Guess the Pic, 4Kids

Juego educativo para Android

Proyecto Fin de Carrera

Ingeniería Informática

30 de junio de 2014

Tania Bergado San Vicente

Director:
German Rigau i Claramunt

Resumen

Las nuevas tecnologías están a la orden del día. Cualquier persona, sea de la edad que sea, dispone de un terminal móvil. Además, en los últimos años, se han desarrollado dispositivos cada vez más sofisticados y con ello un gran número de aplicaciones móviles que pueden tener como finalidad ayudarnos en tareas de la vida cotidiana, enseñarnos o simplemente entretenernos. ¿Quién no espera a su tren jugando con el móvil? ¿Quien no está en la consulta del dentista y ameniza la espera con las redes sociales desde su móvil?

Es por ello, que en este proyecto se ha desarrollado una aplicación móvil para el sistema operativo en auge, Android. Se trata de un juego educativo para niños en el que se les mostrará una imagen y deberán adivinar de que se trata. De este modo, podrán adquirir y ampliar sus conocimientos sobre los números, los colores, las profesiones, las partes del cuerpo, etc, mientras se divierten. Al estar destinado para niños en proceso de aprendizaje de escritura y lectura, con este juego, mejorarán estas habilidades. Además podrán practicar idiomas ya que el juego está disponible en castellano, inglés y euskera.

Agradecimientos

En primer lugar, gracias a mi director del proyecto, German Rigau y al departamento al que pertenece, Lenguages y Sistemas Informáticos. Gracias German, por tu ayuda y tus consejos.

Gracias a mi familia y en especial a mis padres, por estar a mi lado y por haberme apoyado en cada una de las decisiones que he tomado a lo largo de mi vida.

Gracias a mis amigas de toda la vida porque si estoy aquí, en parte, también es gracias al recorrido que he hecho a su lado. Sin dudarlo, también darle las gracias a dos personas muy importantes, Ana y Adri. Dos personas que he tenido la suerte de conocer cuando empecé mis estudios en San Sebastián y que me han enseñado el significado de la amistad incondicional.

Gracias también a mis compañeros de clase que han compartido estos años de carrera conmigo. Gracias por ayudarme en momentos en los que lo he necesitado, sobretodo a aquellos con los que empecé la Ingeniería Técnica en Informática de Sistemas.

No puedo olvidarme de aquellos que este año han permitido que viva y han querido compatir conmigo la mejor experiencia de mi vida, gracias Leuven.

Gracias a Fernando, porque si no fuera por él, hoy no estaría escribiendo estos agradecimientos. Gracias por todo lo que me has enseñado y me enseñas cada día, tanto en lo personal como en los estudios. Gracias por tu paciencia conmigo, por tus ganas de ayudarme constantemente y por tu apoyo ilimitado. Simplemente, gracias por compartir tu vida conmigo.

Y por último, a Miren y Rafa por las traducciones al euskera, a Asier, Gorka, Juan y de nuevo a Fernando, por estar dispuestos a probar mi juego, al Club de Tenis Miranda por prestarse a colaborar en las pruebas realizadas con niños, y en general, gracias a todos aquellos que han contribuido a la finalización de este proyecto.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Este documento es la memoria del Proyecto de Fin de Carrera de la titulación Ingeniería Informática, realizado por Tania Bergado San Vicente bajo la dirección de German Rigau i Claramunt durante el segundo cuatrimestre del curso académico 2013–2014.

Es evidente que la tecnología móvil ha avanzado de una manera espectacular esta última década. Si intentamos hacer memoria, recordaremos los móviles de los años 90, sus grandes dimensiones y peso, con su antena exterior y el antiguo juego del *"snake"*[1]. Parecía imposible pensar que algún día podríamos conectarnos a Internet, escuchar música, utilizar un móvil como cámara de fotos y mucho menos poder chatear y estar en contacto continuo con los nuestros.

Hoy en día todo esto es lo común, y prácticamente está al alcance de todo el mundo. Pasadas las novedades de tener un móvil que reprodujera mp3 o sacara fotografías, ahora comprarse un terminal implica pensar en los Gigabytes que tiene o los megapíxeles de la cámara de fotos.

Los artífices de que se haya producido esta revolución son las compañías móviles cuyos dispositivos inteligentes utilizan los diferentes sistemas operativos existentes. Estas empresas son Apple con iOS, Google con Android, Microsoft con Windows Phone, Nokia con Symbian y Blackberry con BlackBerry OS.

El sistema operativo Android esta cobrando cada día mayor importancia en el mercado de la telefonía móvil, debido a que cada día más fabricantes lo

[1] `http://es.wikipedia.org/wiki/Snake_(videojuego)`

usan en sus dispositivos, desde teléfonos móviles hasta tablets. Sin duda es la plataforma móvil más extendida en el mercado actual.

Las aplicaciones móviles también han sufrido algunos cambios. Básicamente, los móviles se usaban para llamar, enviar mensajes de texto y jugar a cuatro juegos básicos. Gracias a la evolución del hardware, el rango de uso se ha ampliado a varios campos, como el audiovisual, la navegación por Internet o las redes sociales. La mezcla de estos usos ha generado una demanda de aplicaciones que permitan realizar estas acciones de manera sencilla e intuitiva haciendo que los móviles sacien las mismas necesidades que los ordenadores de toda la vida.

Desde que realicé el Proyecto de Fin de Carrera de la Ingeniería Técnica en Informática de sistemas, hace ahora dos años, me quedé con las ganas de realizar una aplicación para móviles. Sabiendo lo anteriormente mencionado, está claro que la telefonía móvil está marcando el presente y sobretodo lo hará en el futuro por lo que no he encuentro una mejor manera de acabar mis estudios superiores que haciendo una aplicación móvil.

La verdad es que siempre tuve claro que quería que mi aplicación a desarrollar fuera en el sistema operativo Android. Tenía varios motivos para ello, el primero y determinante es que no tengo dispositivos que utilicen otro sistema operativo diferente; Android cuenta con el respaldo de Google, una de las empresas más grandes y estables a nivel mundial; numerosos fabricantes han producido sus nuevos modelos con Android como sistema nativo; el crecimiento de ventas de dispositivos con este sistema operativo y sus aplicaciones; el lenguaje de programación utilizado es Java, el lenguaje más conocido por mí por lo que no conlleva una formación adicional; existe un plugin que permite el desarrollo de la aplicación en el entorno de programación Eclipse, habiendo trabajado durante toda la carrera en él; y la principal, Android se lanzó bajo la licencia Apache, una licencia libre y de código abierto, lo que implica que cualquier desarrollador tiene acceso al kit de desarrollo de software.

Una vez escogido el sistema operativo, tenía que decidir que tipo de aplicación quería realizar. Al comienzo pensé en hacer una aplicación que ayudara a su usuario en alguna de sus tareas de la vida cotidiana. Por mucho que pensaba y se me ocurrían ideas que en un principio me parecían brillantes finalmente no resultaban así. Era cuestión de tiempo investigar en el Google Play y darme cuenta de que todas ellas existían o sino, eran de características similares. Es aquí, cuándo empecé a preguntarme si está ya todo inventado aunque puedo suponer, que las empresas, sobretodo Apple y Google, se esfuerzan por hacer ver que no todo lo está y que cada vez que presenten sus nuevos dispositivos nos seguirán dejando con la boca abierta.

Figura 1.1: Las nuevas tecnologías y los niños van de la mano

En lo siguiente que pensé fue en desarrollar un juego inexistente y que estuviera destinado para niños. El motivo de esto es que siempre he tenido debilidad por este grupo de corta edad. Sin duda, las nuevas generaciones se están criando con la tecnología y un niño de cuatro años, que apenas sabe escribir o leer, maneja cualquier dispositivo casi mejor que sus padres. Un ejemplo de esto se ve en la figura 1.1. El juego debía tratar temas educativos ya que iba dedicado a los niños.

Por último ya sólo me quedaba decidir qué juego hacer. Después de unos cuantos días dándole vueltas se me ocurrió que una buena forma de enseñarles las competencias básicas de su edad podría ser mostrándoles imágenes y que ellos tuvieran que adivinar de que se trataba.

Para ello me puse en contacto con una amiga licenciada en Magisterio y discutimos sobre que temas serían convenientes tratar. Finalmente decidí que estos serían los que aparecerían en el juego:

- Animales
- Números y colores
- Alimentos
- Meses, días de la semana y estaciones del año
- Ropa y partes del cuerpo

- Medios de transporte
- Profesiones
- Escenas de la vida cotidiana

Por lo tanto, ya tenía mi juego. Consistiría en una serie de niveles los cuales tratarían cada uno de los temas mencionados arriba, en los que el usuario debería adivinar a qué corresponde cada una de las imágenes que se le muestran. Un ejemplo de ello, se puede observar en la figura 1.2.

Figura 1.2: Pantalla principal del juego

La gran diferencia de esta aplicación con las ya existentes, es que este juego está destinado a un grupo de usuarios reducido, a los niños, cosa que ninguna de las anteriores hacía. Para hacerlo sencillo, se utilizan temas adecuados para sus competencias e imágenes que no dan lugar a confusión y que son lo más animadas posible.

Otra cosa que aporta el nuevo juego es la posibilidad de seleccionar nivel. El juego se agrupa en ocho niveles diferentes con su temática y el niño podrá seleccionar aquel nivel que desee, como en la figura 1.3.

Figura 1.3: Cuatro primeros niveles de Guess the Pic

Dentro de cada nivel, hay subniveles. El jugador sólo podrá seleccionar el primero de ellos y a medida que los vaya superando, se irán desbloqueando el resto, hecho que se puede observar en la figura 1.4, en la que aparecen bloqueados los subniveles a partir del 11. Una vez se hayan desbloqueado los 20 subniveles, se bloquearán de nuevo.

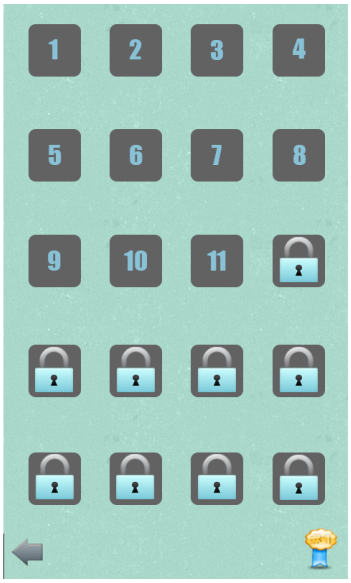


Figura 1.4: Pantalla de los subniveles hasta el 11 bloqueados

1.1. Organización del documento

El presente documento está estructurado de la siguiente forma:

- **Capítulo 1:** Se comienza con una pequeña introducción del proyecto, la motivación que ha llevado al desarrollo del mismo y por último, se explica cómo se va a estructurar este documento.
- **Capítulo 2:** El siguiente capítulo es el documento de objetivos del proyecto, en el cual se describe la planificación confeccionada al inicio del mismo. Incluye los objetivos del proyecto, el alcance del mismo, el método de trabajo a seguir a la hora de la realización del proyecto, la planificación temporal, de calidad y comunicaciones así como el plan de contingencia. Para finalizar se estudia la factibilidad de la realización del proyecto.
- **Capítulo 3:** Este capítulo, introduce el contexto tecnológico en el que se enmarca el proyecto. Sigue con una exposición del trabajo existente hasta el momento.
- **Capítulo 4:** Presenta las diferentes tecnologías elegidas para desarrollar tanto el producto como la documentación del mismo.
- **Capítulo 5:** En este capítulo se hace una explicación sobre la arquitectura del sistema Android.

- **Capítulo 6:** A continuación, se encuentra la captura de requisitos, un apartado relativo al desarrollo del sistema en el que se detallan los casos de uso planteados y el modelo dominio.
- **Capítulo 7:** Seguidamente aparece el análisis del producto en el que se da una primera aproximación del resultado final del juego explicando para ello cada caso de uso.
- **Capítulo 8:** Después, se explica el diseño del sistema con detalles de la implementación, para lo cual se incluye pseudocódigo del juego.
- **Capítulo 9:** 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 muestra el diagrama de todas las clases implementadas.
- **Capítulo 10:** Se detallan todas las pruebas llevadas a cabo a lo largo del desarrollo del producto para cerciorarnos de su correcto funcionamiento.
- **Capítulo 11:** A continuación se explica en que dispositivos y de que manera se ha llevado a cabo la implantación de la aplicación y cuales han sido sus resultados.
- **Capitulo 12:** Se trata del seguimiento y control del proyecto que detalla y reflexiona sobre los cambios acontecidos con respecto al plan inicial del documento de objetivos del proyecto.
- **Capitulo 13:** Seguidamente se hace una breve valoración final del proyecto que incluye los objetivos conseguidos y una valoración personal. Además, se enumeran algunas ideas y líneas de mejora que podrían implementarse para seguir trabajando en el sistema.
- **Capitulo 14:** Se incluye un apartado con las referencias a los libros y páginas webs de las que se ha obtenido información.
- **Apéndices:** Por último se ha incluido como apéndices un manual para el uso de la aplicación desarrollada y los checklist utilizados a la hora de hacer las pruebas a los niños.

Capítulo 2

Documento de objetivos del Proyecto (DOP)

En este capítulo se detalla la planificación elaborada al inicio del proyecto. Se comienza enumerando los objetivos, tanto los directos como los indirectos, para continuar con la definición del alcance del proyecto que incluye de manera detallada la estructura de descomposición del trabajo. A continuación, se muestra el método de trabajo a seguir en el proyecto. Seguidamente aparece la planificación temporal la cual incluye los entregables que generará el proyecto, los hitos principales y las tareas a realizar. También se detallan los planes de calidad, comunicaciones y adquisiciones. Para evitar y/o controlar problemas que surjan a lo largo del desarrollo del proyecto se realiza un plan de contingencia en el que se analizan los riesgos a los que se enfrenta el proyecto, categorizándolos y midiendo su posible impacto. Asimismo, se hace una pequeña descripción de cómo prevenirlos y solucionarlos en caso fallido de su prevención. Para finalizar, se estudia la factibilidad del proyecto.

2.1. Objetivos

El objetivo del proyecto es desarrollar una aplicación para el sistema operativo Android. En concreto, se trata de implementar un juego educativo para niños de 6 años en adelante cuyo idioma nativo sea el castellano. El juego se realizará en dos dimensiones, constará de un menú principal que dará la opción de jugar, configurar el sonido del juego y poder acceder a una breve ayuda de cómo jugar. El juego estará disponible en tres idiomas diferentes, castellano, inglés y euskera.

Se trata de un juego en el que al usuario se le mostrará una imagen de una temática concreta y deberá escribir el nombre correspondiente de lo que se muestra en dicha imagen. A medida que avance, la complejidad de las imágenes y el texto a escribir será mayor. La aplicación constará de un total de ocho niveles divididos en 20 subniveles diferentes por cada nivel. En cada nivel el usuario recibirá una puntuación acorde al tiempo tardado y a los intentos necesarios para escribir correctamente la palabra.

La estructura y el código deberán ser lo más flexibles y escalables posible, con el fin de permitir futuras actualizaciones del juego. Además la interfaz del juego deberá ser lo más sencilla e intuitiva posible debido a la edad de sus futuros usuarios.

A partir de este objetivo se derivan otros indirectos, que son los siguientes:

- **Familiarización con las aplicaciones móviles.** Es evidente que el mercado de las aplicaciones móviles está en auge y a la orden del día por lo que su familiarización supone un paso hacia adelante en el mundo de la informática.

- **Adquisición de conocimientos de la herramienta SDK de Android.** El desarrollo de aplicaciones para Android se hace habitualmente con el lenguaje de programación Java y el conjunto de herramientas de desarrollo SDK por lo que para poder implementar el juego será necesario estudiar esta herramienta previamente.

- **Utilización de Git como herramienta de control de versiones.** Hoy en día es una herramienta fundamental en cualquier proyecto de desarrollo por lo que conviene adquirir conocimiento sobre ello lo antes posible.

- **Conseguir mayor experiencia en los procedimientos para el desarrollo de software.** No es la primera que se lleva a cabo un trabajo de estas características por lo que este proyecto servirá para progresar en el desarrollo de los mismos.

- **Lograr el aprendizaje en la edición de documentos con LaTeX.** Debido a que permite centrarse exclusivamente en el contenido, sin tener que preocuparse de los detalles del formato se trabajará por primera vez con este sistema de composición de textos, lo que implicará su aprendizaje de una manera indirecta.

2.2. Alcance

A continuación se marca el alcance mínimo del sistema que se plantea, las posibles líneas de ampliación que se prevén, los aspectos que en todo caso quedan fuera del alcance del proyecto y por último la estructura de descomposición del trabajo (EDT) de este proyecto.

2.2.1. Mínimo

El juego a desarrollar deberá contar, al menos, con las siguientes funcionalidades:

- Tres idiomas diferentes; castellano, euskera e inglés.
- Ocho niveles y 160 subniveles en total.
- Almacenamiento de las puntuaciones de cada nivel en el dispositivo del usuario, es decir, de manera offline.
- En el menú principal el usuario podrá configurar el sonido del juego y/o acceder a una breve explicación del objetivo del juego y cómo jugar.
- Al comienzo del juego, el jugador sólo podrá acceder al subnivel uno de cada nivel, apareciendo el resto bloqueados.
- Una vez se hayan completado los 20 subniveles, se volverán a bloquear.
- En cada nivel, el usuario podrá acceder a la puntuación total del mismo.
- En cada pantalla, a excepción de la pantalla inicio, el usuario tendrá la opción de volver atrás.
- Por cada subnivel e idioma el usuario tendrá una pista. Si se desea obtener la pista, se le mostrará el número de letras que tiene la palabra a adivinar y alguna letra en la posición que le corresponde.
- Una vez comenzado el juego, el usuario podrá pausarlo. En ese momento aparecerá un menú que le permitirá retomar el juego o volver a la pantalla del menú principal.
- Si el usuario decide salir del juego pulsando el botón de atrás del teclado, se le pedirá confirmación antes de salir.

