego.

Salida:

Nombre: Cambiar Sonido

Responsabilidades: Se encarga de activar o desactivar el sonido del juego.

Precondición:

Postcondición: Invierte el estado del sonido, si esta activado lo desactiva y viceversa.

Salida:

6.6. Iniciar/Detener Juego

6.6.1. Diagrama de secuencia del sistema

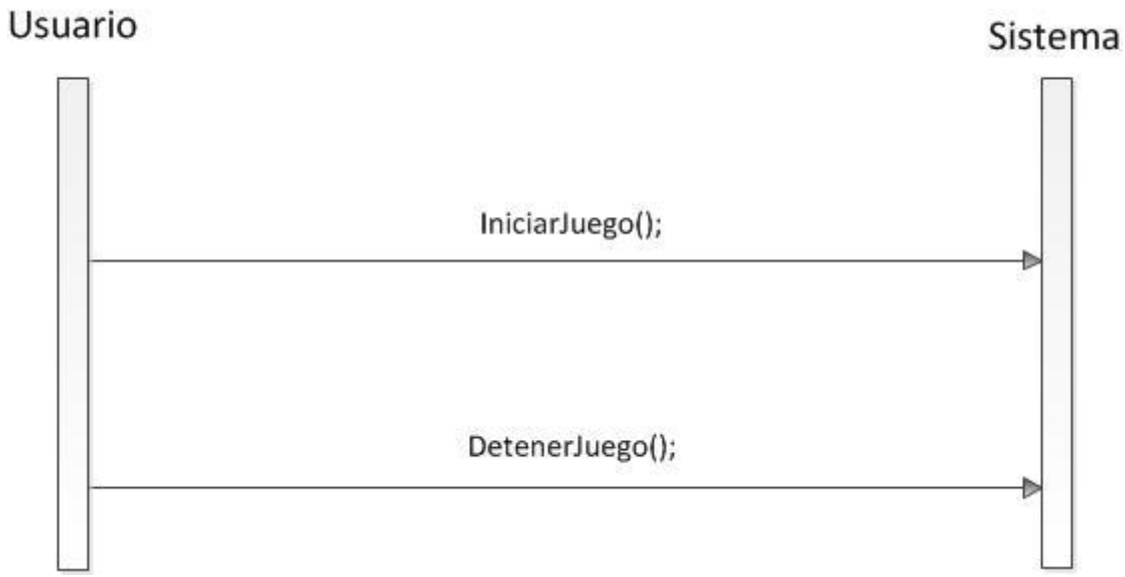


Ilustración 18: Diagrama de secuencia del caso de uso Iniciar/Detener Juego

6.6.2. Contratos

Nombre: Iniciar Juego

Responsabilidades: Se encarga de que, en el momento del encendido del juego, carga todas las imágenes, puntuaciones y ajustes del juego, para su posterior manupulacion.

Precondición: El juego está apagado

Postcondición: Tanto todas las imágenes usadas en el juego, como las puntuaciones y los ajustes son cargados.

Salida:

Nombre: Detener Juego

Responsabilidades: Se encarga de guardar de nuevo todas las puntuaciones y los ajustes en el archivo indicado, para así poder leerlas en la próxima apertura del juego.

Precondición: El juego esta encendido

Postcondición: Todas las puntuaciones y los ajustes de idioma y sonido quedan guardados en archivo.

Salida:

7. Diseño

Se intentará obtener un diseño que conserve el encapsulamiento de los objetos para conseguir un bajo acoplamiento. Las clases serán lo más sencillas posibles para un fácil mantenimiento y mayor facilidad de comprensión y reduciremos el tiempo para encontrar problemas. Se describirán los diseños y se definirán los diagramas de secuencia de todas las funciones.

7.1. Caso de Uso Jugar

7.1.1. Contrato: TocarJugar

Es una de las acciones principales, la cual se necesita para seguir con el desarrollo del juego, mediante un patrón experto, evaluamos la pulsación del usuario en la pantalla. La clase menú se encarga de saber si las coordenadas de la pulsación están dentro del rango, y si esta, crea un juego con los parámetros iniciales.

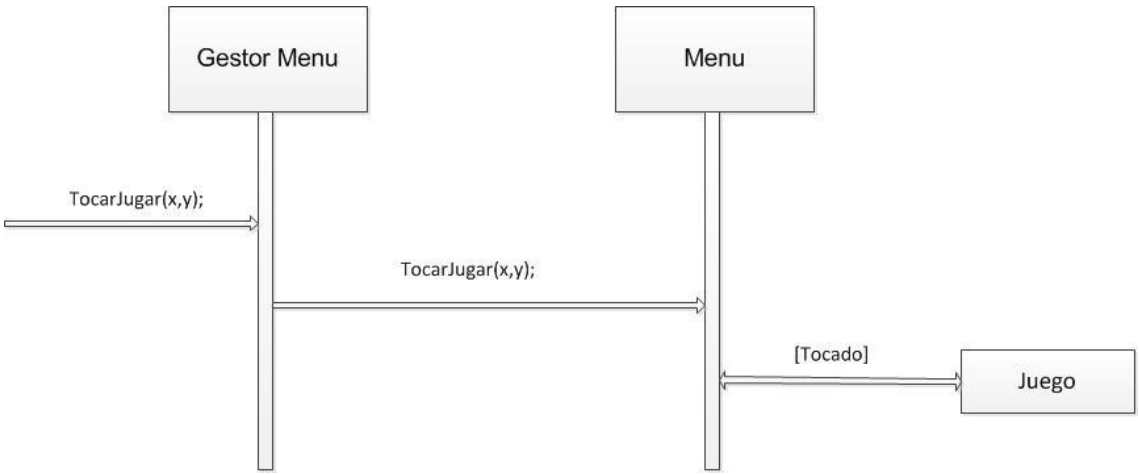


Ilustración 19: Diagrama de secuencia del contrato TocarJugar

7.1.2. Contrato: Guardar Puntuación

Cuando el juego este terminado, la clase juego se encarga de actualizar la lista de puntuaciones con el nuevo tiempo conseguido. La lista se guarda en la clase Ajustes para que después se pueda guardar en archivo junto con el resto de datos.

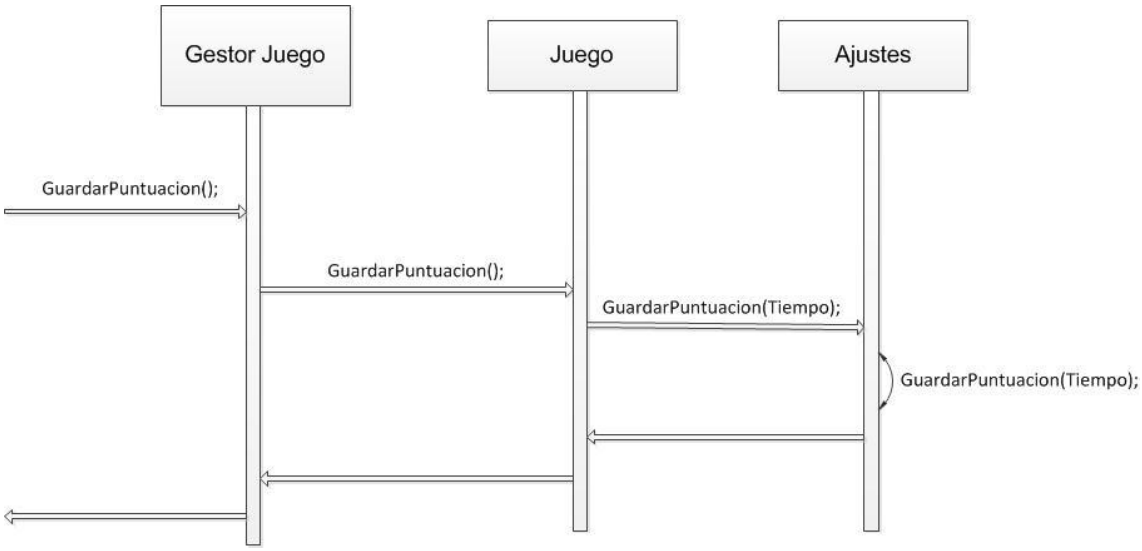


Ilustración 20: Diagrama de secuencia del contrato GuardarPuntuación

7.1.3. Contrato: Reintentar

Cuando el juego termina y las puntuaciones se hayan actualizado, si se ha tocado dentro de la zona determinada, se creara un juego nuevo, exactamente como en el contrato TocarJugar, con los atributos iniciales.

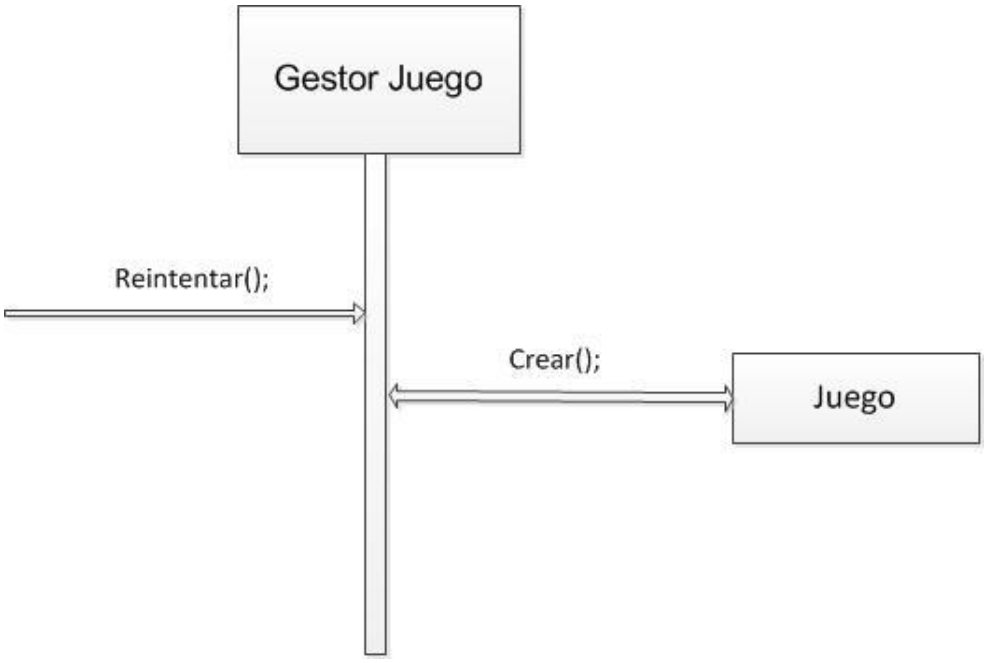


Ilustración 21: Diagrama de secuencia del contrato Reintentar

7.1.4. Contrato: VolverMenu

Cuando el juego termine y se hayan actualizado las puntuaciones, si se ha tocado en el área establecida, el juego creara de nuevo el menú principal, para poder acceder a todas las partes del juego de nuevo, poder cambiar ajustes, y demás.

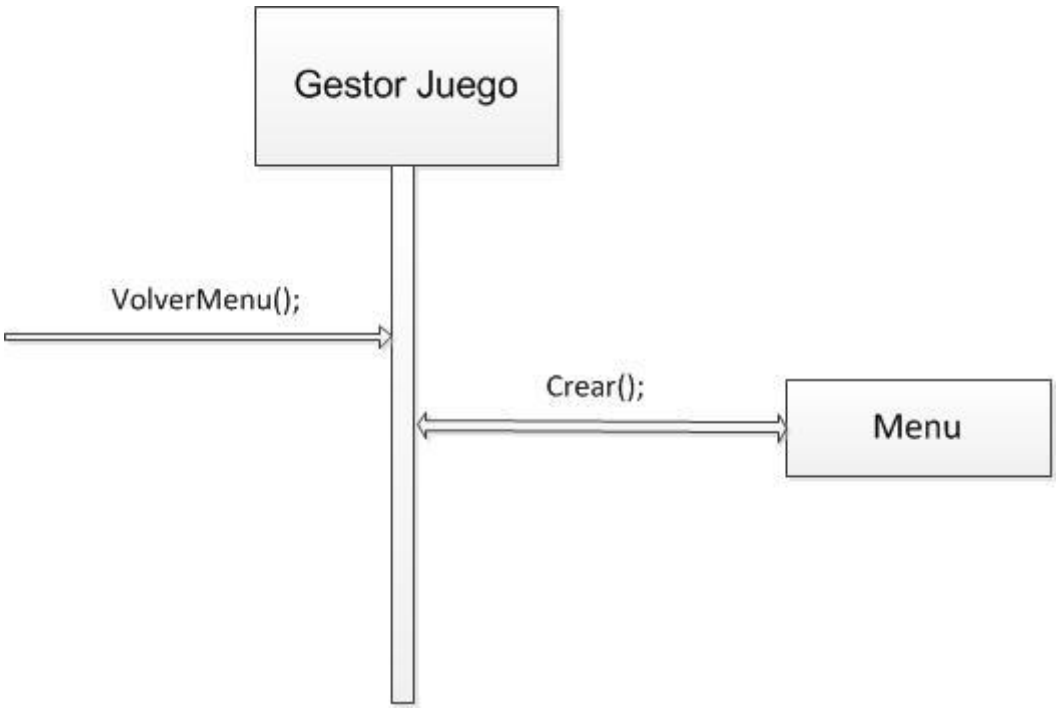


Ilustración 22: Diagrama de secuencia del contrato VolverMenu

7.2. Caso de Uso Jugar Mini juego

7.2.1. Contrato: TocarMecanico

Cuando el juego haya empezado, después de haber pasado la primera prueba, si tocas en alguna de las áreas activadas, se creara un mini juego nuevo de la lista de los mini juegos que falten por jugar.

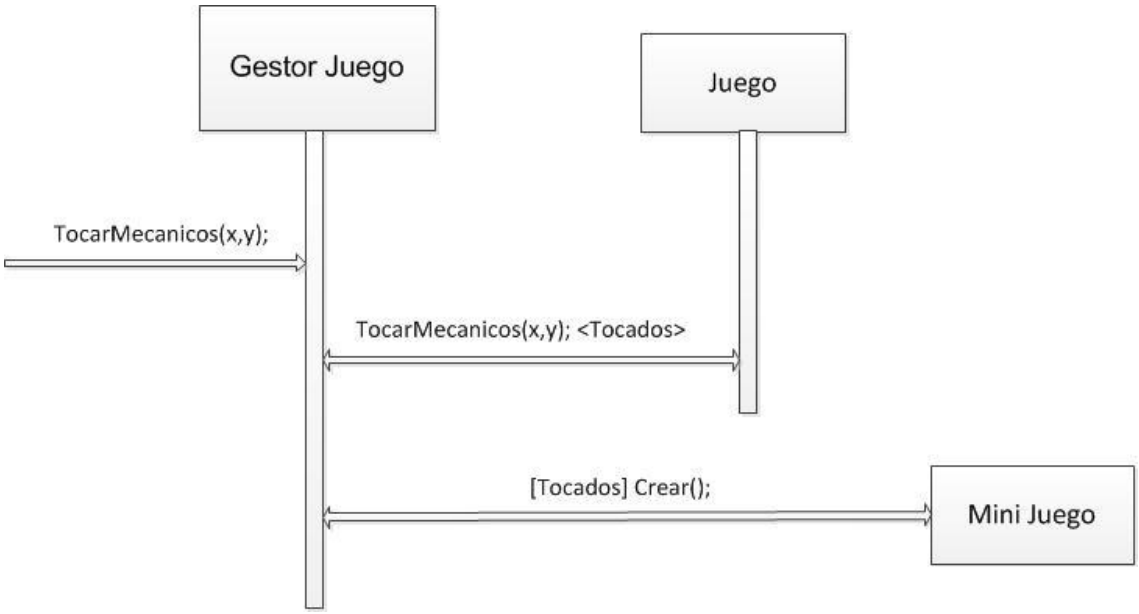


Ilustración 23: Diagrama de secuencia del contrato TocarMecanico

7.2.2. Contrato: EventosMiniJuego

Cuando se ha creado un mini juego, se testearan los movimientos del dedo para saber si son los necesarios para ir completando el mini juego. El mini juego se actualizara a cada frame acorde con los movimientos del los dedos

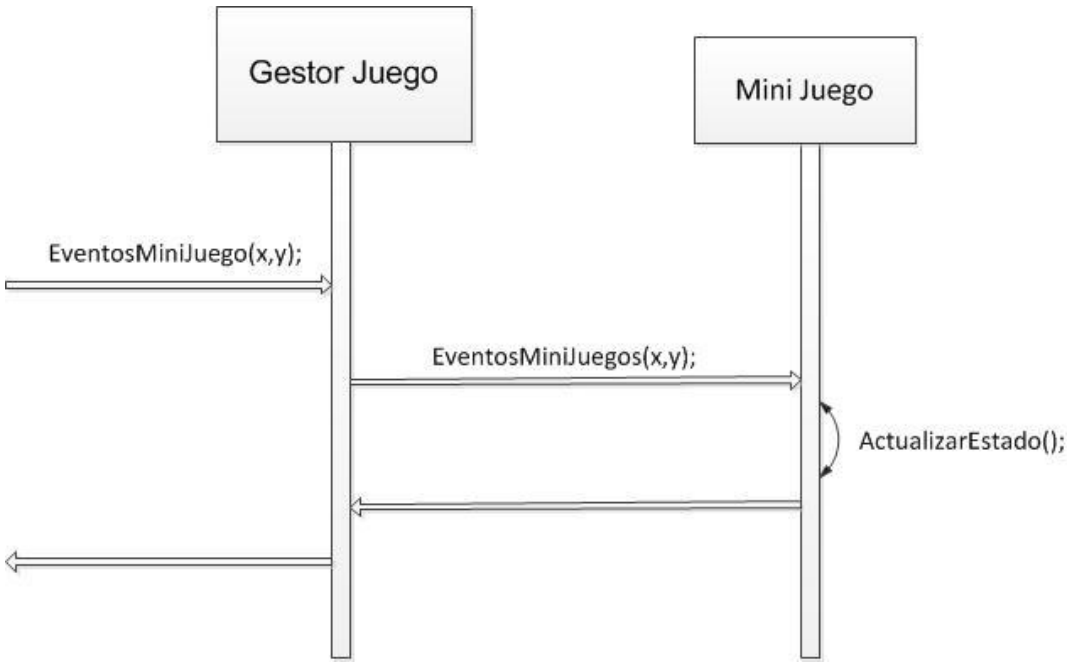


Ilustración 24: Diagrama de secuencia del contrato EventosMiniJuego

7.2.3. Contrato: TerminarMiniJuego

Cuando el juego se haya terminado, el mini juego creara de nuevo el juego principal, pasándole siempre los parámetros de las pruebas superadas hasta el momento y el tiempo utilizado.

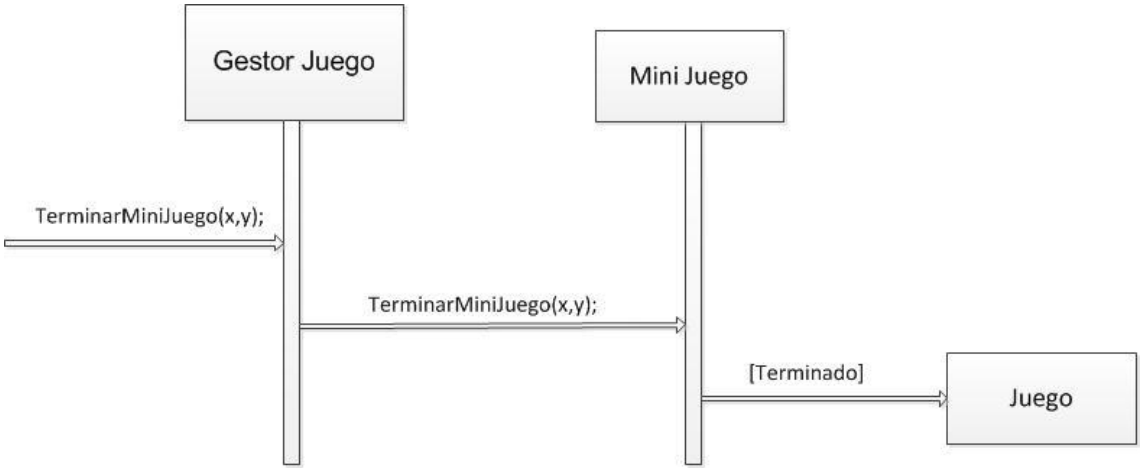


Ilustración 25: Diagrama de secuencia del contrato TerminarMiniJuego

7.3. Caso de Uso Obtener Puntuaciones

7.3.1. Contrato: TocarPuntuaciones

Es otro de los eventos principales del menú, si los eventos de las pulsaciones del usuario están en la zona indicada, creara la pantalla de las puntuaciones.

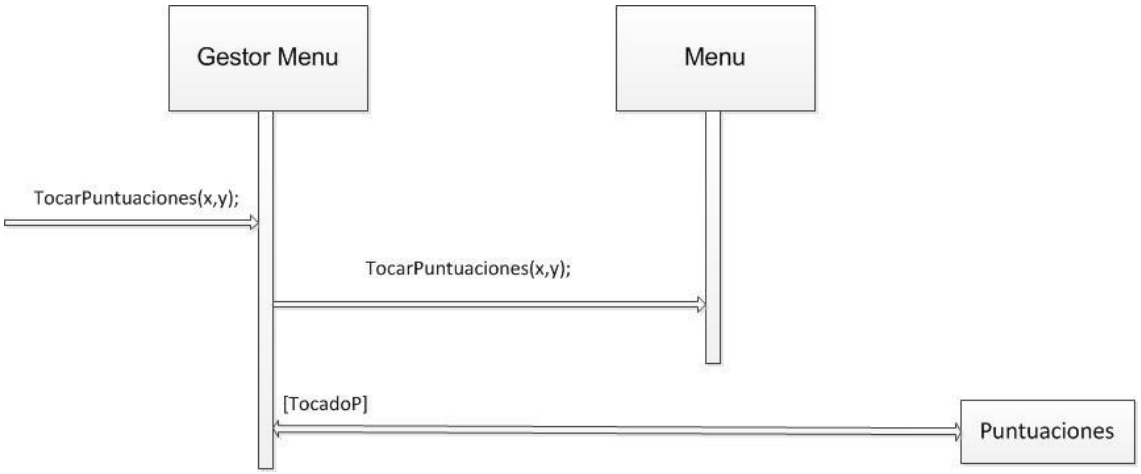


Ilustración 26: Diagrama de secuencia del contrato TocarPuntuaciones

7.3.2. Contrato: ObtenerPuntuaciones

Una vez iniciada la pantalla de puntuaciones, desde esta clase se accederá a las listas de las puntuaciones que están guardada en la clase Ajustes para su guardado en fichero.

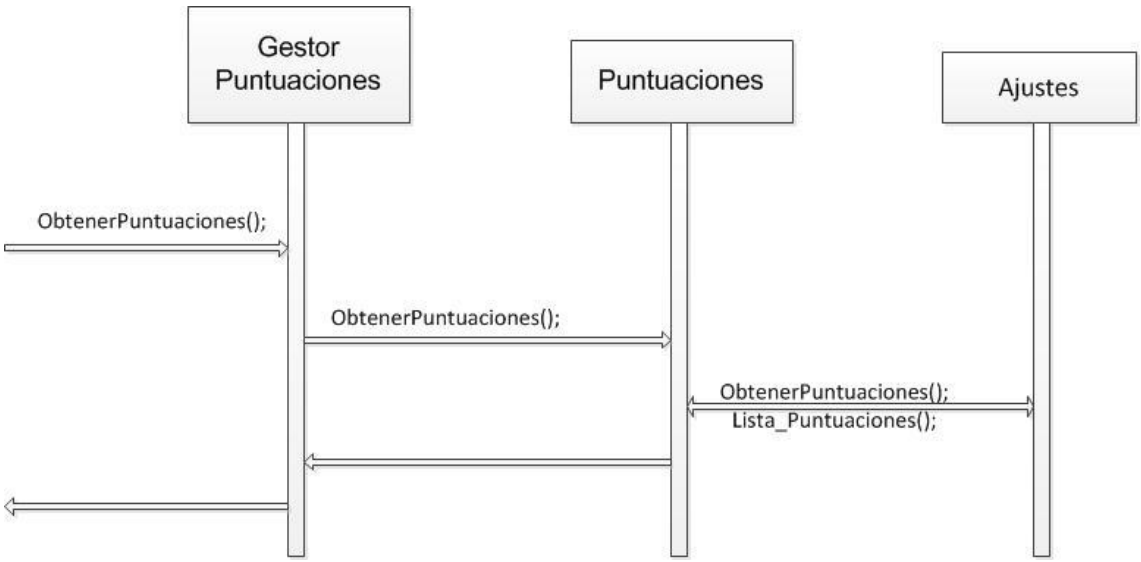


Ilustración 27: Diagrama de secuencia del contrato ObtenerPuntuaciones

7.4. Caso de Uso Obtener Ayuda

7.4.1. Contrato: TocarAyuda

Es otro caso del menú principal. Si el usuario toca en la zona de la pantalla indicada, se creara el menú de ayuda

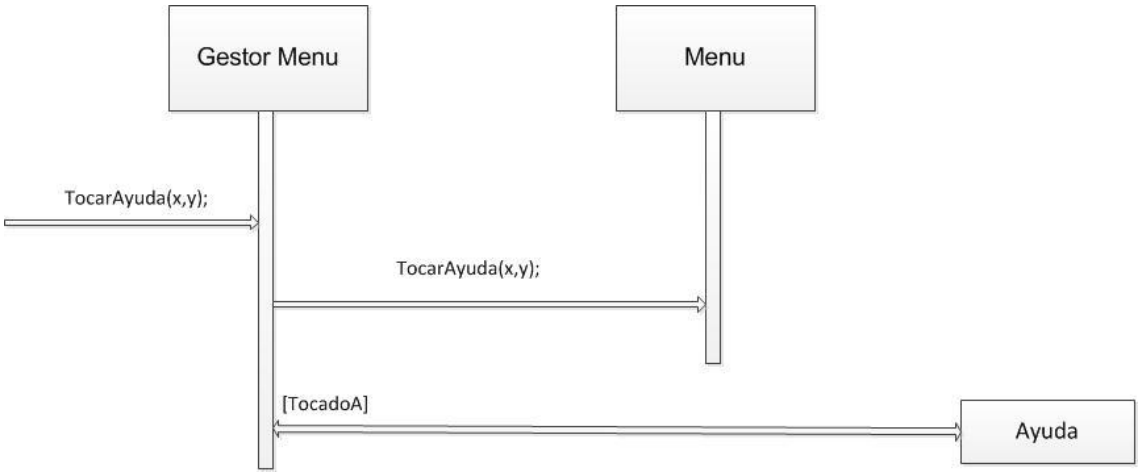


Ilustración 28: Diagrama de secuencia del contrato TocarAyuda

7.4.2. Contrato: Avanzar

Cuando se pulse en la zona marcada del botón de avanze, se aumentara en 1 el Id que marca la pagina que se muestra en el menú de ayuda, hasta un máximo de 7 páginas, después de las cuales se regresa a la pagina del menú del juego.