En cuanto al alcance del proyecto incluye:

- Seguimiento y control del mismo.

- Redacción de un manual para el usuario.
- Formación en la tecnología SDK de Android.

2.2.2. Líneas de ampliación

En caso de que hubiera tiempo suficiente se proponen las siguientes ampliaciones:

- Desarrollar el juego en un cuarto idioma, alemán.
- Añadir un mayor número de niveles y subniveles.
- Además de imágenes, añadir sonidos a cada nivel.
- Almacenamiento de las puntuaciones en un servidor para permitir a los usuarios hacer comparaciones con otros jugadores.

Habría que estudiar el tiempo restante y cuales de las anteriores tareas serían factibles.

2.2.3. Exclusiones

Por lo tanto, queda fuera del alcance del proyecto el siguiente requisito:

- El correcto funcionamiento en aquellos dispositivos que no sean tablets o móviles, además de aquellos cuya versión no esté comprendida entre la 2.2 (Froyo) y la 4.4 (KitKat).

2.2.4. Estructura de descomposición del trabajo

En la figura 2.1 se muestra la EDT del proyecto, en la cual se pueden apreciar los principales componentes o grupos de tareas que se realizarán durante su duración para poder tener así una perspectiva global del mismo y conseguir una buena planificación y organización del proyecto.

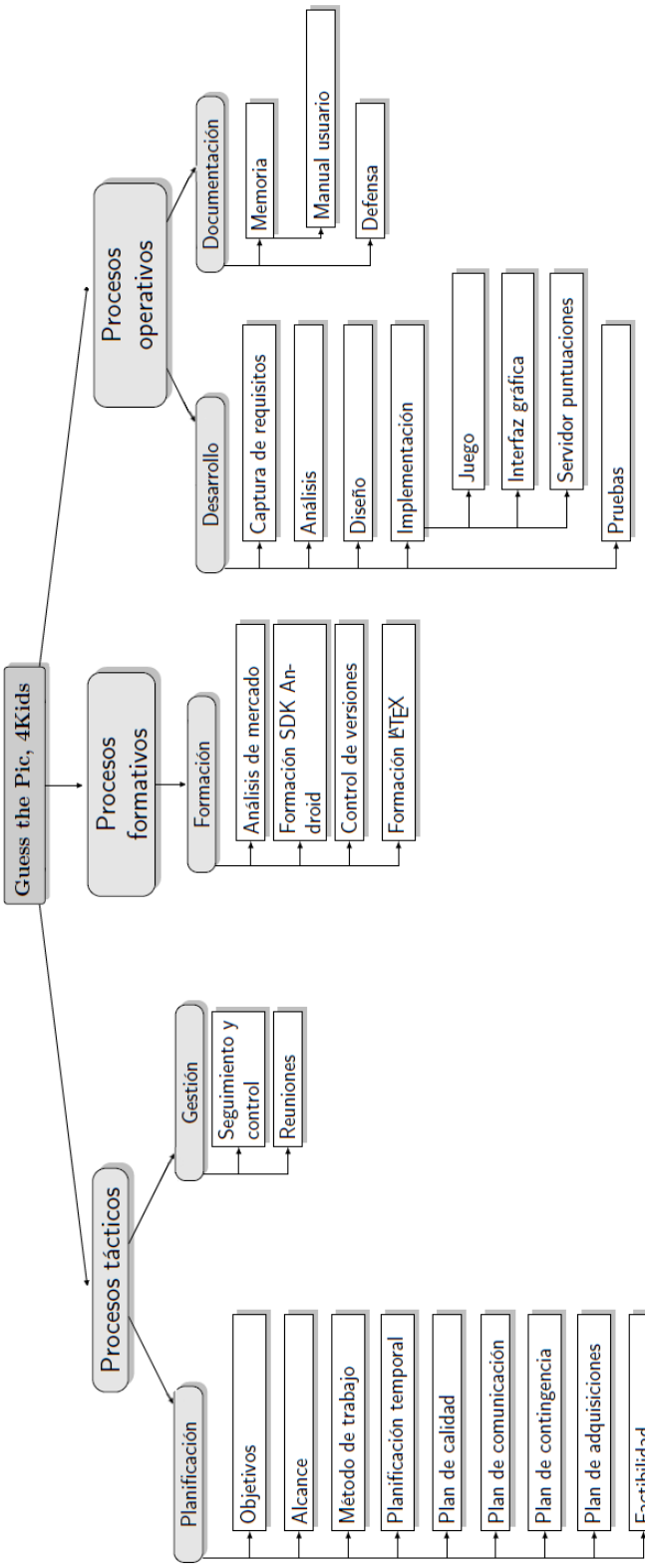


Figura 2.1: Estructura de descomposición del trabajo

2.3. Método de trabajo

Se definen cinco fases para la consecución del proyecto. A las cuales se les atribuyen unos objetivos y un plazo para poder ir controlando el avance del proyecto. Las iteraciones en las que se divide el proyecto son las siguientes:

- **Primera iteración.** Se trata de la fase de planificación del proyecto.

  Esta iteración tendrá lugar durante la primera semana del ciclo de vida del proyecto. En este punto se realiza la primera reunión con el director del proyecto y se plantea una primera idea sobre en qué consistirá el proyecto, sin tener en cuenta detalles sobre el desarrollo. Una vez esté la idea más o menos clara, se procede a desarrollar la planificación del proyecto estableciendo los objetivos, así como su alcance y la estimación temporal del desarrollo.

- **Segunda iteración.** En esta iteración se llevará a cabo la formación de la desarrolladora.

  En primer lugar se realizará un análisis de mercado recabando información sobre aplicaciones en el sistema Android que puedan ayudar al desarrollo del nuevo juego. En esta iteración se decidirán también las tecnologías y herramientas que se utilizarán a lo largo del desarrollo del proyecto. Y por último, será la iteración en la que tendrá mayor peso la formación de la alumna en el control de versiones, en el editor de textos LaTeXy en el SDK de Android.

- **Tercera iteración.** Esta fase consta del desarrollo del proyecto.

  Se trata de la parte más compleja y que más tiempo va a llevar a la hora de realizar el proyecto. Esta tercera iteración consta de cinco partes claramente diferenciadas.

  1. **Captura de requisitos.** En primer lugar se fijan las necesidades y las funcionalidades del juego. Para ello se realizarán todos los casos de uso que comprenden el proyecto y el modelo de dominio.

  2. **Análisis.** Una vez realizada la captura de requisitos hay que centrarse en qué es lo que se quiere conseguir realmente.

  3. **Diseño.** Se define el sistema de información con el suficiente detalle para permitir luego su implementación.

  4. **Implementación.** Tras establecer claramente el diseño y descomponer el trabajo en partes se procede a la codificación de la apli-

cación. Esta fase incluye también la elaboración de la interfaz del juego.

5. **Pruebas.** Después de terminar con la implementación se dedicará tiempo a las comprobaciones y validaciones necesarias del juego. Estas pruebas tendrán dos partes.

   - En primer lugar se realizará una primera evaluación del correcto funcionamiento del juego en diferentes dispositivos Android y con diferentes versiones del sistema operativo.
   - Posteriormente se evaluará la complejidad del juego y de la interfaz. Para ello se realizarán pruebas con niños habiendo elaborado una lista previa de elementos a evaluar en las conductas de estos niños con el juego. Se analizarán sus comportamientos por si se necesitara algún cambio debido al exceso o falta de complejidad.

   Una vez realizadas las pruebas se procederá a efectuar los cambios necesarios tanto a nivel interno como en la interfaz de la aplicación.

- **Cuarta iteración.** Se trata del proceso de documentación.

  En esta fase tendrá lugar la redacción de la memoria del proyecto la cual conllevará un inversión considerable de tiempo. Durante la implementación se irá escribiendo poco a poco parte de la memoria pero cuando mayor tiempo se dedicará a esta actividad será una vez finalizada la implementación del juego por lo que se considera que hasta entonces no comenzará dicha tarea. Esta iteración incluye también la preparación de la defensa del proyecto y la redacción del manual de usuario.

- **Quinta iteración.** Esta última fase consiste en la gestión del proyecto.

  Para garantizar que se va cumpliendo la planificación se realizará un seguimiento y control del proyecto. Dicho seguimiento incluirá el alcance logrado y el tiempo real invertido; para el cual se ha creado un documento excel que permite contabilizar las horas empleadas en cada tarea y las fechas en las que se han llevado a cabo. También se dejará constancia de las desviaciones sufridas de la planificación temporal inicial, las reuniones que hayan tenido lugar entre la desarrolladora y el director con su correspondiente tema principal. Se explicarán también aquellos riesgos que se han materializado así como se abordaron y finalmente todos los cambios realizados que no se contemplaron en el plan inicial.

2.4. Planificación temporal

Debido a la extensión del proyecto, se ha realizado una planificación temporal para su posterior seguimiento durante el desarrollo del mismo. Para llevar a cabo dicha tarea se han identificado los entregables, detallando los hitos clave del proyecto, la estimación del tiempo que deberá de invertirse para completar cada tarea, y el marco temporal en el se planea realizar estas tareas.
Cabe destacar que el proyecto será elaborado en el periodo que abarca desde febrero de 2014 hasta julio de 2014.

2.4.1. Entregables

A continuación se detallan los entregables identificados del proyecto.

- **Memoria.** Documento formal y detallado que contendrá la información relativa al proyecto y lo ocurrido durante el transcurso del mismo.
- **Código fuente.** Se trata del código de la aplicación desarrollada. Estará disponible en el servicio de alojamiento Bitbucket.

2.4.2. Hitos

En la tabla 2.1 se enumera la lista de hitos junto con sus respectivas fechas, todas ellas correspondientes al año 2014.

Hito	Fecha
Finalización de la planificación	21 de febrero
Finalización del prototipo	16 de abril
Decisión de ampliación del alcance inicial	21 de abril
Finalización de la implementación	4 de junio
Finalización de la memoria	13 de junio
Entrega del proyecto	1 de julio
Cierre del proyecto	15 de julio

Tabla 2.1: Hitos

2.4.3. Actividades

Las actividades a realizar para la consecución del alcance mínimo propuesto se distribuyen de la manera en la que se indica en la tabla 2.2.

Tarea	T. estimado
Planificación	**15 h.**
Planificación inicial	10 h.
Replanificaciones	5 h.
Formación	**65 h.**
Análisis de mercado	10 h.
Formación en SDK Android	45 h.
Formación en control de versiones	5 h.
Formación en LaTeX	5 h.
Desarrollo	**190 h.**
Captura de requisitos	10 h.
Análisis	15 h.
Diseño	15 h.
Implementación	130 h.
Pruebas	20 h.
Documentación	**70 h.**
Memoria	40 h.
Manual de usuario	10 h.
Revisión y corrección de errores	10 h.
Defensa	10 h.
Gestión	**20 h.**
Seguimiento y control	15 h.
Reuniones	5 h.
Total	360 h.

Tabla 2.2: Coste temporal estimado para el alcance mínimo

Asimismo, se plantean las tareas asociadas a la línea de ampliación propuesta que se detallan en el tabla 2.3 y que suman un total de 100 horas. Su realización se decidirá tras el análisis de la situación que se realice el día del hito correspondiente (21 de abril).

Tarea	T. estimado
Cuarto idioma: Alemán	20 h.
Más niveles	25 h.
Sonidos	15 h.
Servidor online de puntuaciones	40 h.
Total	100 h.

Tabla 2.3: Coste temporal estimado para la ampliación del alcance

2.4.4. Diagrama de Gantt

En la figura 2.2 se muestra el diagrama de Gantt de este proyecto, mediante el cual se pueden visualizar gráficamente los plazos de dedicación previstos para las diferentes actividades a lo largo de la vida del proyecto.

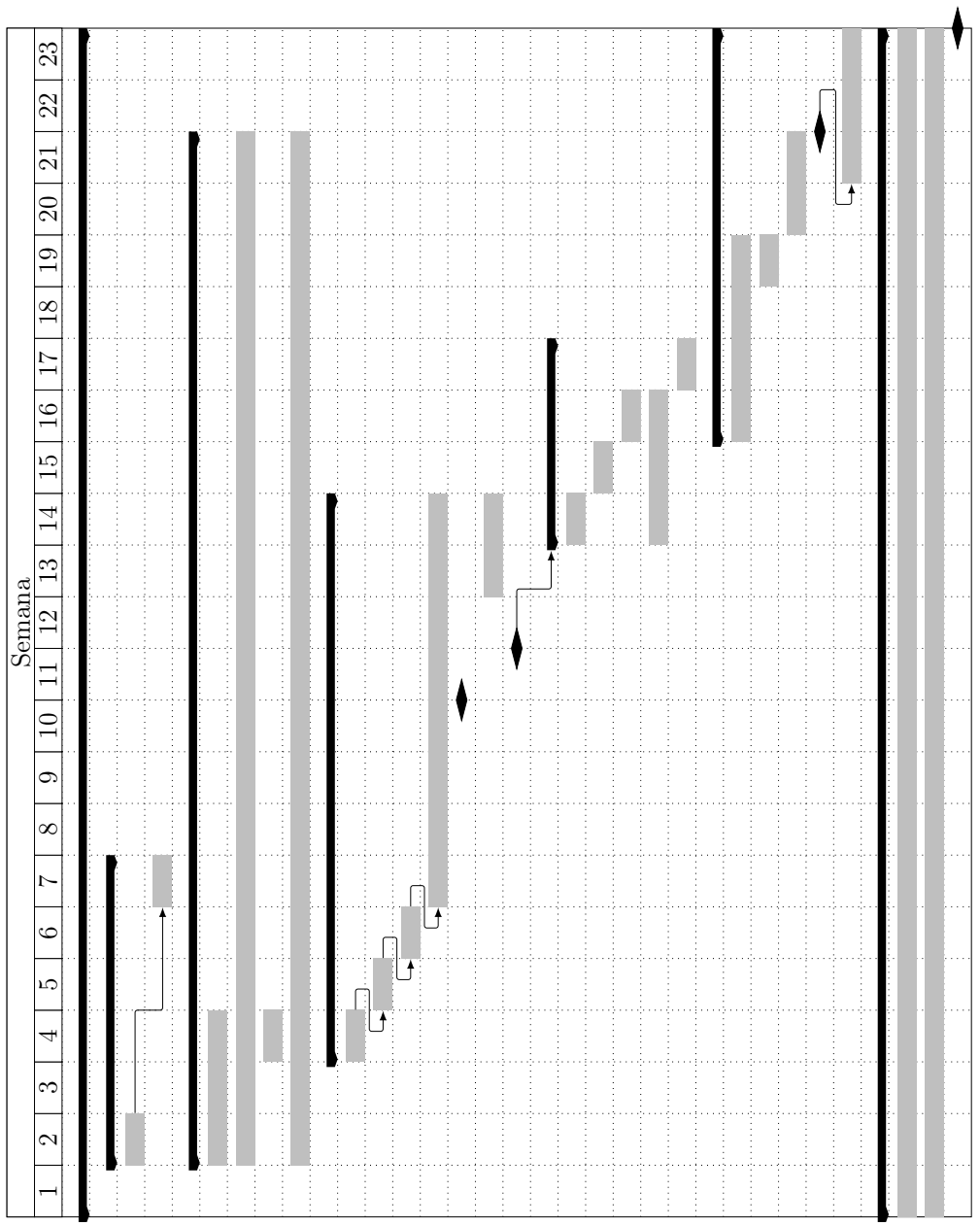


Figura 2.2: Diagrama de Gantt

2.5. Calidad

Para asegurar la calidad de los dos entregables, se plantean dos líneas de actuación:

1. En lo relativo al producto, se ha fijado un hito (el 16 de abril) para la presentación de un prototipo que sirva como muestra de lo que será la aplicación final. Basándose en las críticas recibidas, se elaborará una lista exhaustiva con los detalles y características que debe tener la versión final del juego, además de aquellas funcionalidades que deben ser mejoradas.

2. En cuanto a la documentación, se ha planificado una tarea la cual consiste en la revisión constante en profundidad de la memoria a entregar. Asimismo hay fijado otro hito (el 13 de junio) en el cual se entregará la memoria final y será revisada por el director del proyecto.

2.6. Comunicaciones

Seguidamente se traza el plan de comunicaciones, que consta de la identificación de los interesados en el proyecto, los canales que se prevén para la comunicación entre estos, y la metodología que se utilizará para garantizar el correcto desarrollo del proyecto.