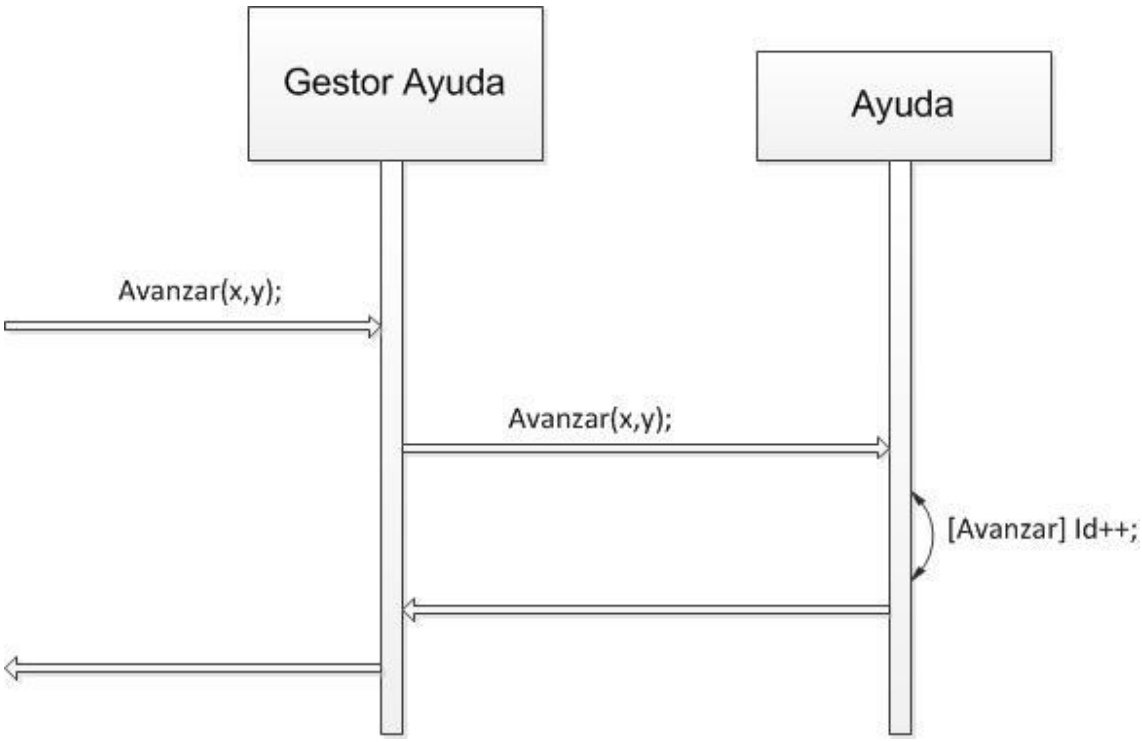


Ilustración 29: Diagrama de secuencia del contrato Avanzar

7.4.3. Contrato: Retroceder

Cuando se pulse en la zona marcada del botón de retroceso, se cambiara el Id que marca la pagina que se muestra en el botón de ayuda. Si el Id baja por debajo del numero 1, se volverá al menú del juego principal.

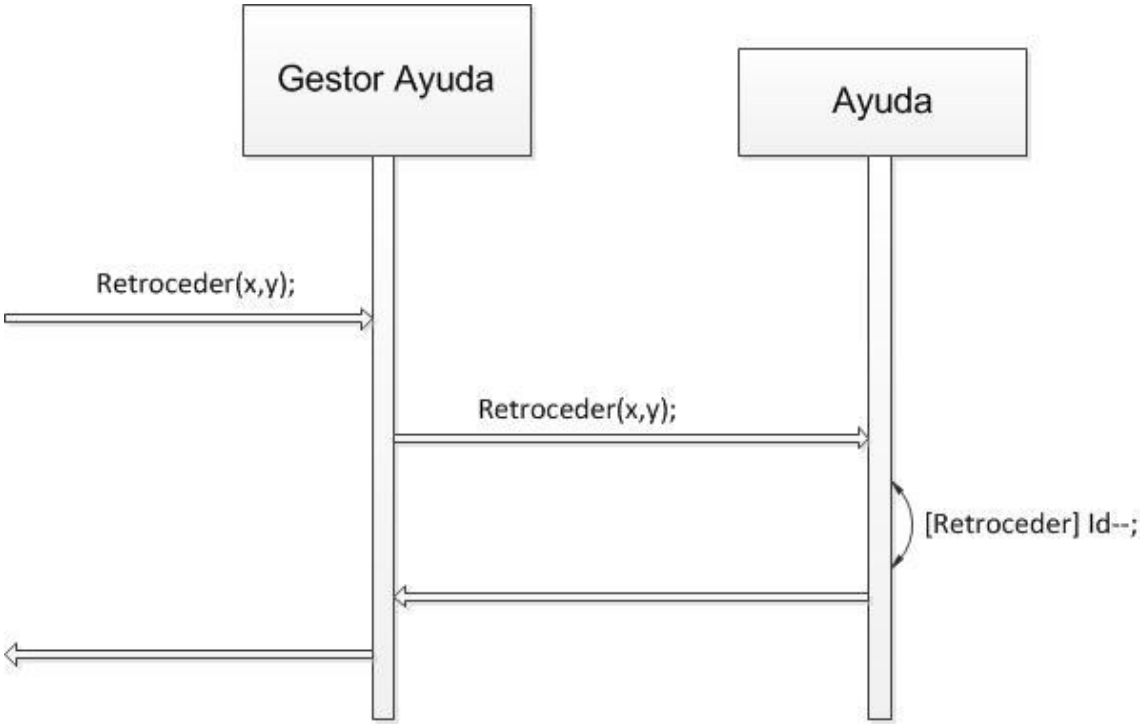


Ilustración 30: Diagrama de secuencia del contrato Retroceder

7.5. Caso de Uso Realizar Ajustes

7.5.1. Contrato: TocarAjustes

Es otro contrato básico del menú principal del juego yo como los anteriores si el usuario pulsa en el icono de ajustes, se generara una instancia nueva de la clase Ajustes, indicando el estado de los ajustes actuales.

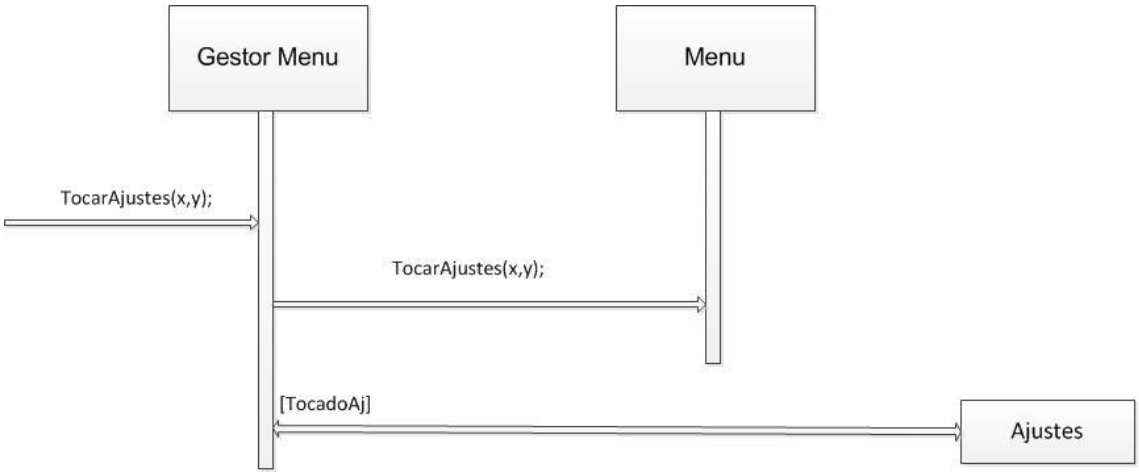


Ilustración 31: Diagrama de secuencia del contrato TocarAjustes

7.5.2. Contrato: CambiarIdioma

Una vez activado el menú de Ajustes puedes acceder a cambiar el idioma fácilmente, pulsando en la zona del idioma deseado, la clase utilizara el idioma deseado para enviárselo a la clase Ajustes y que este cambie el estado del Idioma del juego.

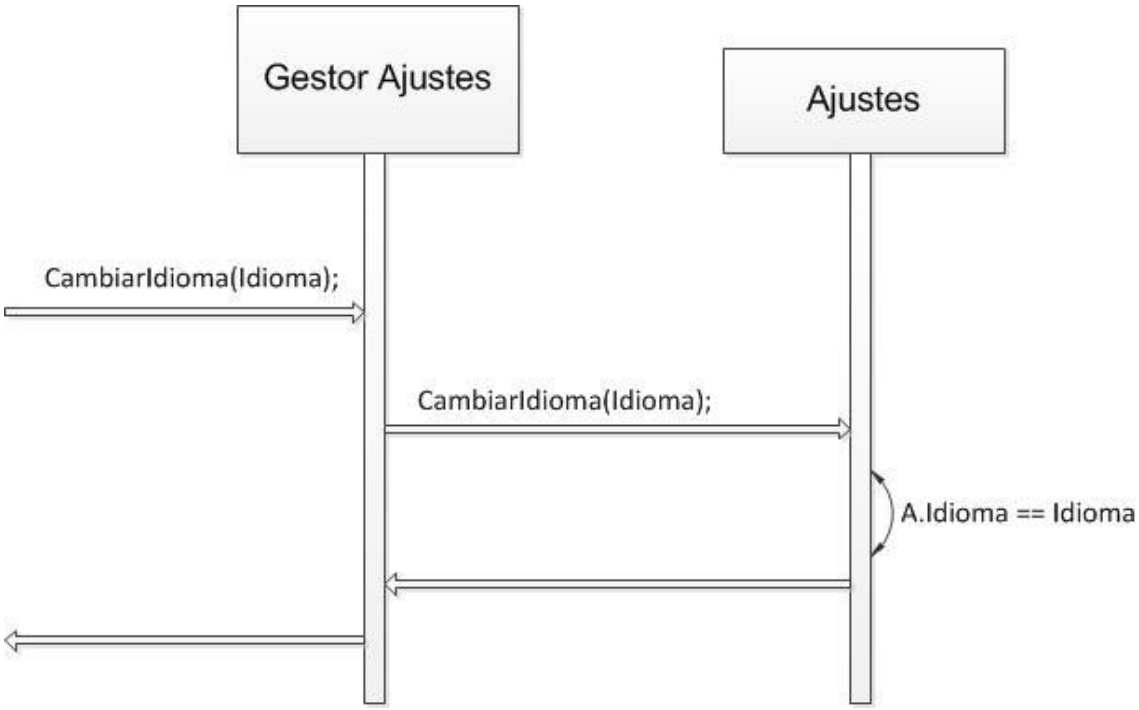


Ilustración 32: Diagrama de secuencia del contrato CambiarIdioma

7.5.3. Contrato: CambiarSonido

Una vez activado el menú de Ajustes puedes acceder a cambiar el sonido fácilmente, pulsando en el icono de sonido, se le enviara a la clase Ajustes la intención de cambiar de sonido y esta invertirá el estado del sonido actual, activándolo si esta desactivado y vicevrsa.

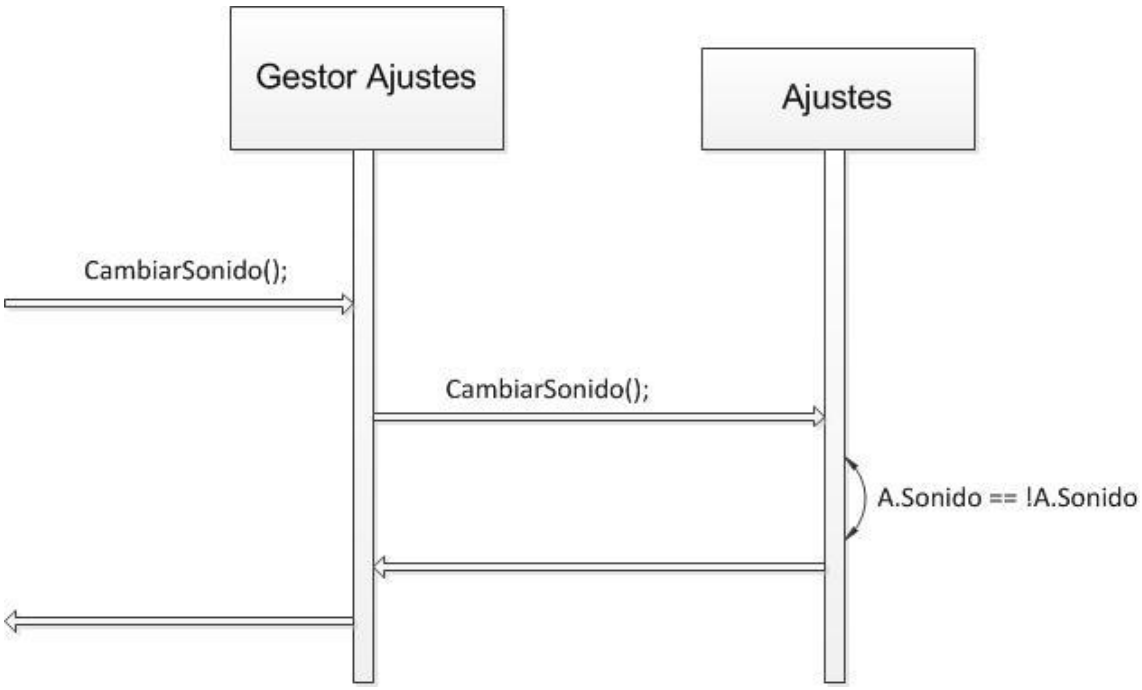


Ilustración 33: Diagrama de secuencia del contrato CambiarSonido

7.6. Caso de Uso Iniciar/Detener juego

7.6.1. Contrato: Iniciar Juego

Cuando se inicia el juego es el momento de leer de fichero. Llamando a la clase ajustes que es la que carga todos los datos de los ficheros y los establece en las variables del juego, adjudicando a cada característica su valor, y rellenando la lista de puntuaciones.