2.6.1. Interesados

Los interesados del proyecto aparecen en la siguiente tabla 2.4:

Nombre	Rol	Correo electrónico
Tania Bergado San Vicente	Desarrolladora/alumna	tpbergado001@ikasle.ehu.es
German Rigau i Claramunt	Director	german.rigau@ehu.es

Tabla 2.4: Interesados en el proyecto

2.6.2. Canales de comunicación

Se establecen dos canales para la comunicación entre las dos personas implicadas en el proyecto:

1. **Conversación en persona.** Es el mejor método de comunicación pero requiere la presencia física de los interlocutores y debido a que la desarrolladora se encuentra en una ciudad distinta a la de la Facultad de Informática de San Sebastián, este método de comunicación será utilizado únicamente cuando sea estrictamente necesario.
2. **Correo electrónico.** Es el método de comunicación por defecto entre los interesados cuando no es posible comunicarse en persona. Se espera una respuesta del receptor del correo aunque esta no tiene que ser inmediata pero sí dentro de un margen de máximo dos días laborables.

Cada canal será utilizado dependiendo de la urgencia, de la necesidad de información y del tipo de la misma.

2.6.3. Métodos de comunicación

Se celebrarán reuniones de seguimiento aproximadamente cada cuatro semanas en las que la desarrolladora informará al director del estado del proyecto y consultará las dudas surgidas, para decidir y dirigir el sentido y los detalles del proyecto.

2.7. Plan de contingencia

En todo proyecto existen una serie de riesgos de mayor o menor gravedad que tienen posibilidad de suceder con mayor o menor probabilidad. A continuación se listan los riesgos a los que se enfrenta el proyecto, así como la clasificación de los mismos según su probabilidad y el impacto que ocasionarían en la consecución satisfactoria de los objetivos del proyecto.

Retrasos

Muchas de las tareas del proyecto tienen dependencias entre ellas y no cumplir los plazos puede generar un retraso en las tareas que dependen de ella.

Probabilidad Alta

Impacto Bajo-medio, dependiendo del retraso acumulado

Prevención Ajustarse al plan establecido tanto como fuese posible y se priorizarán las tareas que tengan dependencias para que se hagan antes para que un retraso en ellas no suponga pasarse de plazo

Solución Replanificar los plazos del proyecto

Planificación inadecuada

Sin duda alguna la planificación en un proyecto es importantísima. Una buena planificación nos permite anticiparnos a lo que hay que hacer y cuando hacerlo, además de mitigar los problemas que pueden surgir a lo largo del proyecto. Sin embargo, no hay ningún mótodo ni forma que ayude a realizar una planificación correcta, únicamente se tiene acceso a los pasos a seguir, por lo que la mayoría de las veces puede que dicha planificación no sea del todo adecuada.

Probabilidad Alta

Impacto Medio

Prevención Tratar de ajustar el esfuerzo a la duración

Solución Replanificar aquello que no haya sido adecuado en el plan original

Requisitos definidos incorrectamente

Debido a una incorrecta captura de requisitos pueden ir arrastrándose los errores hacia las siguientes actividades pudiendo ocasionar una desviación considerable de la idea inicial por lo que provocaría una alteración en el proyecto.

Probabilidad Baja

Impacto Alto

Prevención Tratar de detectar cuanto antes dichos errores, para actuar en consecuencia con el mínimo coste posible

Solución En caso de que la desviación fuese considerable, se redefiniría desde un principio los requisitos, objetivos, etc.

Diseño inadecuado

Un diseño inadecuado implica una implementación inadecuada, lo cual conllevará probablemente al fracaso del producto y por tanto del proyecto.

Probabilidad Media

Impacto Alto

Prevención Asegurar la validez del diseño antes de comenzar la implementación

Solución Redefinir de nuevo el diseño en el menor tiempo posible

No cumplir con el alcance deseado

A medida que avanza el proyecto los cambios en el alcance del producto son más costosos lo que eventualmente impedirá que se puedan realizar.

Probabilidad Baja

Impacto Alto

Prevención Al comienzo será validado por el director del proyecto el alcance propuesto analizando si es factible o no su consecución

Solución Tratar de realizar los cambios al inicio del ciclo de vida del proyecto cuando suponen un menor coste

Pérdida de información

Por diversos motivos se puede producir una pérdida de los datos, tanto de la documentación como del código.

Probabilidad Muy baja

Impacto Muy alto

Prevención Utilización de sistemas para el almacenamiento automático de todo el sistema de información en la nube

Solución Intentar recuperar la mayor cantidad posible de la información perdida

Baja por enfermedad

Tanto la desarrolladora como el director del proyecto pueden causar baja no pudiendo trabajar en el proyecto o no pudiendo dirigir a la primera respectivamente.

Probabilidad Baja

Impacto Medio, dependiendo de los días de baja

Prevención Imposible

Solución En caso de que la baja sea del director se procurará seguir adelante con el proyecto intentando solucionar las dudas que surjan. Por otro lado, en caso de que sea la baja de la desarrolladora, se esperará hasta la vuelta en activo de esta. Se ajustarán las fechas del proyecto según sea necesario

2.8. Recursos externos

A lo largo del ciclo de vida del proyecto será necesario llevar a cabo el uso de una serie de recursos externos.

En lo relativo al producto, de momento, se prevé que únicamente será necesaria el uso de imágenes para los diferentes subniveles, así como las de los pequeños iconos que aparezcan en el juego y las músicas que acompañan cada nivel. Todo ello será adquirido de la web con licencias libres de uso.

Por otro lado, en cuanto al proyecto, se utilizará una plantilla de LaTeXpara proyectos de fin de carrera.

2.9. Factibilidad

La factibilidad del proyecto puede ser estudiada desde tres perspectivas.

En primer lugar se trata del **punto de vista tecnológico** en el que no se ve ningún impedimento para llevar a cabo el proyecto ya que se utilizará software

existente. El mayor problema que se podría encontrar a la hora de realizarlo es la inexperiencia de la desarrollada en programación sobre Android pero debido a la extensa documentación sobre el tema y la posibilidad de poder realizar más aplicaciones para este sistema operativo en un futuro añade motivación para afrontar el proyecto.

En segundo lugar, desde un **punto de vista económico**, la única inversión que se observa a simple vista es el darse de alta como desarrollador en Google Play para poder posteriormente publicar el juego. En caso de ampliación del alcance en el que las puntuaciones se almacenen online, habría que adquirir un hosting. No se trata de grandes inversiones económicas por lo que en esta perspectiva tampoco se encuentra ningún incoveniente.

El último punto de vista puede ser el que mayores problemas cause a la hora de estudiar la factibilidad del proyecto. Se trata del **tiempo necesario** para la realización del mismo. Según la planificación temporal llevada a cabo anteriormente, éste se realizará en el plazo que comprende desde medidados febrero de 2014 hasta principios de julio de 2014. Es decir, se dispone de cuatro meses y medio para el desarrollo del proyecto. Es un tiempo bastante limitado pero la desarrolladora podrá dedicarse a él casi a tiempo completo por lo que en principio no debería existir ningún obstaculo para llevarlo a cabo.

Por lo tanto, analizando los tres puntos de vista se puede concluir que es factible la realización del proyecto.

Capítulo 3

Estado del arte

En este capítulo se presenta Android. Por un lado, se explica de manera breve su historia. Qué es, cómo surgió y sus características principales. Por el otro, se hace un análisis del trabajo ya existente y similar a la aplicación a desarrollar en este proyecto.

3.1. Contexto tecnológico

3.1.1. Android

A continuación, se explica qué es Android, base de todo el proyecto y para finalizar se hace una comparación con el resto de fabricantes móviles y por tanto su competencia.

¿Qué es Android?

Android es una plataforma de software de código abierto[1] que incluye un sistema operativo para dispositivos móviles basado en Linux. Desarrollado por Google y por la Open Handset Alliance.

Esta plataforma permite a los desarrolladores escribir código en Java que se ejecute en móviles mediante las librerías Java desarrolladas por Google. También se pueden escribir aplicaciones en otros lenguajes, como por ejemplo C, para posteriormente ser compiladas en código nativo ARM y ejecutarlas,

[1] `http://source.android.com/source/index.html`

aunque este proceso de desarrollo no está soportado oficialmente por Google. La mayor parte de la plataforma de Android está disponible bajo licencia de software libre de Apache y otras licencias de código abierto.

Breve historia

La historia de Android es bastante corta y reciente. En Julio de 2005, Google adquirió Android Inc, una pequeña startup de California. En esos momentos, la compañía se dedicaba a la creación de software para teléfonos móviles. Una vez en Google, el equipo desarrolló un S.O. basado en Linux para dispositivos móviles. Más adelante, Google adaptó su buscador y sus aplicaciones para su uso en móviles.

En septiembre del 2007, Google tenía varias patentes de aplicaciones sobre el área de la telefonía móvil. El 5 de noviembre del mismo año, se anunció la fundación de la Open Handset Alliance al mismo tiempo que la creación de la plataforma Android. La Open Handset Alliance está formada por un consorcio de 34 compañías de hardware, software y telecomunicaciones, entre las cuales se incluyen Google, HTC, Intel y Motorola entre otras, dedicadas a investigar estándares abiertos para dispositivos móviles.

Siete días más tarde Google lanzaba un entorno de desarrollo con una primera versión del emulador para que cualquier persona pudiera ver cómo iba a ser Android y lo más importante, que los programadores empezasen a desarrollar aplicaciones para Android.

El primer teléfono en el mercado con el sistema operativo Android fue el T-Mobile G1 lanzado el día 22 de octubre de 2008 el cual tenía la versión Android 1.0 preinstalada. Este móvil fue el resultado conjunto de T-Mobile, HTC y Google. Por último, desde el 21 de octubre de 2008, Android está disponible como código abierto. Gracias a esto, cualquiera puede añadir extensiones, nuevas aplicaciones o reemplazar las existentes por otras dentro del dispositivo móvil.

Características

Lo primero a lo que se ha de hacer referencia en Android es que está basado en Linux, es decir, todos los servicios base (gestión de drivers, memoria, seguridad) están basados en el sistema operativo de código abierto.

Dalvik es el nombre de la maquina virtual donde se ejecutan las aplicaciones. Esta, está optimizada para requerir poca memoria y poder usar varias instancias simultáneamente sin que el dispositivo se ralentice. El lenguaje en el que se programa es puramente Java.

El motor de navegación es el Webkit, el mismo que utilizan los Mac o los IPhone. Éste es de código abierto y actúa como base para varias aplicaciones que hay actualmente en el mercado, la más famosa el navegador Safari de Apple que hoy en día se puede encontrar también para Windows.

Android utiliza SQLite para el almacenamiento estructurado de datos. SQLite ya viene incluido en el SDK y se puede acceder plenamente a sus clases. También es posible la utilización de otras bases de datos como Perst o incluso utilizar las clases de almacenamiento de datos de la API de Android sin tener que hacer uso de SQLite.

Otra característica interesante, es el soporte a los formatos más comunes de archivos multimedia, un framework que permite la reutilización de componentes y gráficos optimizados, provenientes de librerías 2D y 3D. Además de aquellos recursos que dependen del terminal como el Bluetooth, 3G, Wifi, cámara y GPS entre otros.

Versiones Android

El sistema operativo de Google no llega a los seis años y en este tiempo ha evolucionado de una manera realmente impresionante. En este apartado se va a repasar las diferentes versiones de Android.

Por alguna razón, las versiones de Android[2] reciben el nombre de postres en inglés. En cada versión el postre elegido empieza por una letra distinta siguiendo un orden alfabético. Las 11 versiones son las siguientes.

- **A: Apple Pie** (v1.0), *[Octubre de 2008]*
  La primera versión incluía una ventana de notificación desplegable, widgets en la pantalla de inicio, la integración de Gmail y el Android Market.
- **B: Banana Bread** (v1.1), *[Febrero de 2009]*
  Esta versión fue dedicada básicamente a reparar errores y a implementar las actualizaciones "over the air", o lo que es lo mismo, de manera inalámbrica, sin tener que conectar el móvil por cable ni para descargar

[2] `http://www.startapp.com/infographics/history-of-android.aspx`

un archivo ni para actualizar mediante un programa y que hasta ese momento ninguna plataforma estaba haciendo.

- **C: Cupcake** (v1.5), *[Abril 2009]*
  Primera gran actualización de Android. Soportaba widgets y carpetas en la pantalla principal.
- **D: Donut** (v1.6), *[Septiembre 2009]*
  Mejora del Market de Android, con soporte nativo para Sprint y Verizon. Soportaba también pantallas táctiles de mayor resolución.
- **E: Éclair** (v2.0/v2.1), *[Octubre de 2009]*
  Mejora del navegador y añade actualizaciones de Google Maps y de la interfaz gráfica. Nació Google Maps Navigation para competir con las unidades de GPS como Tom-Tom.
- **F: Froyo** (v2.2), *[Junio de 2010]*
  Significativamente más rápido debido al compilador Just-In-Time. Soportaba Adobe Flash y Data Tethering.
- **G: Gingerbread** (v2.3), *[Diciembre de 2010]*
  Mejoró la interfaz gráfica, incluyó Comunicación de Campo Cercano lo cual permite a los dispositivos compartir datos sin tener contacto.
- **H: Honeycomb** (v3.0/v3.1/v3.2), *[Febrero de 2011]*
  Desarrollado específicamente para tablets pero no tuvo mucho éxito. Gran mejora del 3D y de la aceleración hardware.
- **I: Ice Cream Sandwich** (v4.0), *[Mayo 2011]*
  Versión refinada de Honeycomd optimizando sus características para los dispositivos móviles.
- **J: Jelly Bean/Gummy Bear** (v4.1/v4.2/v4.3),*[Julio de 2012]*
  Permitía múltiples cuentas de usuarios, notificaciones accionables, "ajustes rápidos" en la barra de notificaciones. Se añadió Google Now y nuevos widgets de bloqueo.
- **K: KitKat** (v4.4), *[Octubre de 2013]*
  Presenta un diseño más elegante y un rendimiento mejorado optimizando la memoria y mejorando la pantalla táctil para que responda más rápido y de forma más precisa. Consta de nuevas funciones como emoticonos en el Teclado de Google, Google Cloud Print que permite imprimir fotos, documentos y páginas web desde el dispositivo y muchas otras funciones[3].

[3] `http://www.android.com/versions/kit-kat-4-4/`

- **L: Lime Pie** (v4.6-5.0), *[Esperada para Junio/Julio de 2014]*

Tipos de dispositivos

El sistema operativo Android se usa en teléfonos inteligentes, ordenadores portátiles, netbooks, tabletas, Google TV, relojes de pulsera, auriculares, etc.

El primer teléfono disponible en el mercado para ejecutar Android fue el HTC Dream, dado a conocer al público el 22 de octubre de 2008. A principios de 2010, Google colaboró con HTC para lanzar su producto estrella en dispositivos Android, el Nexus One. Google continuó la comercialización de la gama Nexus en 2010 con el Samsung Nexus S, en 2011 con el Galaxy Nexus y en 2012 con el Nexus 4 (y las tabletas Nexus 7 y Nexus 10), en 2013 con la segunda generación del nexus 7 con conectividad 4G LTE y el nexus 5 fabricados por LG[4].

Los dispositivos Nexus son utilizados para el desarrollo e implementación de Android, siendo los dispositivos que estrenan las nuevas versiones disponibles.

Android frente a otros sistemas operativos para móviles

Es evidente que en los últimos años ha ido creciendo tanto la venta de dispositivos móviles con sistema operativo Android como la venta de sus aplicaciones.

En la actualidad existen aproximadamente un millón de aplicaciones para Android y se estima que un millón y medio de teléfonos móviles se activan diariamente. En 2013 se llegó a los mil millones de teléfonos inteligentes Android en el mundo y ese mismo año se alcanzó el 92 % en ventas de nuevos smartphones seguido de iOS con un 4.4 %.

En la siguiente figura 3.1, se puede ver una comparación en la cuota de mercado de las diferentes plataformas móviles como Apple iOS, BlackBerry, WindowsMobile y Symbian. La comparación se hace según el número de terminales vendidos hasta el último cuarto del 2013 en el mundo[5].

[4] http://es.wikipedia.org/wiki/Android#Usos_y_dispositivos

[5] http://www.androidcurso.com/index.php/tutoriales-android/31-unidad-1-vision-general-y-entorno-de-desarrollo/98-comparativa-con-otras-plataformas

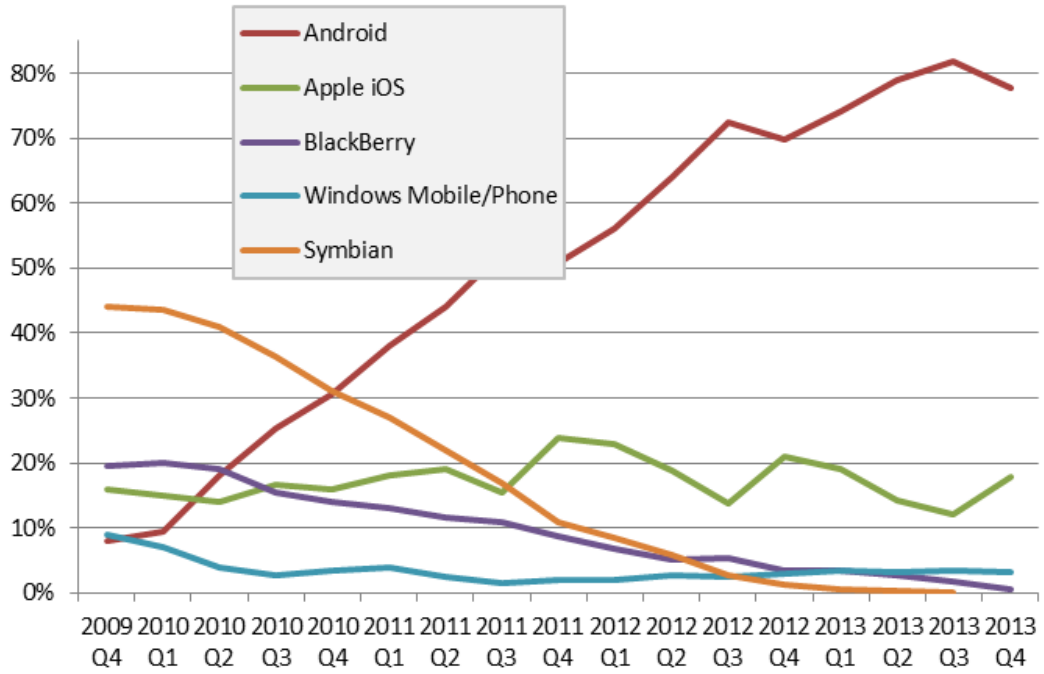


Figura 3.1: Comparación de las ventas de móviles con diferentes SO

Se trata de un estudio realizado por la empresa Gratner Group. La gáfica muestra el importante descenso de ventas de la plataforma Symbian de Nokia, un declive continuo de BlackBerry, y como la plataforma de Windows no parece despegar. Si nos centramos en las dos plataformas más importantes, se observa que Apple tiene afianzada una cuota de mercado en torno al 15 % y el espectacular ascenso de la plataforma Android, que le ha permitido alcanzar en dos años una cuota de mercado superior al 75 %.

3.2. Aplicaciones similares

Como se ha mencionado en el apartado anterior, existen aproximadamente un millón de aplicaciones para Android, por lo que se antoja muy dificil inventar algo que no exista ya.

Investigando en el market de Android, Google Plus, se pueden encontrar alguna aplicación similar a la que se va a desarrollar en este proyecto. A continuación se explica brevemente alguna de ellas y posteriormente se detallará lo que aporta nuestro proyecto y en que se diferencia.

4 Fotos 1 Palabra

En primer lugar, nos encontramos con la aplicación "4 Fotos 1 Palabra"[6]. El objetivo del juego es muy similar al que nosotros buscamos.

El juego muestra cuatro imágenes y el jugador tiene que adivinar la palabra que tienen en común. Por ejemplo, se muestra unas imágenes con césped y una rama, la palabrá en común será verde. Del mismo modo, si se muestra imágenes de lluvia y un lago la palabra en común será agua.

En la siguiente figura 3.2 podemos observar una serie de imágenes que nos muestran una perspectiva general del juego.

Figura 3.2: Imágenes del juego "4 Fotos 1 Palabra"

Adivina la palabra - Foto Quiz

Otro juego de similares características al nuestro es “Adivina la palabra Foto Quiz”[7].

De nuevo, el juego muestra cuatro imágenes y el jugador tiene que adivinar la palabra que tienen en común. Es decir, lo único que cambia respecto al

[6] https://play.google.com/store/apps/details?id=four.pics.one.word

[7] https://play.google.com/store/apps/details?id=com.tapsarena.guess

primero es su interfaz.

En la siguiente figura 3.3 podemos ver dicha interfaz.

Figura 3.3: Imágenes del juego .ªdivina la palabra"

Guess the football club / Guess the capital

Existen también juegos en los que tienes que adivinar una temática concreta, como es el caso de estos dos.

En el primero de ellos[8], se muestran escudos de equipos de fútbol y hay que adiviar a cuál corresponde, como se puede observar en la figura 3.4. En el otro[9], se trata de adivinar a qué capital de un país corresponde la fotografía. En lafigura 3.5 se muestra un ejemplo.

[8] `http://gamelikeapps.com/android/com.gamelikeapps.footballclubs`
[9] `http://gamelikeapps.com/android/com.gamelikeapps.guesscity`

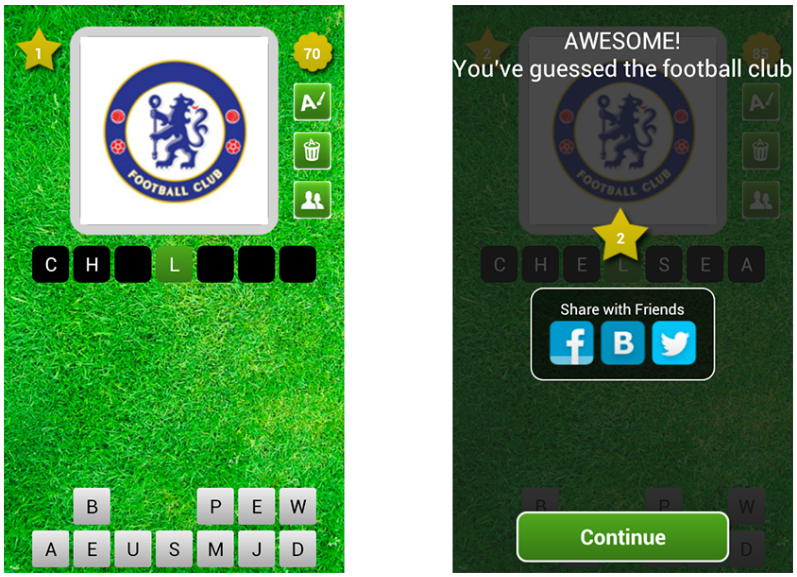


Figura 3.4: Imágenes del juego "Guess the football club"

Figura 3.5: Imágenes del juego "Guess the capital"

Capítulo 4

Elección tecnológica

En este apartado se enumeran y describen los lenguajes de programación, programas y herramientas que se han utilizado para llevar a cabo el proyecto. Además, se incluye una reflexión y justificación de los motivos que llevaron a la elección de cada tecnología.

4.1. Desarrollo del producto

4.1.1. Java

Java[1] es un lenguaje de programación orientado a objetos originalmente desarrollado por Sun Microsystems, la cual fue adquirida por la compañía Oracle. Además, se trata de un lenguaje compilado e interpretado. Es decir, todo programa en Java ha de compilarse y el código que se genera es interpretado por una máquina virtual. De este modo se consigue la independencia de la máquina. El código compilado se ejecuta en máquinas virtuales que si son dependientes de la plataforma.

[1] http://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)

Para desarrollar aplicaciones Android existen dos lenguajes de programación: java y C++. Se eligió el primero porque es la opción más elegida por los programadores en Android y porque además es el lenguaje más conocido y familiar para la desarrolladora.

4.1.2. Eclipse

Eclipse[2] es un entorno de desarrollo integrado (IDE), de código abierto y multiplataforma. Es una potente y completa plataforma de programación, desarrollo y compilación de elementos tan variados como sitios web, programas en C++ o aplicaciones Java.

No se dudó en escoger este entorno debido al conocimiento que la desarrolladora poseía de él.

4.1.3. SDK de Android

Un kit de desarrollo de software o SDK[3] [4] (siglas en inglés de *software development kit*) es un conjunto de herramientas de desarrollo de software que le permite al programador crear aplicaciones para un sistema concreto, por ejemplo ciertos paquetes de software, frameworks, plataformas de hardware, computadoras, videoconsolas, sistemas operativos, etc. Es algo tan sencillo como una interfaz de programación de aplicaciones o API (*application programing interface*) creada para permitir el uso de cierto lenguaje de programación, o puede también, incluir hardware sofisticado para comunicarse con un determinado sistema embebido.

El SDK de Android, incluye un conjunto de herramientas de desarrollo. Comprende

[2] http://www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado/

[3] http://es.wikipedia.org/wiki/Kit_de_desarrollo_de_software

[4] http://es.wikipedia.org/wiki/Desarrollo_de_programas_para_Android

1. Un depurador de código,
2. Biblioteca,
3. Un simulador de teléfono basado en QEMU,
4. Documentación,
5. Ejemplos de código y tutoriales.

La plataforma integral de desarrollo soportada oficialmente es Eclipse, junto con el complemento ADT (*Android Development Tools plugin*).

Por lo tanto, el motivo por lo que se escogió este SDK es porque está diseñado para Eclipse y se tenía claro que se quería trabajar con este IDE.

4.1.4. Adobe Photoshop CC

Adobe Photoshop[5] [6] es un editor de gráficos rasterizados desarrollado por Adobe Systems principalmente usado para el retoque de fotografías y gráficos. Se trata del líder mundial del mercado de las aplicaciones de edición de imágenes y domina totalmente este sector.

En 2013, Adobe anunció una nueva versión de Photoshop, Photoshop CC cuyas siglas provienen del inglés *Creative Cloud*. Esta nueva versión tiene como característica el contenido 3D.

Se decidió utilizar esta herramienta para crear todo el diseño gráfico del juego. La principal razón para su elección es la amplia gama de funcionalidades que ofrece, considerada por la desarrolladora el mejor software de edición de imagen que existe.

[5] http://es.wikipedia.org/wiki/Adobe_Photoshop
[6] http://en.wikipedia.org/wiki/Adobe_Photoshop#CC

4.2. Control de versiones

Se llama control de versiones a la gestión de los diversos cambios que se realizan sobre los elementos de un producto. Aunque esta tarea puede realizarse de forma manual, es interesante el uso de sistemas de control de versiones (SVC), que son las herramientas que facilitan esta gestión almacenando los elementos y guardando un registro histórico de los cambios efectuados para poder, si fuese necesario, revertir los cambios realizados, desarrollar diferentes ramas del producto en paralelo o tener que llevar a cabo la recuperación parcial o total del código debido a la pérdida del mismo.

4.2.1. git

Git[7] es un software de control de versiones diseñado por Linus Torvalds que pone especial atención en el soporte a desarrollos no lineales. A diferencia de otros sistemas, la creación de nuevas ramas y su reunificación son operaciones muy ágiles, con lo que se promueve su uso.

La elección de Git frente a otros sistemas de control de versiones estuvo condicionada por la gran popularidad de Git y por recomendación de un compañero.

4.2.2. Bitbucket

Bitbucket[8] es un servicio de alojamiento basado en web para proyectos que utilizan como sistema de control de versiones Mercurial o Git.

Se escogió Bitbucket frente a GitHub por el hecho de que este último sólo permite de forma gratuita, crear repositorios públicos y visibles por todo el mundo, mientras que Bitbucket si permite la creación de repositorios privados.

[7] http://git-scm.com/
[8] https://bitbucket.org/

4.2.3. SourceTree

SourceTree[9] es un poderoso cliente de escritorio de Git y Mercurial para los desarrolladores de software. Pertenece a la compañía de software con sede en Australia, Atlassian. Posee una interfaz muy simple para gestionar todos los repositorios, alojados o locales evitando de esta manera al desarrollador tener que utilizar la línea de comandos. SourceTree simplifica la forma de interactuar con repositorios Git para poder centrarse únicamente en la codificación y permite mantener los repositorios más limpios con su interfaz intuitiva.

Además es muy fácil ver los cambios realizados en el código así como visualizar las diferentes ramas de desarrollo. Permite también commitear los cambios, descartarlos o combinarlos.

Se ha elegido esta aplicación para evitar el uso de la línea de comandos a la hora de gestionar las versiones del código. El motivo por el que se ha usado este cliente es porque es el único conocido por la desarrolladora.

4.3. Sistema de almacenamiento

4.3.1. Dropbox

Dropbox[10] es un servicio de alojamiento de archivos multiplataforma en la nube. El servicio "cliente de Dropbox" permite a los usuarios almacenar cualquier archivo en una carpeta designada. Ese archivo es sincronizado en la nube y en todos los demás dispositivos del cliente. Los archivos en la carpeta de Dropbox pueden entonces ser compartidos con otros usuarios de Dropbox, ser accedidos desde la página Web de Dropbox o bien ser consultados desde el enlace de descarga directa.

Asimismo, posee soporte para historial de revisiones, de forma que los archivos borrados de la carpeta de Dropbox pueden ser recuperados desde cual-

[9] https://www.atlassian.com/software/sourcetree/overview
[10] http://es.wikipedia.org/wiki/Dropbox

quiera de las computadoras sincronizadas. También existe la funcionalidad de conocer la historia de un archivo en el que se esté trabajando, permitiendo que una persona pueda editar y cargar los archivos sin peligro de que se puedan perder las versiones previas.

Se ha escogido este servicio de almacenamiento para guardar los recursos relacionados con el proyecto, como pueden ser imágenes del producto, músicas, logos, proyectos de Photoshop, versiones de la memoria, etc, para así, poder disponer de ellos en cualquier momento y para evitar la pérdida de alguno. La razón por la que se ha escogido este y no otro sistema, es porque es usado en el día a día por la desarrolladora por lo que no representa un esfuezo adicional.

4.4. Documentación

4.4.1. LaTeX

LaTeX [11] es un sistema de composición de textos, orientado especialmente a la creación de libros, documentos científicos y técnicos que contengan fórmulas matemáticas. Su principal ventaja frente a procesadores de texto como *Microsoft Word* o *LibreOffice Writer* es que separa nítidamente el contenido del documento de su formato, animando así a crear documentos bien estructurados a los que luego se les puede aplicar fácilmente en bloque las propiedades que se desee.

Se escogió este compositor de textos debido a su reciente descubrimiento y a que permitía centrarse exclusivamente en el contenido, sin tener que preocuparse de los detalles del formato. Al ser la primera vez que se trabaja en un proyecto con este compositor, previamente se tuvo que llevar a cabo un breve formación en LaTeX y además se tomó como referencia la plantilla elaborada por Alberto Calvo.

4.4.2. ShareLaTeX

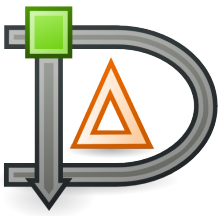
ShareLaTeX[12] es un editor de LaTeX en la nube que facilita la edición de documentos en este lenguaje. Al acceder a él a través del navegador web, permite

[11] `http://es.wikipedia.org/wiki/LaTeX`
[12] `https://www.sharelatex.com/en`

trabajar sobre los documentos en cualquier momento y desde cualquier dispositivo con conexión a Internet, con independencia de la plataforma.

Esto último fue uno de los principales motivos por lo que se consideró la elección de ShareLaTeX frente a entornos de escritorio convencionales. Otros de los motivos fueron, que permite la visualización simultánea del código fuente del documento y el PDF resultante, porque realiza un coloreado de sintaxis y corrector ortográfico en más de sesenta idiomas incluyendo castellano, da la opción de compartir el documento, y por último, debido a que los entornos de escritorio no son sencillos de instalar.

4.4.3. Dia Diagram Editor



Dia[13], es un software de código libre y abierto diseñado para la creación de diagramas. Fue desarrollado como parte del proyecto GNOME. Actualmente se pueden realizar diagramas entidad-relación, diagramas UML, diagramas de flujo, diagramas de redes, etc.

Se utilizó este software y no otros debido a la recomendación del director del proyecto. Y concretamente se ha empleado para la realización de los diagramas de secuencia del sistema del apartado de Análisis y para los diagramas de clases del apartado de Implementación.

4.4.4. Proto.io

Proto.io[14], es una aplicación para la realización de prototipos. Se lanzó por primera vez en 2011 por “ Labs Division of SNQ Digital”. Originalmente, se diseñó para crear prototipos en los dispositivos

[13] http://es.wikipedia.org/wiki/Dia_(programa)
[14] http://proto.io/en/features/

móviles, pero ahora, Proto.io se ha ampliado para permitir a los usuarios crear prototipos de aplicaciones. Proto.io utiliza una interfaz de usuario de arrastrar y soltar y no requiere codificación alguna.

Capítulo 5

Arquitectura del sistema

En este apartado, se explica cuál es la arquitectura de cualquier dispositivo con sistema operativo Android. Además, se especifica que capas han sido utilizadas por Guess the Pic. A continuación, se describen los permisos que son necesarios a la hora de la instalación del juego y por último, se comenta el patrón que se ha seguido a la hora de desarrollar la aplicación.

5.1. Arquitectura

A grandes rasgos, como cualquier sistema operativo, Android está formado por las cinco diferentes capas que se muestran en la figura 5.1.

Cada capa puede utilizar los servicios de la capa inferior. La última capa consiste en el conjunto de drivers basados en Linux, se trata de una parte privada. Un nivel más arriba, está el conjunto de librerías que no son accesibles directamente sino que hay que acceder a ellas a través del nivel superior, el Framework de aplicaciones, que junto a la capa de aplicaciones son totalmente públicas, por lo que los usuarios pueden acceder a ellas libremente.