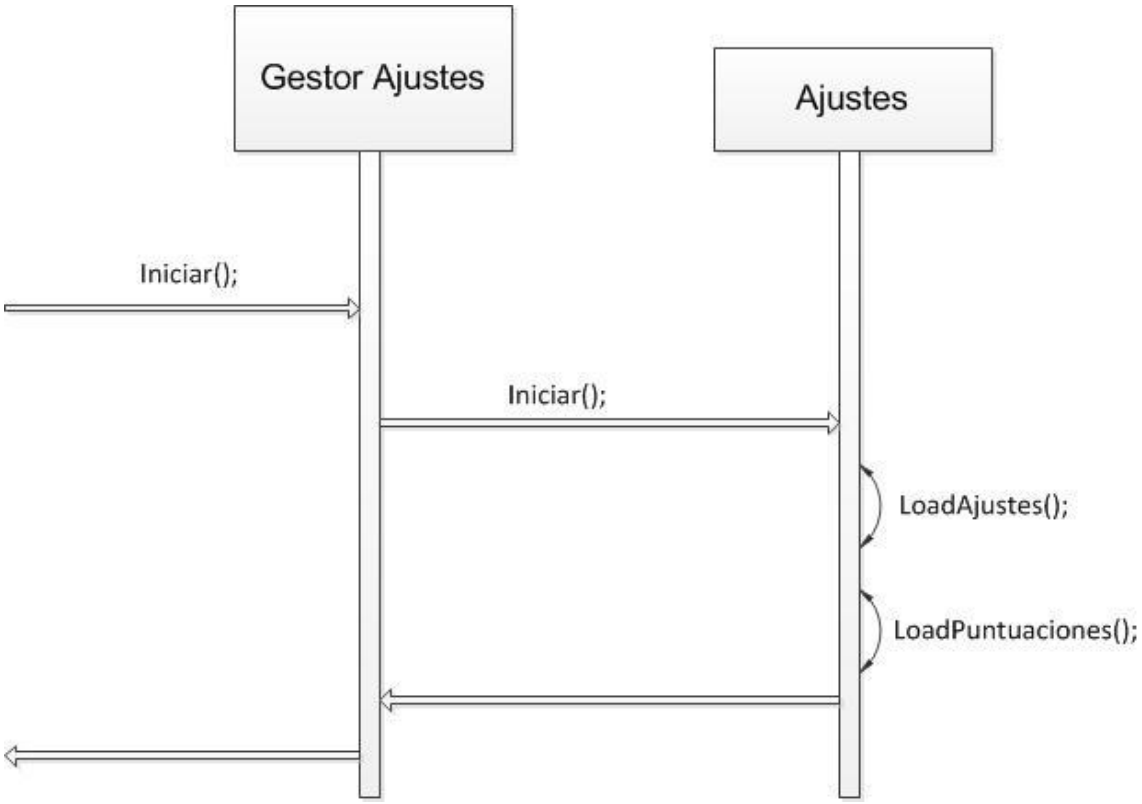


Ilustración 34: Diagrama de secuencia del contrato IniciarJuego

7.6.2. Contrato: Detener Juego

En el momento de terminar un juego es hora de guardar los datos. En este momento la clase Ajustes se encarga de guardar en un fichero el estado de las características de los ajustes tal y cual están en este momento, escribiéndolas en un orden determinado para que en su posterior lectura cuando se inicie de huevo el juego no haya ningún fallo.

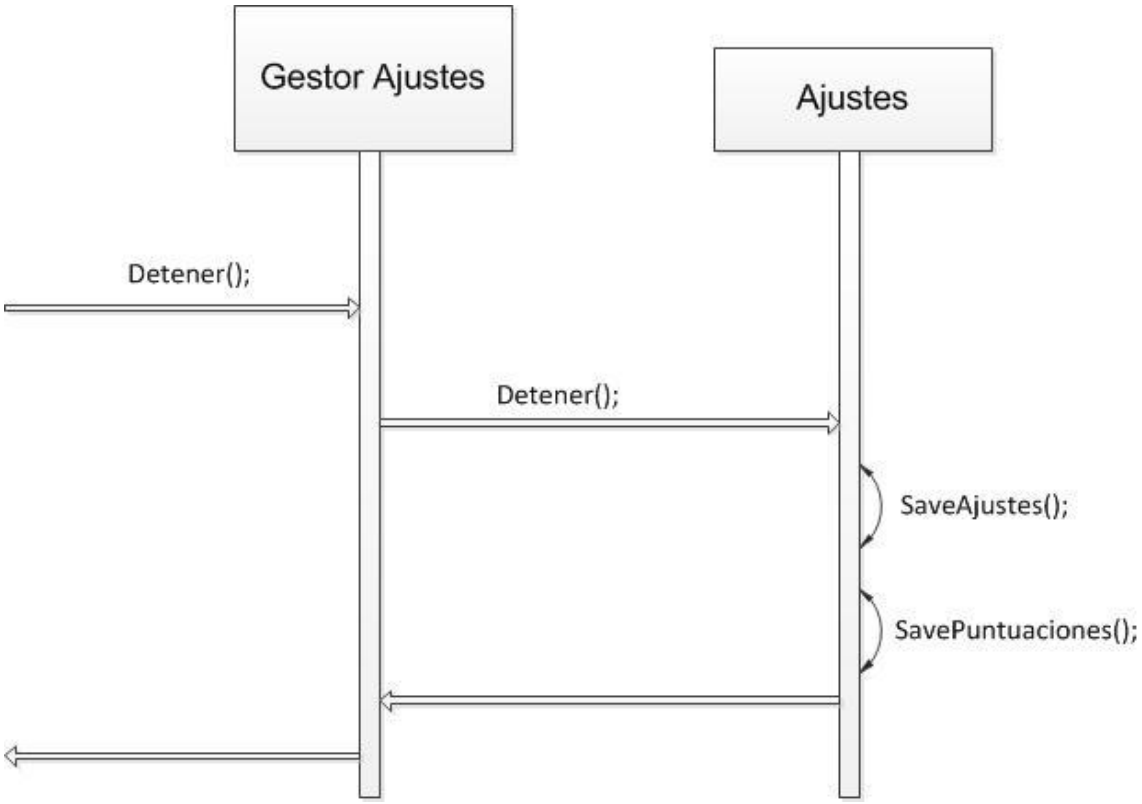


Ilustración 35: Diagrama de secuencia del contrato DetenerJuego

8. Implementación

En este capítulo se detalla la forma en la que se decidió desarrollar la aplicación. La descripción de las clases implementadas, las estructuras utilizadas, sus funcionalidades y demás aspectos para la consecución del producto final. También se mostrara el diagrama de todas las clases utilizadas para la consecución de este proyecto.

8.1. Manual de Implementación

La forma de implementar las clases de la aplicación se puede distinguir en dos partes, la parte de vista, encargada de interactuar con el usuario y recibir los datos, y la parte de controlador, que es la que implementa el motor del juego (elementos gráficos, sonidos, eventos de toque…).

8.1.1. Clases de la capa controlador

Se sabe que la aplicación será para dispositivos Android, por lo que hay que saber cómo funcionan las aplicaciones Android. El ciclo de vida de una aplicación Android es el siguiente.

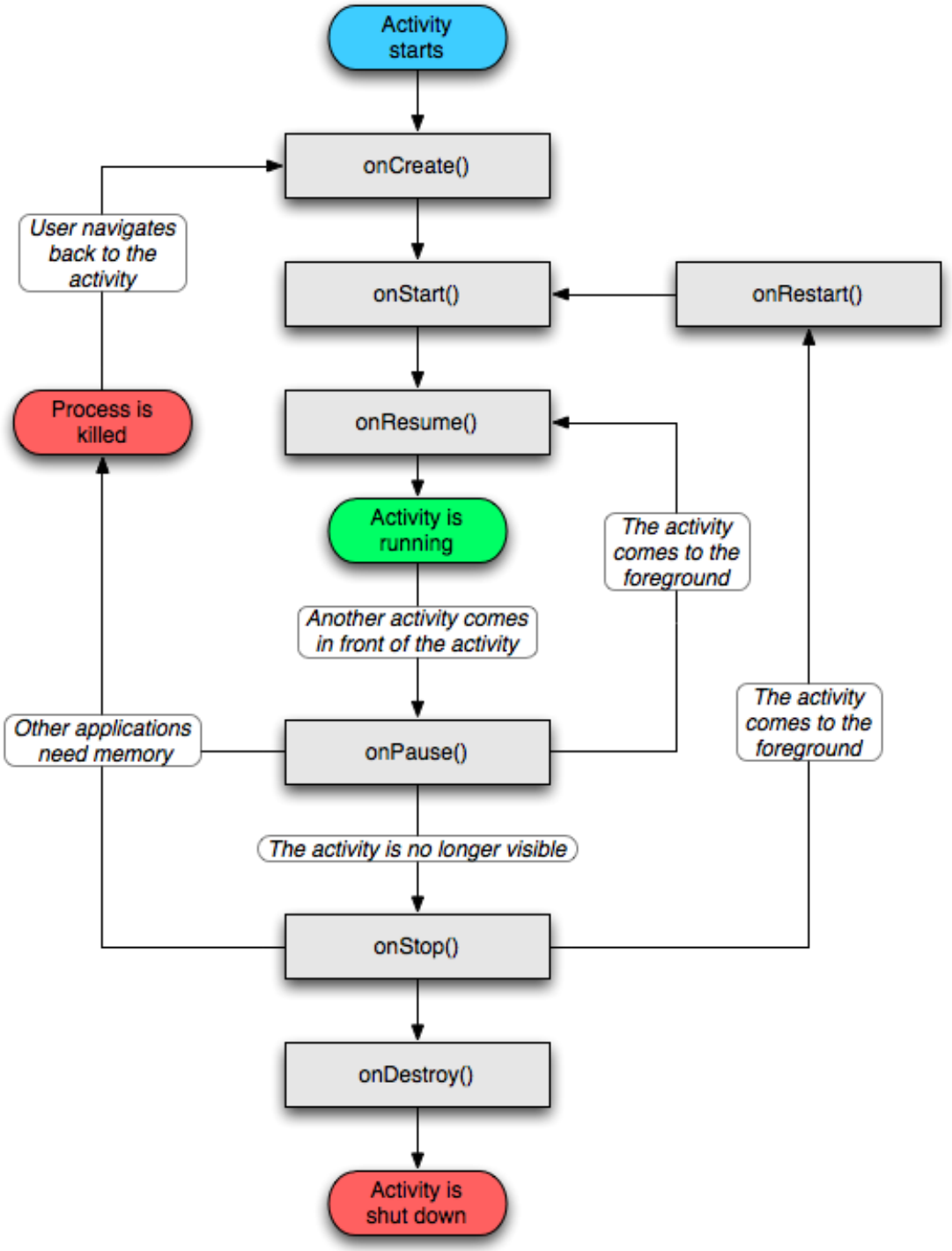


Ilustración 36: Ciclo de vida de una aplicación Android.

Lo más importante de este esquema para el desarrollo de nuestra aplicación es el evento onCreate(), por que con este evento es con el que comenzara nuestra aplicación. Así que la primera clase que vamos a crear será AndroidGame, la cual

extenderá de la clase Activity, para que así herede los métodos onCreate(), onResume() y onPause(). Esta clase AndroidGame, será el motor de nuestro programa y la que nos lleve la gestión de todas las demás clases de la control de imágenes, sonidos y demás.

Así que lo primero es explicar brevemente el funcionamiento de las demás clases de control, nombradas con la palabra "Android" delante para diferenciarlas de las interfaces, y luego entraremos en la clase AndroidGame.

Las clases AndroidAudio, AndroidSound y AndroidMusic se utilizan para crear instancias de Sound y Music procedentes de los archivos de recursos. Sound permite reproducir efectos sonoros que estén cargados completamente en la memoria RAM y Music envía a la tarjeta de sonido del dispositivo el contenido de los archivos de música almacenados en el disco. El método newSound() de la clase AndroidAudio carga un efecto de sonido almacenado en un recurso. Lo guardara en SoundPool y obtendrá una instancia de AndroidSound. En AndroidSound almacenamos SoundPool y el Id que se le ha asignado al efecto de sonido que se haya cargado, para reproducirlo más tarde con los métodos play() y dispose().

La clase AccelerometerHandler es uno de los controladores más sencillos de todos, implementa a la interfaz SensorEventListener. Almacena tres datos, tomando nota de la aceleración de cada uno de los tres ejes del acelerómetro. Fíjese que si el teléfono no tiene acelerómetro, el controlador mostrara que la aceleración de todos los ejes es 0.