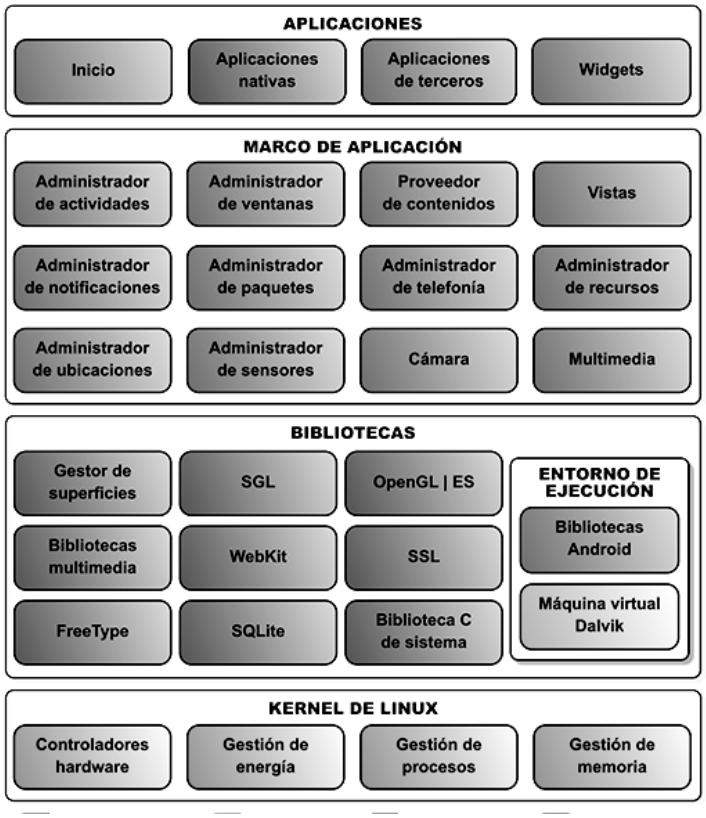


Figura 5.1: Representación de la arquitectura Android

A continuación, se detallan los diferentes componentes principales de la arquitectura de Android[1]:

- **Aplicaciones:** La capa superior de esta pila software la forman, como no podría ser de otra forma, las aplicaciones. En esta capa, se incluyen todas las aplicaciones del dispositivo, tanto las que tienen interfaz de usuario como las que no, tanto las nativas (programadas en C o C++) como las administradas (programadas en Java), tanto las que vienen de serie con el dispositivo como las instaladas por el usuario. Las aplicaciones base incluirán un cliente de email, programa de SMS, calendario, mapas, navegador, contactos, y algunos otros servicios básicos. Aquí es donde se situa nuestra aplicación, Guess the Pic.

- **Marco/Framework de aplicación:** Todos los desarrolladores de aplicaciones Android, tienen acceso total a las APIs del framework usado

[1] `http://columna80.wordpress.com/2011/02/17/arquitectura-de-android/`

en las aplicaciones base. La arquitectura está diseñada de esta forma para que no se generen cientos de componentes de aplicaciones distintas que respondan a la misma acción. Es decir, para simplificar el reuso de componentes dando la posibilidad de que los programas sean modificados o reemplazados por cualquier usuario sin tener que empezar a programar sus aplicaciones desde el principio. Este framework está formado por:

1. Conjunto de **vistas**, para poder desarrollar una aplicación como por ejemplo Buscadores, cajas de texto, botones etc. Algunos ejemplos son la "vista de mapas" y la "vista de navegadores". En nuestro juego, son utilizados algunos de estos recursos. Por ejemplo, a la hora de escribir la respuesta se utiliza una caja de texto, así como para comprobarlo, pausarlo, obtener la pista, etc. son utilizados botones.

2. **Proveedor de contenidos**, que permite a las aplicaciones acceder a información de otras aplicaciones o compartir su propia información como por ejemplo la agenda de teléfono. Guess the Pic no comparte ni utiliza información con otras aplicaciones por lo que no ha sido necesario el uso de este framework.

3. **Administrador de ventanas** que se encarga de organizar lo que se muestra en pantalla, creando superficies que pueden ser rellenadas por las actividades. En nuestro juego, hay diferentes .XML que se muestran en pantalla y que deben ser organizados, por ello, ha sido necesario usar el administrador de ventanas.

4. **Administrador de recursos** que proporciona acceso a recursos que no son código como pueden ser gráficos, cadenas de texto, etc. Guess the Pic utiliza muchas imágenes, prácticamente es lo único que hay. Para poder acceder a estas imágenes, se ha utilizado el administrador de recursos.

5. **Administrador de notificaciones**, que permite mostrar alertas. Las aplicaciones pueden añadir eventos en la barra de notificaciones. En Guess the Pic, no ha hecho falta utilizar este framework debido a que no se muestra ninguna notificación al usuario.

6. **Administrador de paquetes** que permite obtener información sobre los paquetes actualmente instalados en el dispositivo Android, además de gestionar la instalación de nuevos paquetes. Para eso mismo es para lo que se ha utilizado en nuestro juego, para su instalación.

7. **Administrador de actividades**, que permite manejar el ciclo de vida de las aplicaciones y la propia pila de actividades. En nuestro juego se ha utilizado para el manejo de las distintas actividades que lo forman. Es decir, para activar o parar sonido, para pasar de una actividad a otra, etc.

8. **Administrador de ubicaciones**, el cual permite al móvil recibir avisos, notificaciones, eventos, etc, de un lugar específico o por la localización actual del dispositivo. Nuestro juego no requiere ni ofrece en ningún momento la ubicación del dispositivo, por lo que no ha sido necesario utilizar este framework.

9. **Administrador de telefonía**, proporciona acceso a la pila hardware de telefonía del dispositivo Android. Permite realizar llamadas o enviar y recibir SMS/MMS, aunque no permite reemplazar o eliminar la actividad que se muestra cuando una llamada está en curso (por motivos de seguridad). En Guess the Pic, no ha sido necesario este framework debido a que no utiliza la telefonía en ningún momento.

10. **Administrador de sensores**, el cual permite gestionar todos los sensores hardware disponibles en el dispositivo Android: acelerómetro, giroscopio, sensor de luminosidad, sensor de campo magnético, brújula, etc. Nuestro juego no utiliza ningún sensor, por lo que no ha sido necesario hacer uso de este framework.

11. **Cámara** que proporciona acceso a las cámaras del dispositivo Android, tanto para tomar fotografías como para grabar vídeo. Guess the Pic no necesita el uso de la cámara por lo que este framework no ha sido utilizado.

12. Y por último, **Multimedia**, que consiste en un conjunto de bibliotecas que permiten reproducir y visualizar audio, vídeo e imágenes en el dispositivo. En nuestro caso, el framework multimedia, se ha utilizado para gestionar todas las imágenes que se muestran para adivinar, así como para gestionar el audio de los distintos niveles.

- **Bibliotecas:** Está formada por un conjunto de librerías escritas en C/C++. Todas están expuestas a los desarrolladores a través del Framework de aplicaciones. Estas son algunas de las bibliotecas que se incluyen habitualmente:

  1. **Gestor de superficies:** Se encarga de componer las imágenes que se muestran en la pantalla a partir de capas gráficas 2D y

3D. Cada vez que la aplicación pretende dibujar algo en la pantalla, no lo hace directamente sobre esta pantalla. En vez de eso, realiza los cambios en imágenes (mapas de bits) que almacena en memoria y que después combina para formar la imagen final que se envía a pantalla. Esto permite realizar con facilidad diversos efectos: superposición de elementos, transparencias, transiciones, animaciones, etc. Se ha utilizado esta biblioteca para componer todas las imágenes del juego, y para hacer los cambios de pantalla cuando ha sido necesario, por ejemplo, al activar o desactivar el sonido.

2. **SGL:** Es la encargada de representar elementos en dos dimensiones. Es el motor gráfico 2D de Android. Nuestro juego es una aplicación 2D, por lo que esta biblioteca ha sido la utilizada para representar todos los elementos.

3. **OpenGL:** Motor gráfico 3D. Utiliza aceleración hardware (si el teléfono la proporciona) o un motor software cuando no la hay. Como se ha mencionado en el punto anterior, Guess the Pic es una aplicación 2D, por lo que no ha necesitado esta biblioteca.

4. **Bibliotecas multimedia**: Están basadas en OpenCORE, permiten visualizar, reproducir e incluso grabar numerosos formatos de imagen, vídeo y audio. Para poder visualizar las imágenes del juego, ha sido necesario utilizar la biblioteca multimedia.

5. **WebKit:** Se trata del motor web utilizado por el navegador. Para la aplicación Guess the Pic no es necesario el uso de esta biblioteca debido a que no utiliza el navegador.

6. **SSL:** Proporciona seguridad al acceder a Internet por medio de criptografía. En nuestro juego no se necesita en ningún momento el acceso a internet, ya que tanto las puntuaciones como las imágenes y sonidos se encuentran almacenados en el propio .apk por lo que no ha hecho falta utilizar la biblioteca SSL.

7. **FreeType:** Permite mostrar fuentes tipográficas, tanto basadas en mapas de bits como vectoriales. A la hora de escribir la respuesta en el juego se hace uso de esta biblioteca o cuando se le indica al usuario, en la pantalla de puntuaciones, que pulse la pantalla para volver al menú principal.

8. **SQLite:** Es el motor de bases de datos relacionales, disponible para todas las aplicaciones. No se ha hecho uso de ningún base

de datos para nuestro juego, por lo que esta biblioteca no ha sido necesaria.

9. **Biblioteca C de sistema:** Proporciona funcionalidad básica para la ejecución de las aplicaciones. Evidentemente, esta biblioteca ha sido necesaria para poder ejecutar Guess the Pic.

- **Entorno de ejecución:** El entorno de ejecución de Android, aunque se apoya en las bibliotecas enumeradas anteriormente, no se considera una capa en sí mismo, dado que también está formado por bibliotecas. El componente principal del entorno de ejecución de Android es la máquina virtual Dalvik, componente que ejecuta todas y cada una de las aplicaciones no nativas de Android. Las aplicaciones se codifican normalmente en Java y son compiladas. Evidentemente, este entorno, se ha utilizado para poder ejecutar nuestra aplicación, tanto para las pruebas como para disfrutar de la versión definitiva.
- **Kernel de linux:** El núcleo del sistema operativo Android es un kernel Linux versión 2.6. Proporciona una capa de abstracción para los elementos hardware a los que tienen que acceder las aplicaciones. Esto permite que se pueda acceder a esos componentes sin necesidad de conocer el modelo o características precisas de los que están instalados en cada teléfono. Además, el kernel se encarga de gestionar los diferentes recursos del teléfono y del sistema operativo en sí. Indirectamente, Guess the Pic también ha hecho uso de esta capa ya que una vez que se utiliza el juego, se necesita el kernel de linux para su funcionamiento.

5.1.1. Permisos

Cuando se instala una aplicación en Google Play, lo primero que se muestra antes de instalar son los permisos que requiere la aplicación con la lista de todos los recursos e información a la que necesita acceder dicha aplicación para funcionar correctamente.

En Android existen una gran cantidad de permisos[2] para acceder a los recursos de los dispositivos. A continuación, se explican los tres permisos que se requieren a la hora de instalar Guess the Pic:

- **Permiso para modificar/eliminar contenido del almacenamiento externo:** Para ello, se ha necesitado hacer uso del permiso *"WRI-*

[2] `http://developer.android.com/guide/topics/security/permissions.html`

TE_EXTERNAL_STORAGE". Con este permiso, se permite el borrado y modificación de archivos en la memoria exterior. Este permiso es necesario para poder almacenar los niveles y subniveles completados y para almacenar los datos relacionados con las puntuaciones, tiempo, intentos, etc. Al dar permiso para escribir en el almacenamiento externo, también se podrán modificar/eliminar ficheros externos creados por otras aplicaciones.

- **Permiso para leer contenido del almacenamiento externo**: Con *"READ_EXTERNAL_STORAGE"* se permite leer archivos en la memoria exterior. Al igual que se necesita un permiso para almacenar los niveles y subniveles completados y los datos relacionados con las puntuaciones, también se necesita un permiso para poder leer estos datos. Ha sido necesario incluir este permiso para aquellos dispositivos con una versión superior a la 4.1, ya que en versiones anteriores,todas las aplicaciones podían leer en la memoria externa.
- **Permiso para impedir que el teléfono entre en modo de suspensión:** *"WAKE_LOCK"* permite que mientras se esté jugando, el juego no se suspenda y así evitar que se apague la pantalla si se lleva un tiempo sin utilizarla. Este permiso a lo único que afecta es a la batería.
- **Permiso para obtener información de las tareas en ejecución:** El permiso *"GET_TASKS"*, se ha utilizado a la hora de cerrar el juego. Cuando se pulsa el botón de atrás del teclado y el juego solicita confirmación para cerrarlo, se necesita conocer todas las actividades en ejecución del juego para poder cerrarlas y evitar que se quede alguna activa.

5.2. Modelo-Vista-Controlador

Al igual que en otro tipo de aplicaciones que son programadas, para el desarrollo de juegos se pueden aplicar patrones de diseño ya que ayudan a diseñar el código tomando como referencia un escenario determinado.

A la hora de desarrollar esta aplicación, se han seguido algunas ideas del patrón de arquitectura de software **Modelo-Vista-Controlador**[3] [4]. En la

[3] `http://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador`

[4] `http://androideity.com/2012/05/10/la-importancia-del-mvc-en-android/`

figura 5.2, se puede observar un esquema de este patrón.

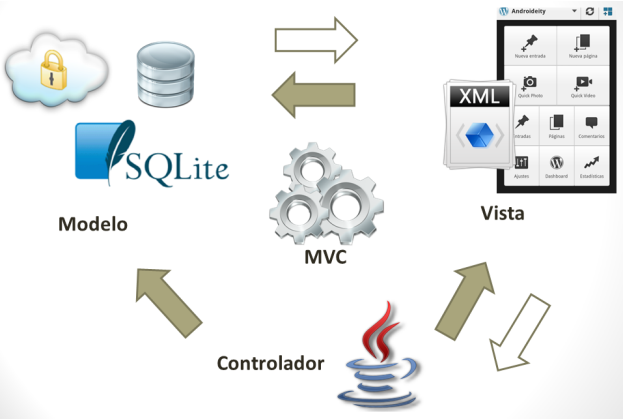


Figura 5.2: Esquema del patrón de arquitectura MVC

La parte del *Modelo* es la que representa todo el contenido del juego de manera abstracta independientemente de los sonidos, de los mapas de bit o eventos de entrada. En esta parte del modelo, también se encuentra el modelo para almacenar información. En nuestro caso, no se ha necesitado ninguna base de datos, ni ningún servicio web por lo que este componente no se ha utilizado.

Por otro lado, está la parte de *Vista* del patrón, que será el código encargado de dibujar Guess the Pic. Para ello, hay un método que dibuja en la pantalla lo que se le pide. En Android, las interfaces se construyen en XML. Los modelos de las clases no se preocupan por cómo se dibujará en la pantalla, aunque esta es una de las partes más importantes a la hora de trabajar con este patrón. Las clases del modelo son independientes, no obstante, las clases de la vista y los métodos dependen de ellas.

La parte del *Controlador* del patrón es la responsable de indicarle a las clases del modelo que cambien su estado basándose en elementos como las entradas del usuario o el tiempo que consume una actividad. Las clases del modelo proporcionan los métodos al controlador. El controlador las empleará para modificar el estado del modelo. No tienen un código que acceda directamente a elementos como la pantalla táctil. Así se consigue que estas clases no dependan de ningún elemento externo.

Utilizando este patrón se puede implementar toda la lógica del juego sin saber nada sobre los gráficos o los sonidos, es decir, se puede modificar un dibujo del juego sin tener que variar las clases del modelo.

En el apartado 9.4 “Manual de implementación”, del capítulo 9 “Implementación”, se explica más detalladamente todas las clases pertenecientes a los componentes Vista y Controlador.

Capítulo 6

Captura de requisitos

En este capítulo se exponen los casos de uso planteados que cubren por completo el funcionamiento de la aplicación y el modelo de dominio confeccionado para el desarrollo del sistema con la correspondiente explicación de cada clase y un prototipo realizado por cada una de ellas.

En este proyecto, se desarrollará una aplicación educativa por medio de la cual se pretende enseñar a los usuarios competencias básicas de su edad como pueden ser por ejemplo los animales, las profesiones, las partes del cuerpo, etc. Además de esto, el juego les permitirá practicar idiomas ya que estará disponible tanto en castellano, como en inglés y en euskera.

Se ha definido un actor que será el único que interaccione con el sistema. Dicho actor recibe el nombre de "Jugador".

El juego comenzará con un menú de inicio, en el que se podrá elegir si Jugar, desactivar o activar el sonido y obtener ayuda sobre cómo jugar. El botón principal será el de jugar por lo que aparecerá en el centro de la pantalla mientras que los otros dos serán secundarios siendo de un tamaño menor y situándose en las dos esquinas inferiores, izquierda y derecha respectivamente.

El Jugador será quien decida cuando iniciar o detener el juego así como cuando pausarlo y reanudarlo una vez que haya comenzado a jugar. Para ello antes deberá seleccionar el idioma en el que quiere jugar y el nivel correspondiente. Además el Jugador podrá también configurar el sonido y acceder a la ayuda en la pantalla inicial mencionada en el párrafo anterior.

6.1. Casos de uso

El diagrama de casos de uso de la figura 6.1 sintetiza la interacción de los actores con el sistema. Seguidamente se detallan las características y el flujo de eventos de los que consta el actor Jugador.

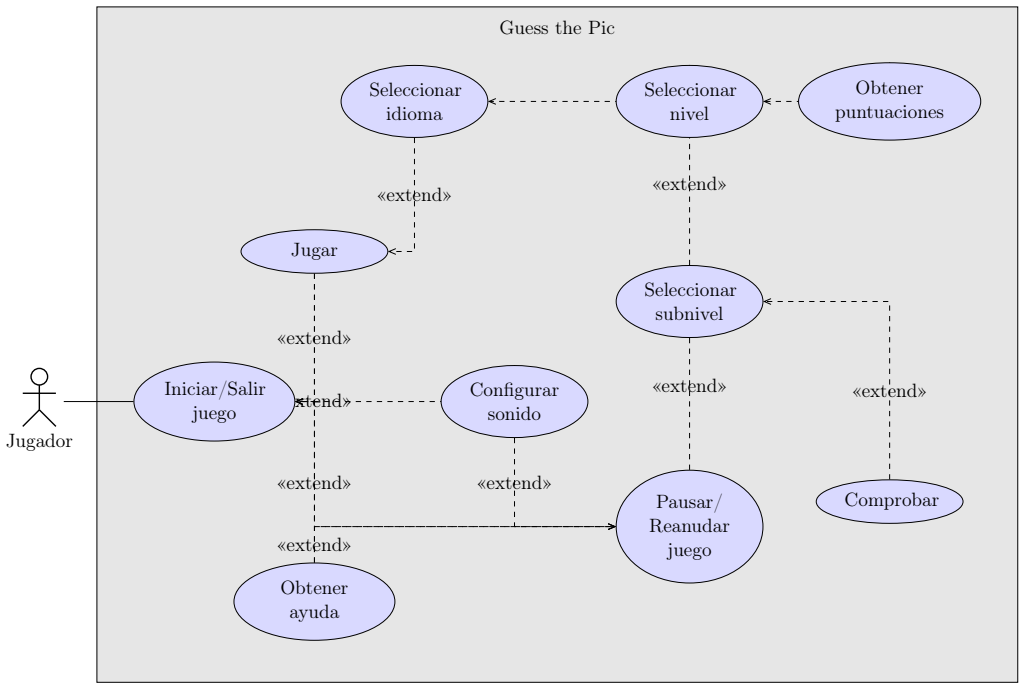


Figura 6.1: Diagrama de casos de uso

6.1.1. Iniciar/Salir juego

Este caso de uso permite al Jugador comenzar o salir del juego. Cuando el usuario pulse el icono del juego sito en sus aplicaciones, como se puede observar en la figura 6.2, este se iniciará cargando todas las imágenes que se utilizarán más adelante así como las puntuaciones guardadas y los datos de los ajustes del sonido.
De igual modo, en cualquier momento el Jugador puede salir del juego.

Actor: Jugador

Precondiciones: El Jugador tiene instalado el juego / El Jugador ha iniciado el juego

Escenario principal:

1. El Jugador inicia el juego
2. El sistema carga las imágenes, puntuaciones y ajustes guardados. Muestra la pantalla de inicio con las opciones de Jugar, Configurar el sonido u Obtener ayuda
3. El jugador puede salir del juego cuando quiera

Poscondiciones: (Ninguna)

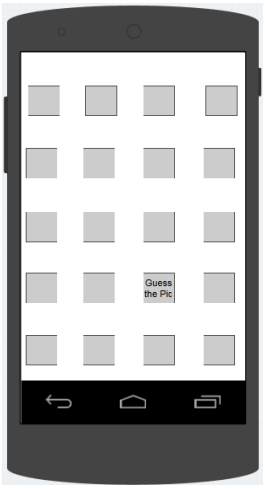


Figura 6.2: Prototipo del caso de uso "Iniciar/Salir juego"

6.1.2. Jugar

Este caso de uso permite al Jugador comenzar a jugar. Para ello deberá elegir la opción de Jugar en el menú principal, mostrado en la figura 6.3.

Actor: Jugador

Precondiciones: El Jugador ha iniciado el juego

Escenario principal:

1. El Jugador selecciona la opción Jugar en el menú principal
2. El sistema muestra la opción de seleccionar idioma

Poscondiciones: (Ninguna)

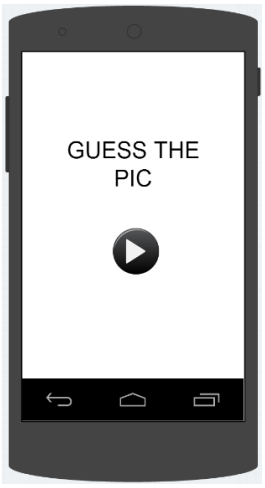


Figura 6.3: Prototipo del caso de uso "Jugar"

6.1.3. Configurar sonido

Este caso de uso permite al Jugador activar o desactivar el sonido del juego pulsando en el botón que aparece en la parte inferior izquierda del menú principal, el cual se puede ver en la figura 6.4.

Actor: Jugador

Precondiciones: El Jugador ha iniciado el juego

Escenario principal:

1. El Jugador selecciona la opción de activar o desactivar el sonido dependiendo del estado actual
2. El sistema activará o desactivará el sonido del juego mostrando de forma clara cual es la opción escogida

Poscondiciones: Almacena el nuevo estado del sonido para futuras partidas

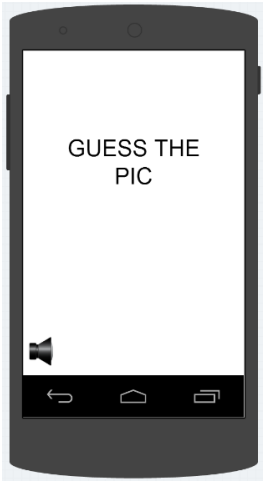


Figura 6.4: Prototipo del caso de uso "Configurar sonido"

6.1.4. Obtener ayuda

Este caso de uso permite al usuario obtener una breve explicación de cómo jugar. Para ello deberá pulsar en el botón de la parte parte inferior derecha que se muestra en la figura 6.5

Actor: Jugador

Precondiciones: El Jugador ha iniciado el juego

Escenario principal:

1. El Jugador selecciona la opción Ayuda de la parte inferior derecha de la pantalla del menú principal
2. El sistema muestra en varias pantallas y de forma sencilla cómo se debe jugar

Poscondiciones: (Ninguna)

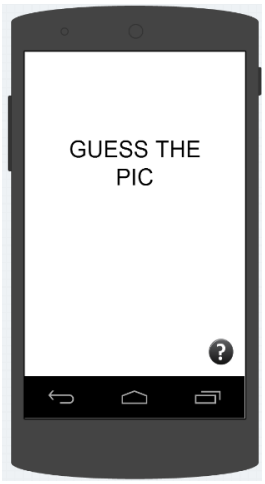


Figura 6.5: Prototipo del caso de uso "Obtener ayuda"

6.1.5. Seleccionar idioma

En este caso de uso el Jugador podrá seleccionar el idioma en el que quiere jugar. Para ello se le mostrará una pantalla con los tres idiomas, como se puede ver en la figura 6.6

Actor: Jugador

Precondiciones: El Jugador previamente ha seleccionado la opción Jugar.

Escenario principal:

1. El sistema muestra los tres idiomas a elegir: Castellano, inglés o euskera
2. El Jugador elige el idioma
3. El sistema carga los niveles del idioma seleccionado

Poscondiciones: (Ninguna)

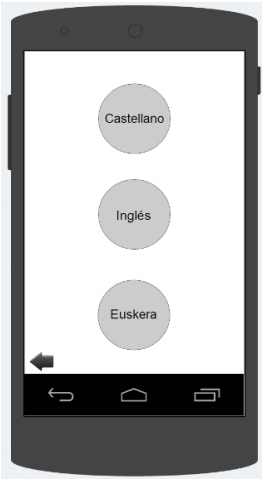


Figura 6.6: Prototipo del caso de uso "Seleccionar idioma"

6.1.6. Seleccionar nivel

En este caso de uso el Jugador deberá seleccionar el nivel en el que quiere jugar y será cuando se inicie la nueva partida. Para ello se le mostrarán dos pantallas similares a la de la figura 6.7

Actor: Jugador

Precondiciones: El Jugador previamente ha seleccionado el idioma en el que quiere jugar.

Escenario principal:

1. El sistema muestra los ocho niveles de los que consta el juego
2. El Jugador selecciona el nivel en el que quiera jugar
3. El sistema carga las puntuaciones correspondientes al nivel elegido por el Jugador
4. Al completar el nivel el sistema almacena la puntuación correspondiente al nivel, así como los intentos que se han realizado para conseguirlo y el tiempo empleado

Escenario alternativo 1:

2. El Jugador selecciona un nivel completado

2.1. El sistema reorganizará de nuevo los 20 y bloquea todos los subniveles menos el primero.

Poscondiciones: En caso de completarse el nivel, el sistema almacena su puntuación

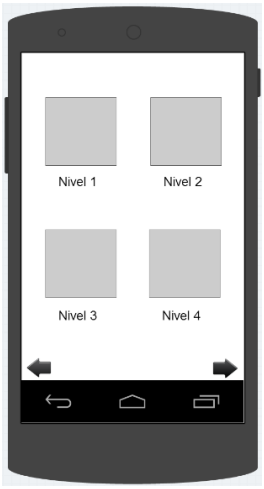


Figura 6.7: Prototipo del caso de uso "Seleccionar nivel"

6.1.7. Seleccionar subnivel

En este caso de uso el Jugador deberá seleccionar el subnivel en el que quiere y que puede jugar. Habrá 20 subniveles por nivel, como en la figura 6.8.

Actor: Jugador

Precondiciones: El Jugador previamente ha seleccionado el nivel en el que quiere jugar.

Escenario principal:

1. El sistema muestra los 20 subniveles correspondientes al nivel elegido por el Jugador bloqueando aquellos niveles que no se hayan superado
2. El jugador selecciona el último subnivel desbloqueado
3. El sistema inicia el juego y muestra en la esquina superior derecha el tiempo empleado y los intentos hasta el momento en el subnivel, la imagen a adivinar y una pequeña pista
4. Al completar todos los subniveles, el sistema muestra la puntuación total del nivel, le muestra un mensaje y reproduce un sonido de enhorabuena

Escenario alternativo 1:

2. El Jugador selecciona un subnivel completado
3. El sistema muestra en la esquina superior derecha de la pantalla los intentos y el tiempo empleado guardados sin importar que ya haya completado ese subnivel

Escenario alternativo 2:

2. El Jugador selecciona un subnivel bloqueado
3. El sistema muestra un texto en el que el Jugador comprende que no puede acceder a dicho subnivel

Escenario alternativo 3:

3. El Jugador quiere una pista y pincha sobre el botón

3.1. El sistema muestra un mensaje con el número de letras de la palabra y alguna letra en la posición correspondiente

Poscondiciones: En caso de completarse los 20 subniveles, el sistema almacena las puntuaciones correspondientes al nivel

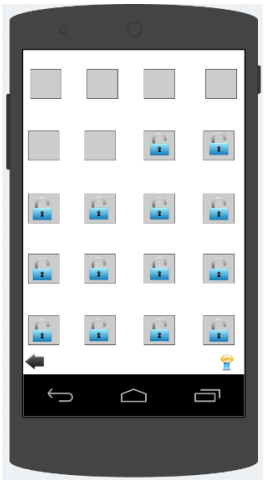


Figura 6.8: Prototipo del caso de uso "Seleccionar subnivel"

6.1.8. Obtener puntuaciones

El caso de uso permite al Jugador obtener las puntuaciones de cada nivel. En la misma pantalla que en el caso de uso anterior, podrá seleccionar las

puntuaciones en el icono de la parte inferior derecha y se le mostrará una pantalla como la de la figura 6.9.

Actor: Jugador

Precondiciones: El Jugador previamente ha seleccionado el nivel en el que quiere jugar

Escenario principal:

1. El sistema muestra un icono en la parte derecha inferior de la pantalla
2. El Jugador pulsa el botón de puntuaciones
3. El sistema muestra una nueva pantalla con la mejor puntuación total, las estrellas conseguidas, el tiempo empleado y el número de intentos
4. El jugador toca la pantalla
5. El sistema muestra la pantalla de seleccionar niveles

Escenario alternativo 1:

3. Si no se han completado aún los 20 subniveles, el sistema muestra un mensaje indicándoselo al Jugador.

Poscondiciones: (Ninguna)

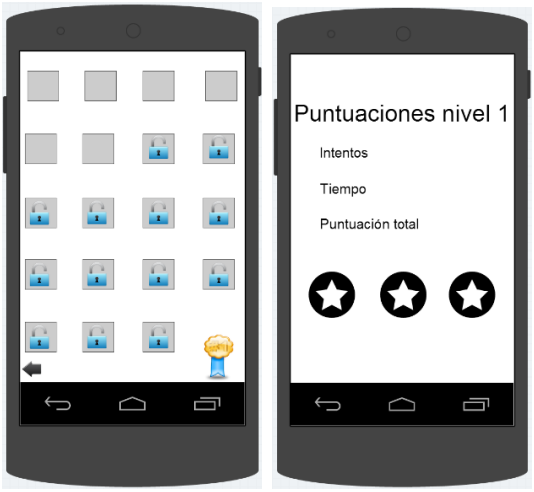


Figura 6.9: Prototipo del caso de uso "Obtener puntuaciones"

6.1.9. Pausar/Reanudar juego

El caso de uso permite al Jugador poner en pausa el juego una vez que se ha iniciado. En la figura 6.10, se puede ver esta opción. Además al parar la partida, se le dará la opción de reanudarla o de volver a empezar, la cual redirigirá al menú principal.

Actor: Jugador

Precondiciones: El Jugador ha seleccionado el subnivel en el que quiere jugar

Escenario principal:

1. El sistema muestra en todo momento en la parte superior izquiera de la pantalla un botón de pausa
2. El Jugador pulsa el botón de pausa
3. El sistema muestra una nueva pantalla con las opciones de reanudar juego y volver al menú principal
4. El jugador selecciona reanudar juego
5. El sistema retoma la partida donde el Jugador la dejó

Escenario alternativo 1:

4 El Jugador selecciona la opción de empezar

5 El sistema le muestra la pantalla del menú principal y almacena las puntuaciones conseguidas hasta el momento en el nivel

Poscondiciones: En el caso de seleccionar la opción de menu principal el sistema almacena las puntuaciones conseguidas hasta el momento

Figura 6.10: Prototipo del caso de uso "Pausar/Reanudar juego"

6.1.10. Comprobar

Este caso de uso es el que permite al Jugador comprobar si su respuesta es correcta o no y seguir avanzando en los subniveles. En la pantalla principal del juego, hay un botón con esta opción, el cual se puede observar en la figura 6.11.

Actor: Jugador

Precondiciones: El Jugador ha seleccionado el subnivel en el que quiere jugar

Escenario principal:

1. El sistema muestra en todo momento un botón para comprobar la respuesta
2. El Jugador pulsa el botón correspondiente habiendo escrito su respuesta
3. El sistema comprueba la respuesta
4. Si la respuesta es correcta pasa al siguiente subnivel, incrementa el número de intentos y muestra un mensaje indicando que es correcta

Escenario alternativo 1:

4 Si la respuesta es incorrecta incrementa el número de intentos

y muestra un mensaje indicando que es incorrecta

Poscondiciones: Si la respuesta es correcta almacena el subnivel acertado

Figura 6.11: Prototipo del caso de uso "Comprobar"

6.2. Modelo de dominio

El juego se divide principalmente en cuatro partes destacables.

La primera de ellas es el propio juego, el cual consta de ocho niveles diferentes y a su vez, cada uno de ellos consta de 20 subniveles. Por cada nivel, subnivel e idioma se guarda si ha sido completado o no. En caso afirmativo, se indica con un 1 sino con 0. Además contiene un array con cada uno de los niveles y subniveles asociados dicho nivel. Por cada nivel, se almacena el nivel del que se trata como identificador (del 1 al 8) y un array, el cual contiene la ruta de la imagen del subnivel, el nombre en castellano, inglés y euskera y las pistas en los tres idiomas. De igual modo, se puede conocer el subnivel por un entero que lo identifica (del 1 al 20).

La segunda parte es la de las puntuaciones. Por cada nivel, bien cuando sea completado o cuando el usuario decida abandonar el juego, se almacena una puntuación. En concreto, se almacenan los intentos que han sido necesarios para completar el nivel, el tiempo empleado, la puntuación total que es calculada a partir de una ponderación entre los intentos y el tiempo.Para ello,

suma el resultado de dividir un valor fijo, 200.000, entre el número de intentos, y el resultado de dividir 2.550.000 entre los segundos empleados. Se escogió 200.000 para que la puntuación máxima de los intentos fuera 10.000 ya que como mínimo en cada nivel se necesitan 20 intentos. El motivo de 2.550.000, fue gracias a las pruebas realizadas a los niños, observando que de media tardaban ocho minutos y medio en completar un nivel. Se quería que la puntuación máxima del tiempo fuera 5.000, por lo que se eligió ese valor. En un principio, se estableció que el valor fijo sería 450.000, ya que se estimó que por nivel mínimo se necesitaba un minuto y medio, es decir, 90 segundos, pero finalmente se concluyó que era muy poco tiempo para niños que están empezando a escribir. Por último, también calcula las estrellas conseguidas dependiendo de la puntuación total. El rango es entre cero y tres estrellas.

- **Menor o igual que 2.784**: Cero estrellas.
- **Entre 2.785 y 4.819**: Media estrella.
- **Entre 4.820 y 6.854**: Una estrella.
- **Entre 6.855 y 8.889**: Una estrella y media.
- **Entre 8.890 y 10.924**: Dos estrellas.
- **Entre 10.925 y 12.959**: Dos estrellas y media.
- **Mayor o igual que 12.960**: Tres estrellas.