La clase SingleTouchHandler implementa a la interfaz TouchHandler (una pequeña clase que a su vez implementa a OnTouchListener, y hace de elemento cohesionador entre SingleTouchHandler y MultiTouchHandler), que quiere decir que queremos que implemente a OnTouchListener. A continuación, tomamos nota del estado de la pantalla táctil para un dedo y de dos listas que obtienen los eventos TouchEvent. Tambien tenemos los métodos ScaleX y ScaleY responsable de adaptar la lectura de los eventos del dedo a todas las resoluciones de pantalla posibles

Seguimos con la clase MultiTouchHandler que, como la clase SingleTouchHandler, implementa a la interfaz TouchHandler, para que también implemente a OnTouchListener y así disponga de un par de miembros para guardar los datos de los dedos y los eventos. En vez de almacenar el estado de un único puntero, lo haremos hasta de 20. También tendremos los eventos de ScaleX y ScaleY, explicados en el párrafo anterior. El método es parecido al SingleTouchHandler, lo que pasa es que en este también guardaremos el dato de TouchEvent.pointer, que indica cual es el índice del evento (en el Single, siempre es 0), para saber a qué dedo está asociado cada evento.

Una vez definidos estos tres últimos controladores podemos implementar la interfaz Input en la clase AndroidInput. Esta clase reúne todos los controladores del acelerómetro y los eventos de pantalla. Todas las llamadas que se hagan a los métodos se delegarán en el controlador correspondiente. La unica responsabilidad que tendremos que implementar será la de TouchHandler, que usando como referencia la versión de Android del dispositivo, delegara el control a SingleTouchHandler o a MultiTouchHandler, si la versión del SDK es inferior a la 5, se usará el SingleTouch dado que el dispositivo no está diseñado para procesar más de un evento a la vez, sino, se utilizara MultiTouch.

La clase AndroidGraphics dibujara pixels, líneas, rectángulos y pixmap en el framebuffer. Podemos recurrir al Bitmap como un framebuffer virtual al que dirigir todas las llamadas relacionadas con el dibujo a través de Canvas. En mayor medida se usara para generar instancias de pixmap procedentes de archivos de recursos. Estos pixmap los dibujaremos mediante el método drawPixMap(), que dibujara todo el pixmap en el framebuffer virtual en las coordenadas que se le especifiquen. Las clases getWidth() y getHeight() nos darán el tamaño del frame buffer virtual de la instancia de AndroidGraphics.

La clase AndroidFastRenderView, que extiende de la clase SurfaceView, se encarga de pasar a usar un hilo de ejecución independiente y dibujar de forma continua. Este hilo nos ayuda a la ejecución del bucle principal del juego, y podemos configurarlo para que utilice canvas para dibujar constantemente en SurfaceView. Esta clase guarda una referencia a la instancia Game, que utilizaremos para obtener el elemento Screen activo. Llamaremos constantemente a Screen.update() y Screen.present(), desde el hilo de ejecución FastRenderView. Guardará también un registro del intervalo de tiempo que transcurre entre fotogramas, para llevar un control sobre la velocidad del juego, para que sea siempre la misma en cualquier terminal en el que se instale. Y tomara también el framebuffer virtual que dibujará la instancia AndroidGraphics en SurfaceView para modificar su escala si fuera necesario.

La parte controlador del juego está ya casi descrita, solo nos queda explicar la clase AndroidGame, como elemento cohesionador de todos los controladores hasta ahora descritos.

La clase AndroidGame implementa la interfaz Game. Se encargará de la gestión de la ventana, esto implica que tenemos que configurar una actividad y una vista AndroidFastRenderView y hacernos cargo del ciclo vital de la actividad de forma clara. AndroidGame también gestiona WakeLock para que la pantalla no pase a modo ahorro de energía y pierda su brillo, inicia y dirige las referencia destinadas a Audio, Graphics y Input a las partes interesadas, y lo que le da visión a la actividad, gestionar Screen e integrarlo en el ciclo vital de la actividad.

El objetivo es tener una única clase AndroidGame, de la cual podamos obtener todos los elementos que necesitamos.

Aún habiendo explicado sólo las clases controladoras ya se puede entender el por qué de haber diseñado la arquitectura así, y es que de esta manera se puede reutilizar perfectamente las clases controladoras descritas anteriormente con otros juegos que podamos desarrollar en el futuro.

8.1.2. Clases de la capa Vista

Las clases de la capa Vista son las más simples ya que se limitan a usar los controladores vistos anteriormente para su funcionamiento y de lo único que tienen que responsabilizarse es del comportamiento de cada una dependiendo de la pantalla en la que se encuentren.

La aplicación se inicia desde la clase F1PitStopActivity que extiende de la clase AndroidGame, a continuación llama la clase loadingScreen, que carga todos los datos de imágenes y sonido utilizando sus respectivos controladores, y más adelantes carga las puntuaciones y los ajustes. Una vez hecho esto pasamos al juego de pantallas.

A partir de aquí, todas las clases generadas extienden de la clase Screen. Estas clases irán jugando entre ellas, llamándose las unas a las otras. Cada vez que se llame a una clase nueva se utilizara el método game.setScreen() que cambiará el estado del Screen actual a la pantalla que se le especifique.

Cada una de estas clases tiene sus métodos update() y present() para su correcta manipulación desde los controladores. En el método update() se llevan a cabo todas las operaciones, cambios de variables y control en general de la pantalla en cuestión, y en el método present() se irá componiendo el pixmap que pintara el controlador grafico de acuerdo a las especificaciones presentes en cada Screen.

Los Screen que existen en este juego son los siguientes: PantallaPrincipal, Puntuaciones, Ayuda, Ajustes, Juego, PruebaJackMans, PonEnOrden, PorElCamino, MueveTuercas, TocaRapido y Final.

8.2. Alternativas de Implementación

En este apartado se estudiara la información que se ha encontrado, así como otras posibles formas de resolución de los problemas que ha planteado la implementación del proyecto.

Sobre el tema Android hay infinidad de información. Este sistema cuenta con una cantidad de desarrolladores inmensa. Por lo tanto la información relevante a este

tema es aún mayor, así que manuales de programación en Android hay miles. Casi todo el contenido esta en ingles, así que la búsqueda de información se hace más larga y costosa, pero no supone un problema grave ya que el alumno domina el nivel medio de dicho idioma. La implementación se ha seguido siguiendo un único manual, pero como he dicho antes, hay tantos manuales que la aplicación podría haberse desarrollado de mil maneras distintas.

Aunque se haya basado la creación del juego en un único manual, siempre es inevitable tener que hacer consultas en la red sobre distintos tema que no quedan bien definidos en el manual. Aquí llega un pequeño problema, que es que habiendo tanta información sobre diseño en Android, encontrar aquello que necesitabas justamente para tu implementación no resultaba fácil, y al final se sacaba la solución sacando una pequeña idea de cada párrafo de código, logrando así el algoritmo que necesitaba para este proyecto.

Salvo esas pequeñas consultas el manual fue de gran ayuda en la realización del juego, con grandes partes de comentarios sobre cierto código que a priori puede resultar confuso, y ayudando a una finalización más vistosa del juego dando algún pequeño consejo sobre diseño grafico de la aplicación.

8.3. Diagrama de Clases

Las clases descritas anteriormente pueden verse en la imagen que se muestra a continuación. En esa se ven las clases de control y las clases de vista y como están relacionadas entre ellas.

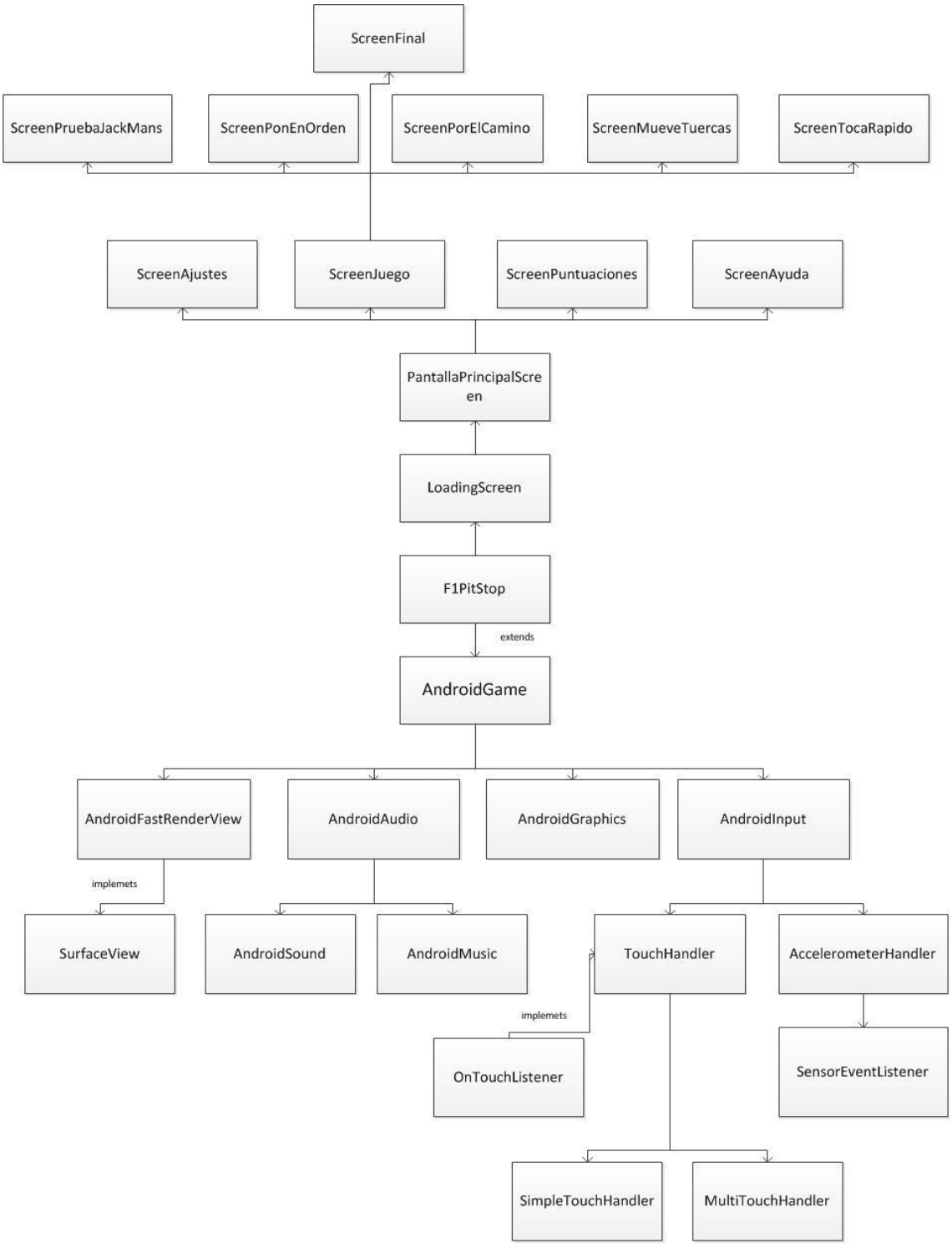


Ilustración 37: Diagrama de clases.

8.4 Estructura de ficheros

En este apartado describiremos como están distribuidos los ficheros dentro del workspace. Qué tipo de archivos se encuentran en cada carpeta y mediante la Ilustración 36, donde encontrar cada carpeta. Hay que tener en cuenta que solo voy a mencionar las más importantes, dado que hay carpetas que auto genera eclipse y que su funcionalidad se reserva solo para ese entorno de trabajo y no interesa en este proyecto.

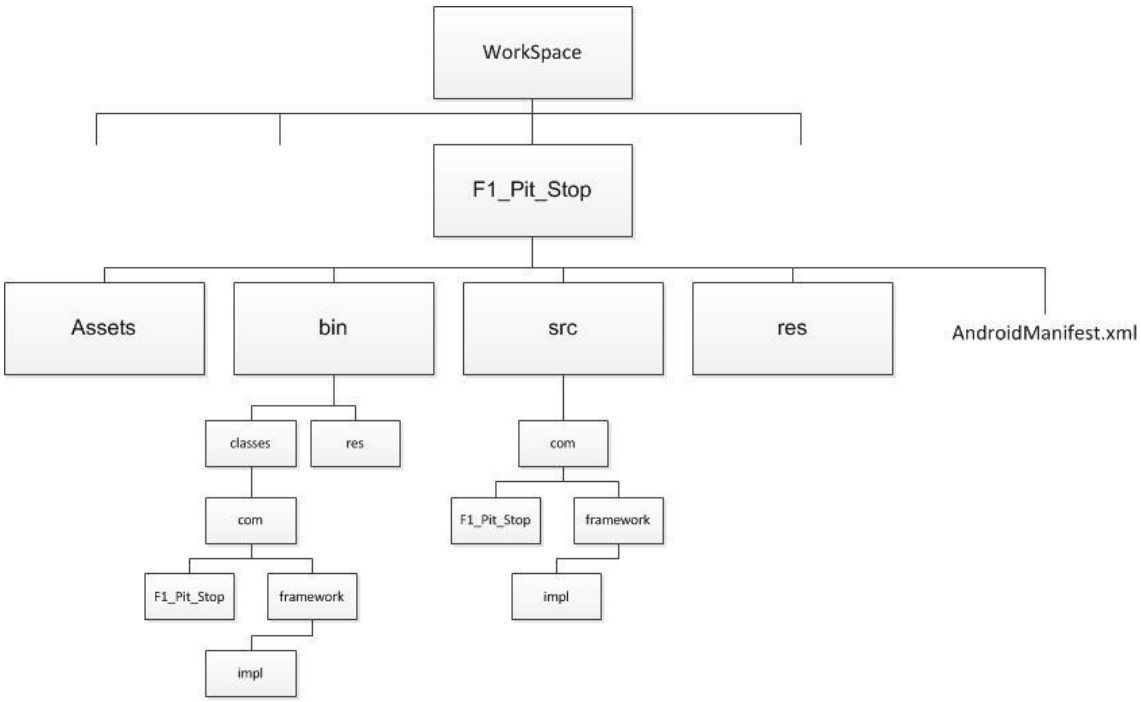


Ilustración 38: Estructura de archivos.