Estos rangos se establecieron teniendo en cuenta que como máximo podrían conseguirse 15.000 puntos y como mínimo 0. Además se tuvo en cuenta que había siete posibilidades diferentes de obtener las estrellas. Por lo tanto, lo único que hubo que hacer fue dividir 15.000 entre siete obteniendo 2.142 y a partir de este valor se calcularon los rangos.

La tercera parte es la de Ayuda. En ella se mostrará al Jugador que tiene que hacer para conseguir el objetivo del juego.

La cuarta y última parte es la de los ajustes del sonido. Esta última, mostrará si el sonido está activado o no, y dará opciones para cambiarlo. Esta clase guarda un booleano para saber el estado del sonido.

En la figura 6.12 se muestra el modelo de dominio que permite visualizar de forma global lo explicado anteriormente, es decir, la composición del sistema así como las relaciones entre las diferentes clases.

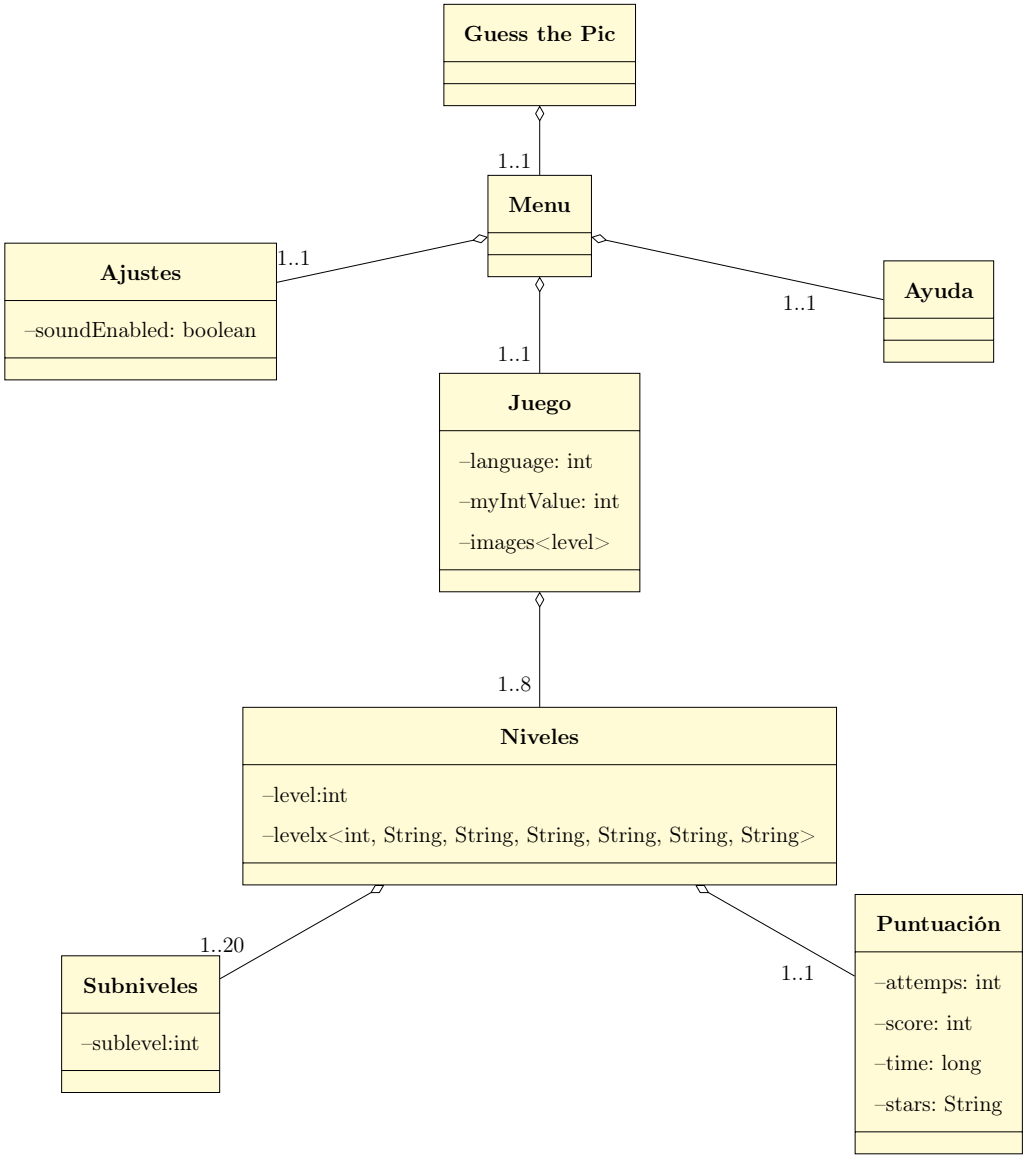


Figura 6.12: Modelo de dominio

Capítulo 7

Análisis

En este apartado se realizará un análisis más detallado de los casos de uso del capítulo anterior. Mediante el diagrama de secuencia de sistema, se mostrarán los eventos generados por el actor Jugador, su orden y las respuestas del sistema a estos. Una vez identificados los eventos que el actor genera se pasará a describirlos más detalladamente mediante los contratos especificando los cambios que realiza en el sistema y la información mostrada.

7.1. Caso de uso "Iniciar/Salir juego"

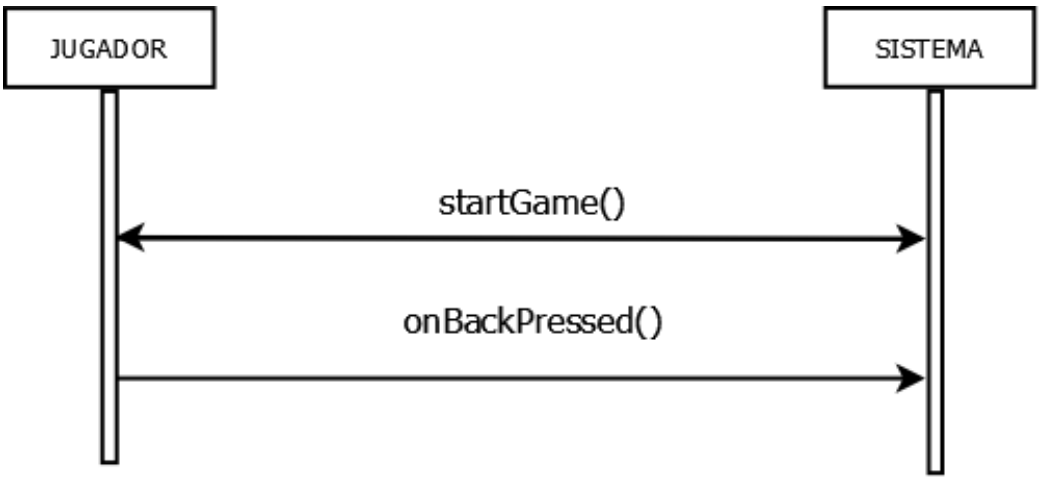


Figura 7.1: Diagrama de secuencia del caso de uso "Iniciar/Salir juego"

7.1.1. Contrato: startGame

Nombre: startGame()

Responsabilidades: Cuando se inicia el juego, carga todas las imágenes y sonidos que se usarán posteriormente así como el estado del sonido. En el caso de que esté activado, reproducirá la melodía principal.

Precondiciones: El juego no está iniciado.

Poscondiciones: El sistema carga las imágenes, sonidos y ajustes del juego.

Salida: (Ninguna)

7.1.2. Contrato: onBackPressed

Nombre: onBackPressed()

Responsabilidades: Se encarga de finalizar el juego cuando el usuario pulsa en el botón de atrás de su teclado pidiéndole una confirmación previa.

Precondiciones: El juego está iniciado.

Poscondiciones: Para la música del juego, lo finaliza y además si el usuario se encuentra en pleno juego guarda los intentos, el tiempo empleado, la puntuación total y las estrellas conseguidas.

Salida: (Ninguna)

7.2. Caso de uso “Jugar”

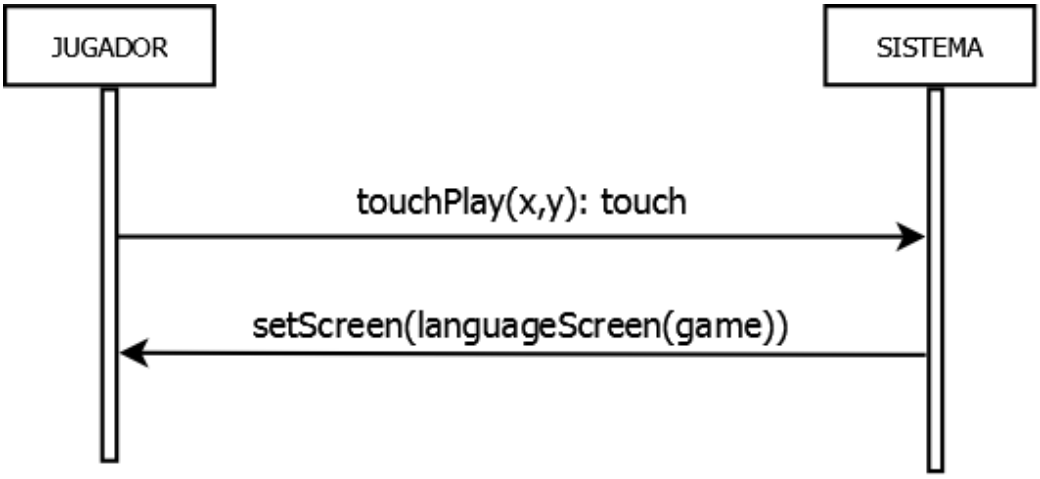


Figura 7.2: Diagrama de secuencia del caso de uso "Jugar"

7.2.1. Contrato: touchPlay

nombre: touchPlay(x,y)

Responsabilidades: Se encarga de controlar si el usuario ha pulsado en el botón de play, y de haberlo hecho, comenzará la partida.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP, es decir, el evento se detecta cuando el dedo se despega de la pantalla.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente al botón de play.

7.2.2. Contrato: setScreen(languageScreen)

nombre: setScreen(languageScreen(game))

Responsabilidades: Se encarga de establecer la pantalla de selección de lenguaje pero sin pintarla.

Precondiciones: El identificador *game* del juego no es nulo y existe.

Poscondiciones: Carga los niveles.

Salida: (Ninguna)

7.3. Caso de uso "Configurar sonido"

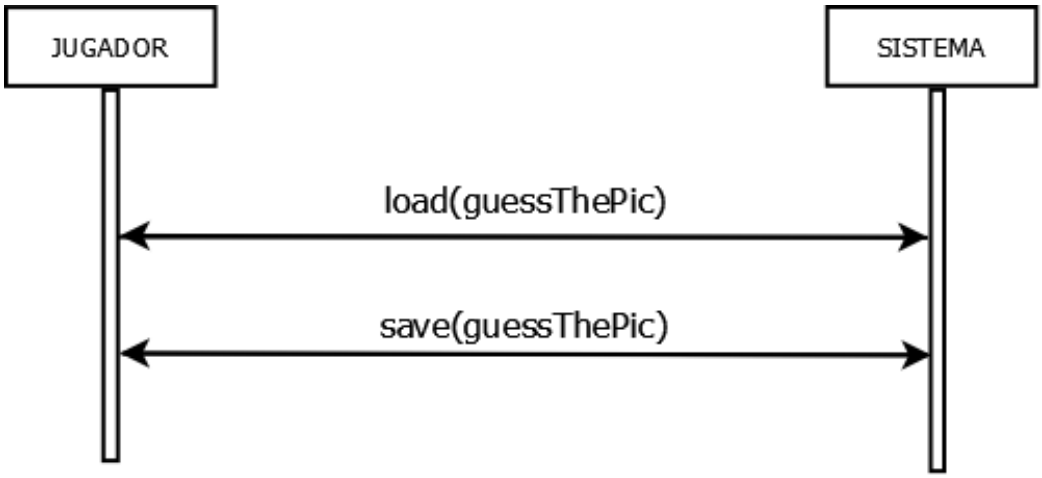


Figura 7.3: Diagrama de secuencia del caso de uso "Configurar Sonido"

7.3.1. Contrato: load

Nombre: load(guessThePic)

Responsabilidades: Se encarga de cargar del fichero *guessThePic*, el estado del sonido al iniciar el juego.

Precondiciones: El identificador *guessThePic* no es nulo y existe.

Poscondiciones: Carga en la variable *soundEnabled* el estado del sonido y en caso de estar activado, reproducirá la música y los sonidos del juego.

Salida: (Ninguna)

7.3.2. Contrato: save

Nombre: save(guessThePic)

Responsabilidades: Se encarga de guardar en el fichero *guessThePic*, el estado del sonido cuando el Jugador lo modifica.

Precondiciones: El identificador *guessThePic* no es nulo y existe.

Poscondiciones: Almacena en la variable *soundEnabled* el estado del sonido.

Salida: (Ninguna)

7.4. Caso de uso "Obtener ayuda"

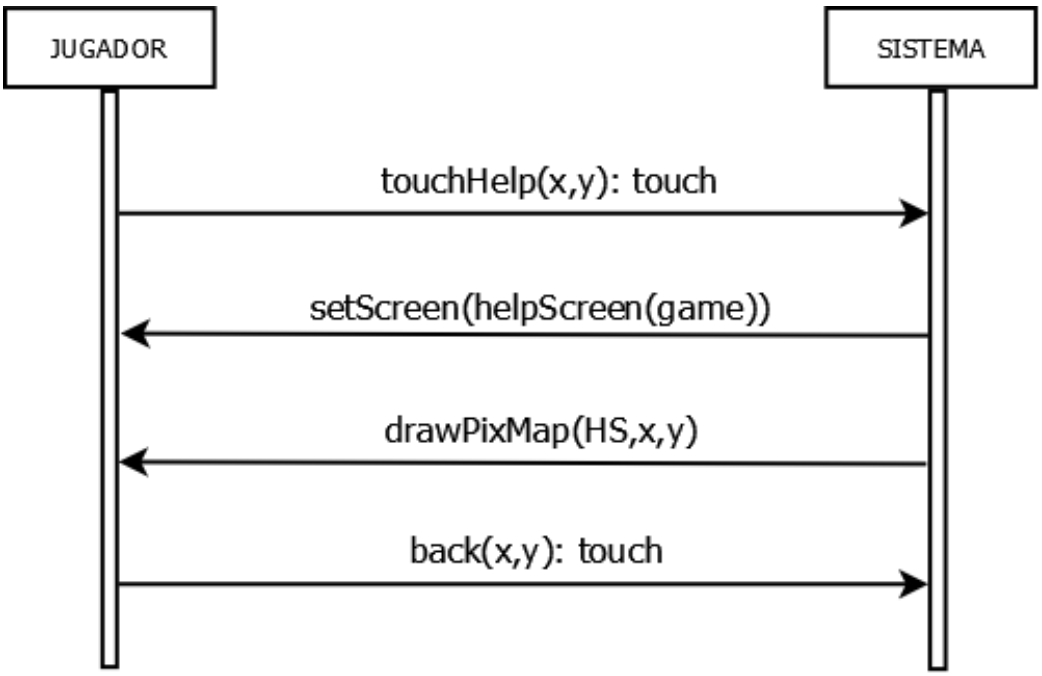


Figura 7.4: Diagrama de secuencia del caso de uso "Obtener Ayuda"

7.4.1. Contrato: touchHelp

Nombre: touchHelp(x,y):touch

Responsabilidades: Se encarga de controlar si el usuario ha pulsado en el botón de ayuda, y de haberlo hecho, le mostrará la pantalla correspondiente.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente al botón de ayuda.

7.4.2. Contrato: setScreen(helpScreen)

nombre: setScreen(helpScreen(game))

Responsabilidades: Establece cual será la pantalla de ayuda que se pintará depués.

Precondiciones: El identificador *game* del juego no es nulo y existe.

Poscondiciones: Carga la ayuda.

Salida: (Ninguna)

7.4.3. Contrato: drawPixMap

Nombre: drawPixMap(HS,x,y)

Responsabilidades: Se encarga de dibujar la primera pantalla de ayuda en las coordenadas especificadas.

Precondiciones: *HS*, x e y no son nulos y existen.

Poscondiciones: Dibuja la pantalla de ayuda.

Salida: (Ninguna)

7.4.4. Contrato: back

Nombre: back(x,y): touch

Responsabilidades: Se encarga de controlar si el usuario ha pulsado en el botón de atrás, y de haberlo hecho, le mostrará la pantalla del menú principal.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente al botón de atrás.