Dentro del workspace, nos encontramos todos los proyectos que hemos creado entre ellos el nuestro, F1_Pit_Stop. Dentro de esta carpeta vamos a diferenciar las más importantes.

Empezaremos explicando la carpeta Assets, en ella se guardan todas las imágenes, música y sonidos que utiliza la aplicación. La carpeta res guarda los iconos de la aplicación de distintos tamaños, para usarlos en smartphones de distintos tamaños, o tablets y netbook, que tienen un tamaño considerablemente distinto. En la misma altura que estas dos carpetas nos podemos encontrar el archivo AndroidManifest.xml, que es el archivo en el que se declara la orientación de la aplicación, los permisos requeridos, la clase principal, detalles de versiones mínimas, etc.

Ahora explicare las carpetas bin y src. Empecemos con la carpeta bin. En esta carpeta encontramos un archivo importante, el .apk. Cada vez que compilemos la aplicación se guardara en esta carpeta /bin. Dentro de esta, encontramos otras dos carpetas, clases y res. En res, tenemos lo mismo que en la del nivel superior, las imágenes de los iconos. Pero si abrimos /classes nos encontramos con la carpeta /com, y dentro de esta, dos carpetas donde se dividen todos los archivos binarios que componen el juego. En la carpeta F1_Pit_Stop estarán todos los archivos binarios de las clases del modelo vista. Mientras tanto, en la carpeta Framework estarán los archivos binarios de las interfaces usadas en la capa de control del juego, así mismo, en aquí nos encontramos una carpeta llamada /impl, en la que dentro estarán los binarios de las implementaciones de estas interfaces.

La carpeta /src, en el mismo nivel que la /bin, tiene la misma estructura que la /bin, exceptuando el primer nivel de esta, la única diferencia es que en ella se guardan los archivos .java, en vez de los .class.

9. Pruebas

Durante y tras la implementación se han realizado una serie de pruebas sobre el programa para determinar el correcto funcionamiento del mismo.

Se ha intentado buscar casos de prueba que tengan altas probabilidades de hacer fallar el programa para una depuración correcta del mismo. Se han probado todas las partes del programa, tanto botones como movimientos en la pantalla o diferentes tipos de configuración.

Aun así, por muchas pruebas que se realicen, no se garantiza un programa perfecto, ya que este tipo de programas tienen infinidad de movimientos y posicionamiento de elementos, con lo cual nunca se llegaría a probar todas las posibilidades. Por esa razón los programas deben de pasar por un proceso de depurado después de su finalización que se encargue de corregir los posibles errores que puedan haber surgido.

9.1. Pulsar el botón “Back” y “Home”

Estos botones deberían de permitirnos salir del programa en cualquier momento. El programa se cerrara y la única forma de volver al programa será ejecutarlo de nuevo. Se comprueba que este botón funciona correctamente.

9.2. Un Toque en la Pantalla

Se tiene que poder tocar la pantalla en las zonas no activas. Esto en principio no debería de realizar ninguna acción ni siquiera al desplazar el dedo.

9.3. Fallo en prueba de las marcas

En la prueba de las marcas se encontró un grave fallo. Cuando las dos marcas azules eran las más cercanas a la misma marca roja, solo uno de los mecánicos se desplazaba a su posición. Esto hacia que el juego se bloqueara, dado que el juego solo seguirá funcionando cuando los dos jackmans hayan alcanzado su posición final. Revisando el código se encontró los fallos y se corrigieron.

9.4. Fallo al insertar una puntuación nueva

Cuando se ha intentado insertar una puntuación nueva en el array de puntuaciones y actualizarlo para ponerlo en orden y que solo muestre las cinco mejores, ocurria un fallo fatal, bloqueaba el programa por completo impidiendo incluso su cierre. Este fallo todavía no ha sido solucionado y se está trabajando en ello, mientras tanto el programa no guardara las puntuaciones nuevas.

9.5. Sin permiso de E/S en tarjeta

Cuando se procedía a instalar la aplicación solo pedía el permiso de bloque de pantalla, cuando también debería pedir el de acceso a la tarjeta, habiendo incluido ya los dos permisos en el AndroidManifest, donde se especifican los permisos de la aplicación. Sin este permiso resulta imposible crear un archivo en el teléfono donde guarda ajustes y puntuaciones. Comprobando más analíticamente el AndroidManifest, comprobé que el error radicaba en un fallo de escritura del permiso, una simple letra escrita en minúscula en vez de en mayúsculas.

9.6 Llevar la cuenta de tiempo y pruebas

Durante el transcurso del desarrollo del juego me di cuenta de que se me antojaba una duda como llevar la cuenta de las pruebas completadas hasta la fecha, dado que cada vez que se termina un mini juego se inicia de nuevo la clase de juego principal. La solución era tan sencilla como pasarle a la constructora una variable que marque las pruebas hasta la fecha. Esta variable también se le pasaría a los mini juegos y cuando uno de estor termine, amplia en uno si valor y crea la clase de juego pasándole este valor.
Esta técnica también se ha aplicado para llevar la cuenta del cronometro y el tiempo durado en cada prueba.

9.7. Fallo indeterminado

En mitad de la implementación el juego simplemente dejo de funcionar. A la hora de iniciar el juego, este no iniciaba y visualizaba un globo de alerta en el que decía que la aplicación había dejado de funcionar. Cuando intentamos volver a escribir el código como estaba no lo conseguí, y de hecho, la cosa empeoro. No logre encontrar el error, y persistió durante varios días.
Así que mi mejor opción fue restablecer el código usando uno de los back ups existentes.

9.8. Movimiento del coche

Cuando empieza el juego y el coche aparece desde un lado de la pantalla y va decelerando hasta colocarse en su posición. El fallo era que al hacer la función de movimiento decelerado del coche, este no hacia lo que debía y, o se pasaba de la marca o no llegaba o hacia movimientos extraños y entraba en bucle infinito. Revisando el código, averigüe el error y a base de intentos se solucionó.

9.9. Pruebas de Implantación

Se deberá de probar el correcto funcionamiento de la aplicación en diferentes dispositivos. Con diferentes tipos de hardware. También deberían de tener diferentes versiones de software. Como testearlo en todos los dispositivos del mercado y con todas sus versiones es una tarea prácticamente imposible se testeara en varios dispositivos diferentes. Esta prueba ha sido completada satisfactoriamente en todos los dispositivos probados hasta la fecha.

9.10. Pruebas de Configuración

También se comprueba con diferentes configuraciones. Por ejemplo en el modo avión de los dispositivos, con el brillo muy bajo en la pantalla, desactivando internet... Ninguna de las configuraciones debería dar ningún problema al programa.

10. Implantación

10.1 Especificaciones

La aplicación se ha programado utilizando la API de Android versión 7, lo cual permite ejecutarla en dispositivos que contengan Android 2.1 o superior. Se ha elegido llegar hasta esta API por que la aplicación utiliza el modo MultiTouch, que solo está disponible en dispositivos con Android 2.1 o superior y el juego no funcionaria con versiones anteriores. En cuanto al tema de hardware el único requisito es que el dispositivo disponga de pantalla táctil, porque cualquier dispositivo que sea capaz de ejecutar Android 2.1 podría ejecutar también la aplicación.

10.2 Compilación

La compilación de la aplicación se realiza mediante Eclipse. El SDK nos genera un archivo de la extensión .apk en la carpeta /bin del proyecto con el nombre del proyecto. Este archivo se genera automáticamente cada vez que se ejecuta la aplicación en las maquinas virtuales del SDK. Este archivo .apk es el que debemos de copiar y pegar dentro del dispositivo donde lo queramos instalar. Este archivo contendrá todos los datos necesarios para el funcionamiento de la aplicación, como el manifiesto de Android, los binarios, las imágenes y sonidos, etc…

10.3 Instalación en dispositivos

La instalación en cualquier dispositivo se explicará más detalladamente en el apéndice de la guía de instalación para el usuario. Este proyecto ha sido probado en 5 terminales Android distintos y en los 5 ha funcionado correctamente sin ningún problema:

- Samsung Galaxy Ace con Android 2.1 y Android 2.3.5
- Samsung Galaxy Ace II con Android 2.3.6
- Samsung Galaxy S II con Android 4.0 y Android 4.1
- Samsung Galaxy S III con Android 4.0.4
- Tablet BQ Kepler, con Android 2.1

Todos ellos cumplían los requisitos y se veía la aplicación en la pantalla perfectamente ajustada a pesar de que cada uno tenía un tamaño de pantalla y resolución distinta. Ha estado bien poder testear la aplicación en alguna tablet, para poder comprobar el funcionamiento en dispositivos con diferencias más importantes que las que pueda tener un teléfono con su siguiente modelo. Aun así no pude probarlo en un netbook con este SO, pero gracias a las maquinas virtuales proporcionadas por el SDK se pudieron especificar los parámetros de un netbook y hacer la prueba de su funcionamiento, y la aplicación funciono a la perfección.

11. Gestión

Desde que se empezó el proyecto se supo que iba a ser un reto. Y si se quería entregar en los plazos, resultaría un trabajo difícil y duro, que iba a tener múltiples problemas y que su desarrollo seria costoso. Prueba de ello ha sido que algunas características no ha sido posible desarrollarlas y que el tiempo ha estado muy justo entre la finalización del proyecto y la entrega.

La primera iteración fue más fácil de lo esperado, utilice esta iteración para la formación sobre programación Android. Conseguí todas las herramientas necesarias y dar mis primero pasos en la programación Android con programas simples. Una vez hice esto la cosa se empezó a complicar cuando comencé a implementar la estructura del programa, toda la conexión entre interfaces y clases supuso un trabajo muy duro, donde hubo muchas trabas y muchos parones. El trabajo se desarrollaba muy lento y poco a poco, pero en el momento en el que tuve todos los controladores entrelazados y en funcionamiento todo fue a mejor. Además, aunque fuera un trabajo duro, solo tenía una asignatura que estudiar a la vez que hacia el proyecto, con lo cual podía dedicar todas las horas a resolver los problemas surgidos, y en un par de semanas muy intensas tuve este tema de los controladores resuelto. Esto termino sobre finales del mes de Octubre 2012.

La segunda iteración fue más entretenida, ya que consistió en hacer las clases visuales y fue en trabajo más creativo visualmente hablando. Hubo que crear las imágenes utilizando Photoshop e ir montando las pantallas juntando las imágenes de la forma que quisiera para crear el juego.

Una vez creadas todas las imágenes y conseguido todos los sonidos se estableció la conectividad entre ellas y el funcionamiento de cada una, ya fueras los mini juegos o las pantallas de transición, o las de ajustes. Aquí también hubo algunos problemillas pero nada importante a excepción de uno que fue imposible de hallar. Este fallo me trajo de cabeza durante varios días, y después de no conseguir resolverlo ni de poder volver al estado anterior, porque también fallaba, decidí utilizar el último back up realizado y empezar de nuevo desde el punto en donde lo deje en el back up. Termine esta parte en Diciembre de 2012, y con ella la aplicación quedaba prácticamente realizada.

A partir de aquí el tiempo se invirtió en terminar la memoria del proyecto, cosa que me costó bastante más de lo que pensé en un principio, ya que no era algo que incitara al alumno, y se realizaba muy lentamente.

Las horas invertidas han sido prácticamente las estimadas solo que no están distribuidas como creía que iban a quedar. En un principio se estimo que iba a durar entre 150 y 180 horas, Aunque se han empleado alguna horas más de la debida queda dentro del margen aceptable. Este recargo de horas se debe a la inexperiencia del alumno en el campo de redactar memorias tan extensas, y necesidad de formación en programación Android, pero esto último en menor medida.

Tareas	Horas Planificadas	Horas Reales
DOP	7	7
Captura de Requisitos	8	9
Análisis	13	12
Diseño	14	17
Implementación	65	55
Pruebas	5	5
Memoria	20	45
Formación	45	50

Ilustración 39: Tabla Horas previstas vs Horas Reales

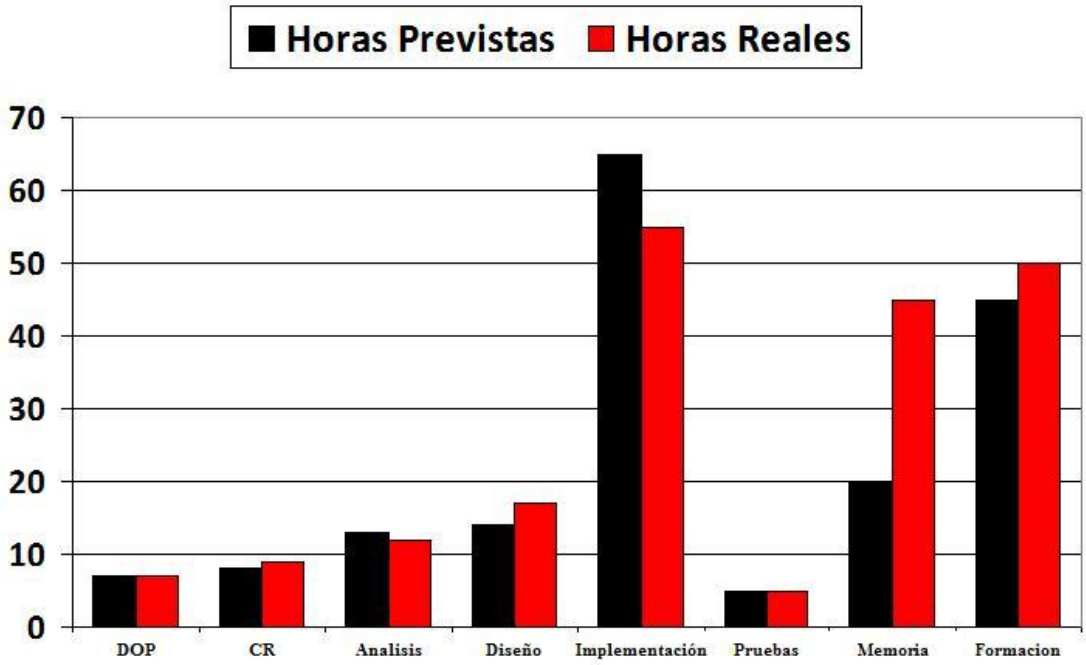


Ilustración 40: Grafico Horas previstas vs Horas Reales

Al igual que el número de horas invertidas contra las estimadas no tienen una diferencia extremadamente grande, con el diagrama de Gantt ocurre lo mismo. Al realizar el proyecto en unos pocos meses las diferencias que pueden existir entre los plazos de realización no son extremadamente grandes. La diferencia más grande es de 10 días como mucho.

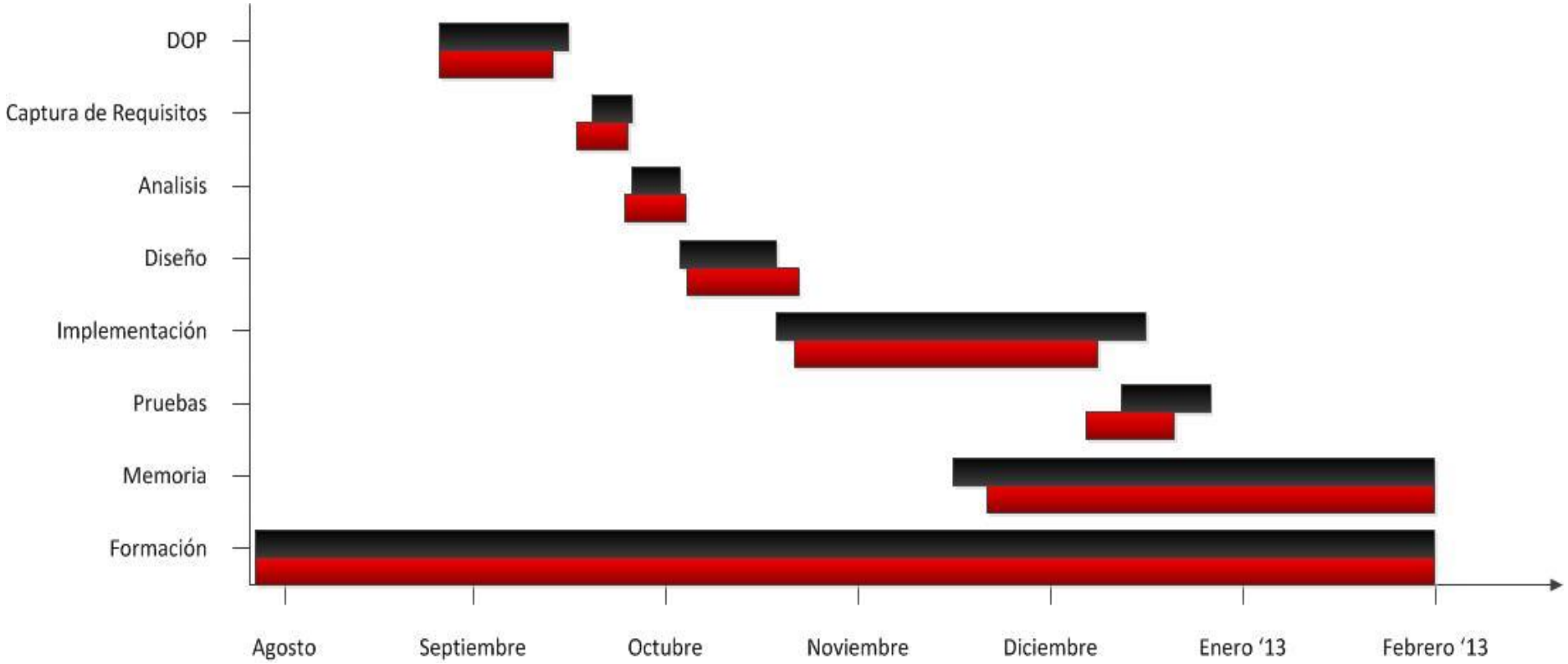


Ilustración 41: Diagrama de Gantt Real vs Gantt Planificado

12. Conclusiones

12.1 Objetivos logrados

El objetivo del proyecto era entender y desarrollar los aspectos básicos de programación en Android, tratamiento de imagen y sonido, eventos de entrada y salida, lectura de archivos, etc. Por lo tanto, el objetivo se da por cumplido y estoy satisfecho por ello. El trabajo ha sido un éxito ya que tiene todos sus mecanismos funcionan perfectamente.

Hubiera estado bien poder invertir más horas en la aplicación para realizar las ideas mencionadas en la parte de Mejoras futuras, pero ha resultado imposible porque el número de créditos que tiene la asignatura impide la inversión de más tiempo. Esta sería una pequeña pega, ya que me gustaría haber hecho alguna cosa más, aunque lo más importante y complejo sí que ha sido posible realizarlo.

En cuanto a lo referente a la experiencia de programar para Android me ha gustado bastante. Como cualquier entorno de programación al principio cuesta entender cómo se programa para Android, pero en cuanto le coges el truco, todo resulta más fácil y surgen nuevas ideas.

12.2 Valoración personal

Me siento satisfecho con la realización de este proyecto. He cumplido todos los retos que me he propuesto en el tiempo estimado y con buen resultado. La verdad es que me ha resultado más fácil de lo que pensaba al principio, pero ya que Android se programa en Java y ya estaba muy familiarizado con este lenguaje me ha sido más fácil. Los únicos contratiempos surgieron con la idea de integrar multi touch y lectura de memoria, pero aún así se logró con éxito. Todos los conocimientos que he adquirido han sido de forma autodidacta ya que no había recibido ninguna clase de específica que tuviera que ver con el tema de programación en Android.

Por todo esto uno se siente orgulloso de haber llegado a este punto. Por ver que todas las horas invertidas han logrado cumplir los objetivos propuestos.

En definitiva me siento muy orgulloso de haber realizado el proyecto de esta manera. Está claro que las aplicaciones para smartphones ofrecen multitud de posibilidades para el futuro y es un campo en el que vendrá bien tener conocimientos. Si tuviera que volver a elegir un proyecto, sin duda seria una aplicación para Android.

12.3 Mejoras Futuras

El estado del juego tras finalizar el proyecto ha quedado bastante completo y con multitud de funcionalidades que ofrece el teléfono así que queda poco espacio para ofrecer mejoras futuras.

Las pocas mejoras que se me ocurren serian únicamente aspectos de jugabilidad, nada en cuanto a la programación. Se podría insertar un menú de opciones más

ámplio, la capacidad de desbloquear equipos diferentes de formula 1 a medida que avances en el juego y poder elegirlos en este menú o también insertar mas sonidos, como al coche de formula 1 en su llegada o cuando los mecánicos ponen la rueda en su sitio.

Otra mejora, aunque es de un nivel altamente superior, sería hacer un ranking de puntuación on-line, aparte de tener el ranking propio, también tendrías opción de ver las 10 mejores puntuaciones de todas las personas que jueguen a este juego, y así darle más emoción y meter un plus de enganche al juego. Esto requeriría una identificación del usuario si lograse entrar en el ranking.

13. Apéndice

13.1 Manual de Usuario

Tras cargar la aplicación se mostrara el menú principal, en el tienes todas las opciones que ofrece el juego; Jugar, Puntuaciones, Ayuda, que estarán en una lista a la izquierda, y el botón de ajustes, que está en la esquina inferior derecha, como muestra la Ilustración 42.

Ilustración 42: Menú principal del juego.

Ahora pasare a explicar cada una de las opciones. Lo primero que explicare serán los ajustes, que es el botón de la esquina, como ya he mencionado antes. Cuando pulse el botón se abrirá un menú desplegable con las opciones de los ajustes (Ilustración 39). Estas opciones son las de activar y desactivar sonido, y la de cambiar el idioma. Todos los cambios se efectúan en el momento, el cambio directo de idioma se ve entre las ilustraciones 43 y 44. El idioma seleccionado destaca con brillo verde y el sonido sera tachado con una equis roja si esta desactivado. El menú se cerrara pulsando en la flecha naranja que hay debajo de los ajustes.

Ilustración 43: Menú de ajustes.

Ilustración 44: Menú de ajustes en ingles.

Pulsando en la zona de puntuaciones de la pantalla, se mostrara una pantalla nueva, en la que aparecen las cinco mejores puntuaciones conseguidas hasta la fecha en el juego, imagen recogida en la Ilustración 45. Para volver al menú principal se debe pulsar en la flecha naranja de la esquina inferior izquierda.

Ilustración 45: Menú de puntuaciones.

El siguiente menú es el de ayuda, para acceder a él solo hay que pulsar en la zona indicada en la pantalla. Después de esto aparece la primera pantalla de ayuda del juego, Ilustración 46. Para avanzar entre las pantallas de ayuda se debe pulsar la flecha naranja de apunta hacia la derecha, situada en la parte inferior de la pantalla. Para retroceder se utilizara la flecha que está al lado de esta ultima mencionada, pero mirando a la izquierda. Cuando llegues a la última pantalla de ayuda y vuelvas a dar a avanzar, volverás al menú principal, de igual manera que si estas en la primera y pulsas en retroceder. Estas pantallas, como todas las demás, también sufren los cambios de idioma si se requiere.

Ilustración 46: Menú de Ayuda en Español.

Ilustración 47: Menú de Ayuda en Ingles.

La última opción que queda por explicar es la de jugar. Como el resto, basta con pulsar en la zona de la palabra Jugar y comenzara el juego automáticamente. Lo primero que aparece será un coche de formula 1 llegando al lugar de boxes, en el momento en el que llegue el coche comenzara de inmediato el primer juego, que se muestra en la Ilustración 48. En este juego aparecerá una barra horizontal con dos marcas rojas, habrá una tercera marca azul que se moverá de un lado al otro, lo que hay que hacer es pulsar la pantalla en el momento en el que la marca azul este lo más cerca posible de una de las marcas, después de haber ajustado una de las marcas, deberás acercar la marca azul a la otra marca roja restante. Cuanto más cerca estén las marcas azules de las rojas más rápido serán los movimientos de los hombres de los gatos, y solo se podrá continuar cuando los dos hombres de los gatos estén en sus posiciones.

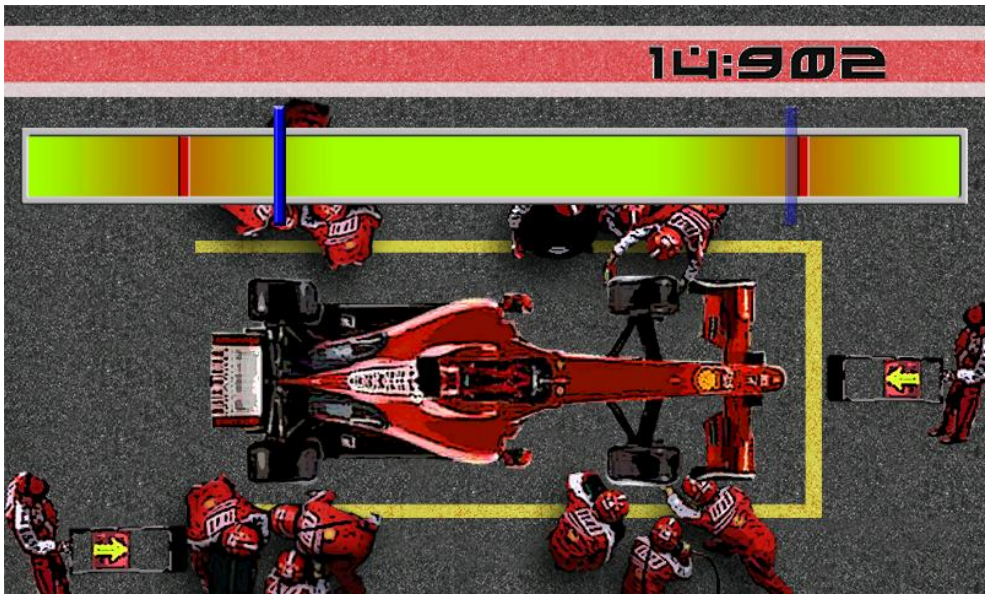


Ilustración 48: Juego de las marcas, con una marca ya parada.

Una vez completado el primer mini juego de los hombres de gato, se verá claramente el aspecto de juego principal. En esta pantalla se verán los cuatro grupos de mecánicos en los que hay que pulsar para comenzar los mini juegos. Los grupos que estén con el brillo naranja, serán los que queden por pulsar para activar los mini juegos. El escenario principal del juego se muestra en la Ilustración 49, en este caso se ve que quedan todas las pruebas por resolver, ya que los cuatro grupos de mecánicos están brillando.

Ilustración 49: Escenario del Juego.

Los mini juegos se realizaran en un orden diferente cada vez que se juegue al juego. Así que el orden en el que los explique aquí no tiene porque ser el mismo que se muestre en una partida cualquiera.

Empezare explicando el juego de "Tuercas en orden". Este juego consiste en pulsar rápidamente las seis tuercas que aparecen en la pantalla en orden ascendente, según el número que tengan encima, primero la tuerca 1, luego la 2, etc. Cuando pulses una tuerca de manera correcta, brillara en color verde, si en algún momento fallases, todas las tuercas se apagarían y habría que volver a empezar desde la tuerca numero 1. Las tuercas se posicionan de una manera distinta cada vez que se juega. En la ilustración 50 se puede ver este juego, y en este caso, el usuario ha pulsado correctamente hasta la tuerca nuero 5 y solo le queda pulsar la sexta para concluir el mini juego.

Ilustración 50: Mini juego Tuercas en orden.

El siguiente juego es "Por el camino". Consiste en llevar la rueda hasta su posición en el coche, sin salirse del camino naranja. Para mover la rueda únicamente inclina el móvil hacia la dirección donde lo quieras mover. Cuanto mayor sea el grado de inclinación, mayor será la velocidad a la que se mueva la rueda. Como en todos los mini juegos, en el momento que falles, el mini juego volverá a empezar, en este caso se refiere a que si sales del camino naranja, la rueda volverá a su posición inicial. En la Ilustración 51 se puede ver la rueda a medio camino.

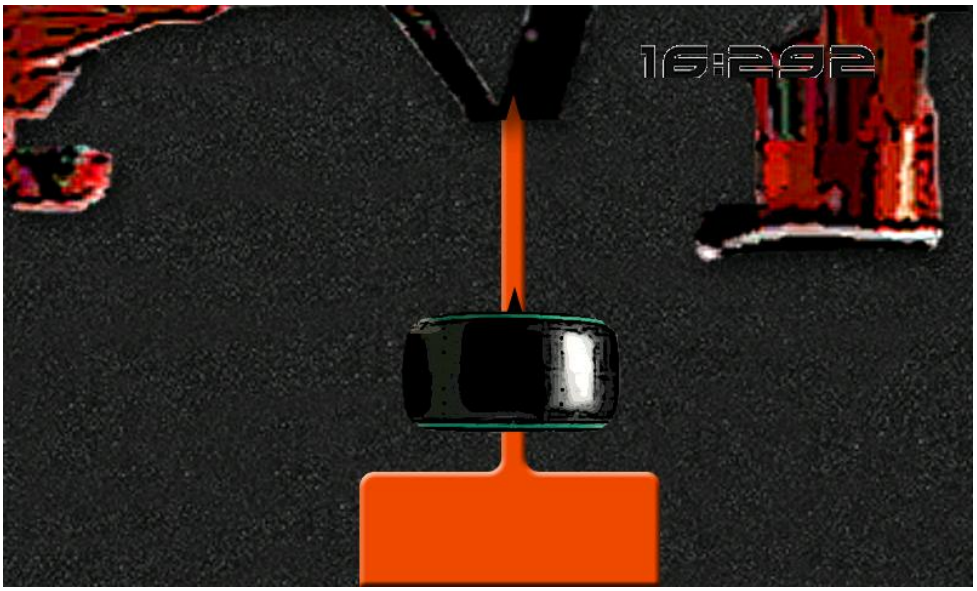


Ilustración 51: Mini juego Por el camino.

El siguiente juego es "Mover tuercas". Se distribuyen por la pantalla dos tuercas numeradas, la tuerca 1 y la 2. Además, también se reparten dos huecos de color naranja con la misma numeración. El juego consiste en mover las tuercas con los dedos, hasta llevarlas a sus respectivos huecos, la tuerca 1 al hueco numero 1, y la 2 al hueco 2. Pero tiene truco, las tuercas solo se moverán si las mueves las dos a la vez. No puedes mover primero una de las tuercas y luego la otra, es necesario mantener como mínimo dos dedos en la pantalla, de lo contrario las tuercas no se moverán. La Ilustración 52 muestra el escenario de este juego.

Ilustración 52: Mini Juego Mover Tuercas.

El ultimo mini juego se llama “Pulsa Rápido”. Aparecerá en la pantalla un gran botón rojo a la izquierda de la pantalla y una barra vacía a la derecha. El juego consiste en pulsar lo más rápido que se pueda el botón rojo, y con cada pulsación se llenara una poco de la barra contigua, cuando la barra se llene completamente el juego terminara. La barra se vacía a un ritmo constante, así que si dejas de pulsar el botón, o lo hacen demasiado lento, la barra se vaciara y no conseguirás superar el mini juego. El la Ilustración 53 se puede ver es escenario del mini juego con la barra a medio llenar.

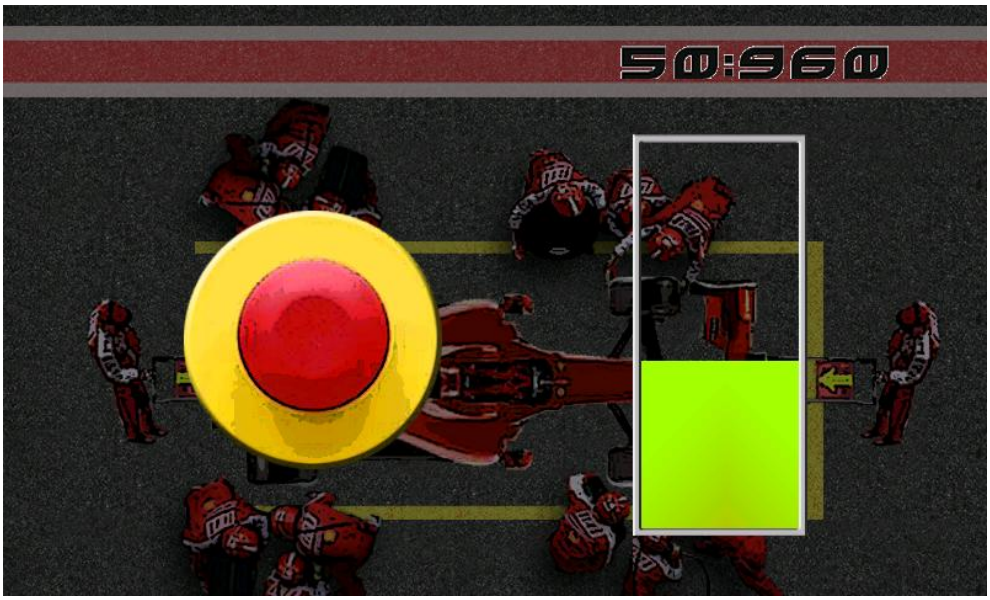


Ilustración 53: Mini juego Pulsa rápido.

Cuando termines todos los mini juegos, automáticamente aparecerá la pantalla final. En ella se te muestra el tiempo conseguido, y te da las opciones de reintentar, que hace que el juego vuelva a empezar al instante y verías como llega el coche de nuevo a la parada de boxes y comienza todo de nuevo, y la opción de volver al menú, esta opción como su nombre indica te lleva de nuevo al menú principal, para poder hacer todas las cosas descritas anteriormente. La apariencia de esta pantalla se muestra en la Ilustración 54.

Ilustración 54: Pantalla Final.

13.2 Guía de instalación para el Usuario.

En la elección tecnológica se especifico que se iba a usar DropBox como sistema de almacenamiento. Este sistema cuenta con una categoría de archivos públicos donde se pueden subir archivos para compartirlos con cualquier persona, tenga o no tenga DropBox.

Desde aquí el usuario podrá descargarse el archivo binario y podrá instalarlo en cualquier dispositivo con Android, ya sea smartphone, tablet o un netbook. Este archivo, necesario para instalar el juego, se encuentra en la siguiente dirección web: https://dl.dropbox.com/u/77286529/F1_Pit_Stop.apk . Al acceder a esta dirección entraras directamente en la descarga del archivo. Este archivo es de extensión .apk, que es la extensión de los archivos binarios de Android, y su nombre completo es F1_Pit_Stop.apk.

Uno de los requisitos del proyecto era que no tuviera que depender de ningún otro software para instalarse, así que no hay que instalar nada para hacer que funcione la aplicación. El único requisito previo a la instalación es que debe de habilitarse la instalación de aplicaciones que no estén en el Android Market de Google (Fuentes Desconocidas). Para habilitar esto nos dirigimos a Ajustes -> Seguridad, en esta sección deberemos activar la opción que pone Fuentes Desconocidas, como muestra la Ilustración 55.

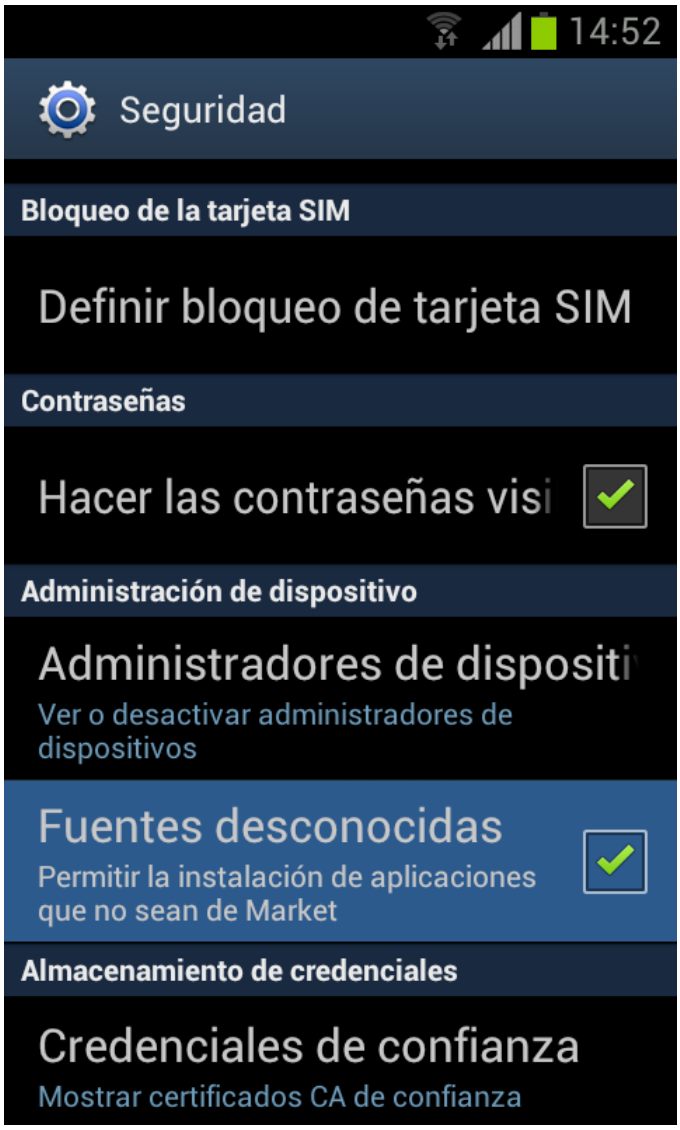


Ilustración 55: Habilitar Fuentes Desconocidas.

Una vez hecho esto, y con el archivo binario en el teléfono, basta con iniciar el archivo para comenzar la instalación y disfrutar de la aplicación.

14. Agradecimientos

En la consecución de este proyecto han contribuido varias personas, ya sea corrigiendo, aconsejando, apoyando o animándome cuando más lo he necesitado.

Lo primero es dar las gracias a mi director de proyecto Germán Rigau Claramunt y al departamento de Lenguajes y Sistemas Informáticos por estar siempre dispuestos a ayudarme con las dificultades y las dudas que me surgieron, y por guiarme en el proceso de desarrollo del proyecto.

Gracias a mi familia, a mis padres, por que sin ellos yo nunca podría haber llegado aquí, siempre me han animado a seguir adelante y no desistir en mi trabajo. Y a mi hermana Sara, por estar a mi lado siempre que ha podido, por el cariño y el afecto que me ha mostrado durante todos estos años. Entre los tres habéis hecho lo posible para hacerme las cosas más fáciles y poder conseguir mi objetivo, gracias.

Gracias a mis compañeros Adrian, Tania, David, Fernando, Asier, Gorka, David I., Diego, Ion y Daniel, con los que he compartido aprobados y suspensos, y que tantos momentos hemos pasado. Entre todos siempre nos hemos ayudado con nuevas ideas, apuntes incompletos o problemas imposibles, pero lo mejor de todo es haber pasado estos tres años con vosotros.

Y por último, todo juego necesita ideas y mi mente no es la más creativa, así que gracias a Adrian R., Igor y Patricia por aportar esas ideas que han hecho que el juego tenga el aspecto que tiene actualmente.

A todos vosotros, gracias.

15. Biografía

En este capítulo se cogen las fuentes de información utilizadas a los largo de todo el proceso de creación del proyecto:

Android:

- Desarrollo de Juegos Android
  Por Mario Zechner

- Android, guía para desarrolladores.
  Por Frank Ableson, Charlie Collins y Robi Sen

- Guía de Instalación de la API de Android en Eclipse.
  http://developer.android.com/sdk/index.html

- Documentación de la API de Android.
  http://developer.android.com/sdk/android-2.1.html

- Tutorial básico sobre juegos en Android.
  http://knol.google.com/k/juan-de-dios-maldonado-s%C3%A1nchez/gu%C3%ADatutorial-para-programar-juegos-2d/yf7xkfmg7vie/3#

- Blog de desarrollo de aplicaciones sobre Android.
  http://droideando.blogspot.com/

Documentación:

- Wikipendia
  www.wikipedia.org