7.5. Caso de uso "Seleccionar idioma"

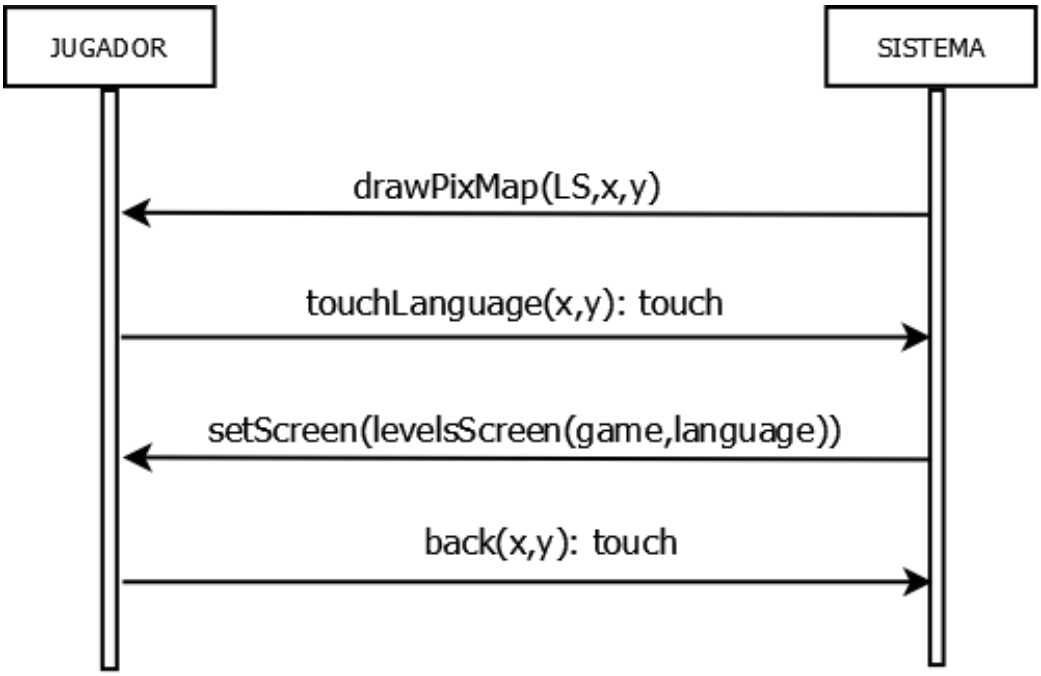


Figura 7.5: Diagrama de secuencia del caso de uso "Seleccionar idioma"

7.5.1. Contrato: drawPixMap

Nombre: drawPixMap(LS,x,y)

Responsabilidades: Se encarga de dibujar la pantalla de selección de lenguage en las coordenadas especificadas.

Precondiciones: *LS*, x e y no son nulos y existen.

Poscondiciones: Dibuja la pantalla de selección de lenguaje.

Salida: (Ninguna)

7.5.2. Contrato: touchLanguage

Nombre: touchLanguage(x,y):touch

Responsabilidades: Se encarga de controlar que lenguage ha escogido el usuario.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente a uno de los lenguajes.

7.5.3. Contrato: setScreen(levelsScreen)

nombre: setScreen(levelsScreen(game,language))

Responsabilidades: Establece cual será la pantalla correspondiente a los niveles del lenguaje *language* pero sin llegar a pintarla.

Precondiciones: Los identificadores *game* y *language* no son nulos y existen.

Poscondiciones: Carga los niveles.

Salida: (Ninguna)

7.5.4. Contrato: back

Nombre: back(x,y): touch

Responsabilidades: Se encarga de controlar si el usuario ha pulsado en el botón de atrás, y de haberlo hecho, le mostrará la pantalla del menú principal.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente al botón de atrás.

7.6. Caso de uso "Seleccionar nivel"

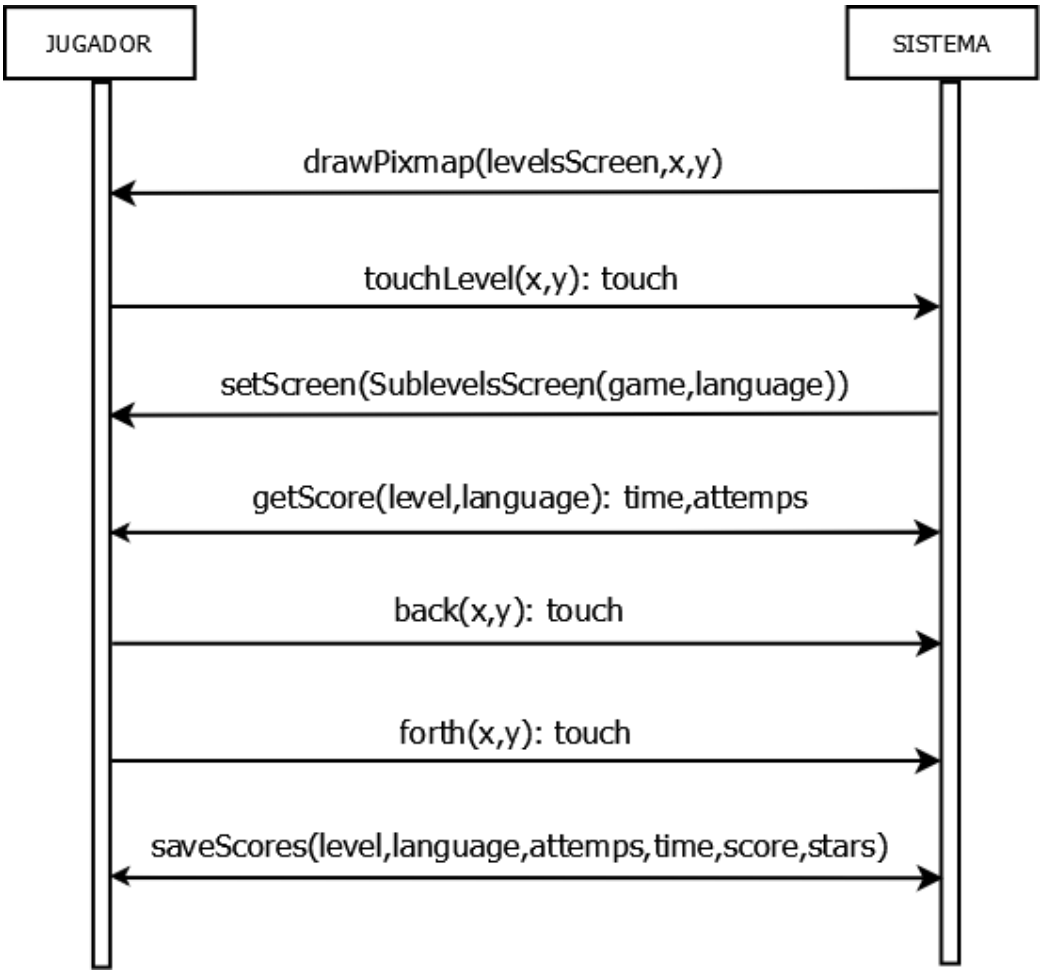


Figura 7.6: Diagrama de secuencia del caso de uso "Seleccionar nivel"

7.6.1. Contrato: drawPixMap

Nombre: drawPixMap(levelsScreen,x,y)

Responsabilidades: Se encarga de dibujar la primera pantalla de selección de nivel en las coordenadas especificadas.

Precondiciones: *levelsScreen*, x e y no son nulos y existen.

Poscondiciones: Dibuja la pantalla de selección de nivel.

Salida: (Ninguna)

7.6.2. Contrato: touchLevel

Nombre: touchLevel(x,y):touch

Responsabilidades: Se encarga de controlar que nivel ha escogido el usuario.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente a uno de los niveles.

7.6.3. Contrato: setScreen(sublevelsScreen)

nombre: setScreen(sublevelsScreen(game,language))

Responsabilidades: Establece cual será la pantalla correspondiente a los subniveles del nivel desde el que se ha llamado al método y al lenguaje *language* pero sin llegar a pintarla.

Precondiciones: Los identificadores *game* y *language* no son nulos y existen.

Poscondiciones: Carga los subniveles correspondientes.

Salida: (Ninguna)

7.6.4. Contrato: getScore

nombre: getScore(level,language): time,attemps

Responsabilidades: Se encarga de obtener las puntuaciones del nivel *level* y el lenguaje *language*.

Precondiciones: Los identificadores *level* y *language* no son nulos y existen.

Poscondiciones: (Ninguna)

Salida: Obtiene el tiempo total *time* y los intentos *attemps* empleados hasta el momento en el nivel seleccionado.

7.6.5. Contrato: back

Nombre: back(x,y): touch

Responsabilidades: Se encarga de controlar si el usuario ha pulsado en el botón de atrás, y de haberlo hecho, le mostrará de nuevo la pantalla de selección de lenguaje.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente al botón de atrás.

7.6.6. Contrato: forth

Nombre: forth(x,y): touch

Responsabilidades: Se encarga de controlar si el usuario ha pulsado en el botón de siguiente, y de haberlo hecho, le mostrará la pantalla correspondiente a la selección de niveles del 4 al 8.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente al botón de siguiente.

7.6.7. Contrato: saveScores

nombre: saveScores(level,language,attemps,time,score,stars)

Responsabilidades: Se encarga de guardar los intentos *attemps*, el tiempo *time*, la puntuación total *score* obtenida y las estrellas *stars* conseguidas en el nivel *level* y el lenguaje *language* en el caso que se completen los 20 subniveles.

Precondiciones: Los identificadores *level*, *language*, *attemps*, *time*, *score* y *stars* no son nulos y existen.

Poscondiciones: Almacena en el sistema dichas puntuaciones.

Salida: (Ninguna)

7.7. Caso de uso “Seleccionar subnivel”

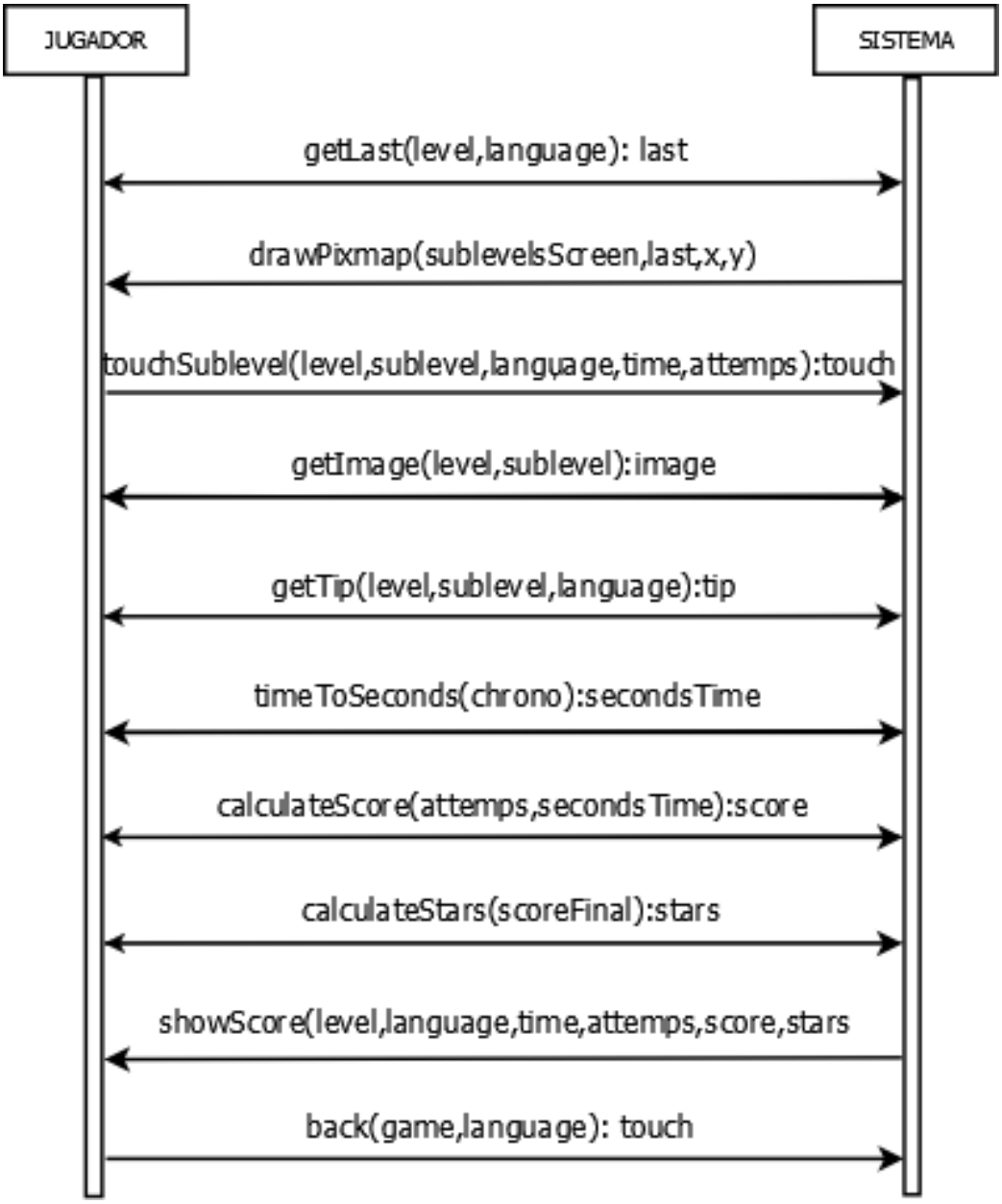


Figura 7.7: Diagrama de secuencia del caso de uso "Seleccionar subnivel"

7.7.1. Contrato: getLast

Nombre: getLast(level,language):last

Responsabilidades: Es el encargado de obtener el último subnivel desbloqueado.

Precondiciones: *level* y *language* no son nulos y existen.

Poscondiciones: (Ninguna)

Salida: last = Se trata de un entero entre 1 y 20 que indicará cual es el último nivel desbloqueado.

7.7.2. Contrato: drawPixMap

Nombre: drawPixMap(sublevelsScreen,last,x,y)

Responsabilidades: Se encarga de dibujar la pantalla de selección de subniveles en las coordenadas especificadas bloqueando aquellos subniveles con mayor identificador que *last*.

Precondiciones: *sublevelsScreen*,*last*, x e y no son nulos y existen.

Poscondiciones: Dibuja la pantalla de selección de subniveles.

Salida: (Ninguna)

7.7.3. Contrato: touchSublevel

Nombre: touchSublevel(level,sublevel,language,time,attemps):touch

Responsabilidades: Se encarga de controlar que subnivel ha escogido el usuario. Cuando el usuario selecciona el subnivel al que quiere jugar, se encarga de cargar el juego con todos los datos, *time* y *attemps*, correspondientes a dicho subnivel, el nivel al que pertenece y en el idioma seleccionado previamente para poder mostrarsela al usuario en la pantalla del juego.

Precondiciones: *level*, *sublevel*, *language*, *time* y *attemps* no son nulos y existen.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente a uno de los subniveles.

7.7.4. Contrato: getImage

nombre: getImage(level,sublevel):image

Responsabilidades: Una vez que el Jugador ha seleccionado el subnivel, se encarga de obtener la imagen correspondiente al nivel *level*, y al subnivel emphsublevel,

Precondiciones: Los identificadores *level* y *sublevel* no son nulos y existen.

Poscondiciones: (Ninguna)

Salida: image = Ruta en la que se encuentra la imagen deseada.

7.7.5. Contrato: getTip

nombre: getTip(level,sublevel,language):tip

Responsabilidades: Si el jugador desea obtener una pista de la solución, esta función se encargará de obtener la pista correspondiente al nivel *level*, al subnivel *sublevel* y al lenguaje *language*.

Precondiciones: Los identificadores *level*, *sublevel* y *language* no son nulos y existen.

Poscondiciones: (Ninguna)

Salida: tip = String que da una pista al Jugador sobre la solución del subnivel.

7.7.6. Contrato: timeToSeconds

nombre: timeToSeconds(chrono):secondsTime

Responsabilidades: A partir del tiempo necesitado *chrono* completar el nivel obtenido en el cronómetro (00:00:00), se encarga de convertir ese tiempo en segundos.

Precondiciones: El identificador *chrono* no es nulo y existe.

Poscondiciones: (Ninguna)

Salida: secondsTime = Entero que representa el tiempo invertido para completar el nivel.

7.7.7. Contrato: calculateScore

nombre: calculateScore(attemps,secondsTime):score

Responsabilidades: A partir del tiempo necesitado *secondsTime* y los intentos *attemps* empleados para completar el nivel, se encarga de calcular la puntuación total del nivel.

Precondiciones: Los identificadores *secondsTime* y *attemps* no son nulos y existen.

Poscondiciones: (Ninguna)

Salida: scoreFinal = Entero que representa la puntuación total obtenida

7.7.8. Contrato: calculateStars

nombre: calculateStars(scoreFinal):stars

Responsabilidades: A partir de la puntuación final *scoreFinal*, se encarga de calcular las estrellas conseguidas en el nivel.

Precondiciones: El identificador *scoreFinal* no es nulo y existe.

Poscondiciones: (Ninguna)

Salida: Stars = String que representa a una letra comprendida entre {a,b,c,d,e,f} donde cada una de ellas se corresponde a cada una de las siete posibilidades de las estrellas conseguidas {ninguna,media,una,una y media,dos,dos y media,tres}

7.7.9. Contrato: showScore

nombre: showScore(level,language,time,attemps,scoreFinal,stars)

Responsabilidades: Cuando se completa el subnivel 20, se encarga de mostrar las puntuaciones calculadas en los dos últimos métodos del nivel *level* y el lenguaje *language*.

Precondiciones: Los identificadores *level*, *language*, *time*, *attemps*, *scoreFinal* y *stars* no son nulos y existen.

Poscondiciones: Muestra las puntuaciones correspondientes al nivel y lenguaje escogidos.

Salida: (Ninguna)

7.7.10. Contrato: back

Nombre: back(game,language): touch

Responsabilidades: Se encarga de controlar si el usuario ha pulsado en el botón de atrás, y de haberlo hecho, le mostrará de nuevo la pantalla de selección de nivel del idioma *language*.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP y los identificadores *game* y *language* no son nulos y existen.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente al botón de atrás.rawPixMap(sublevelsScreen,last,x,y)

7.8. Caso de uso "Obtener Puntuaciones"

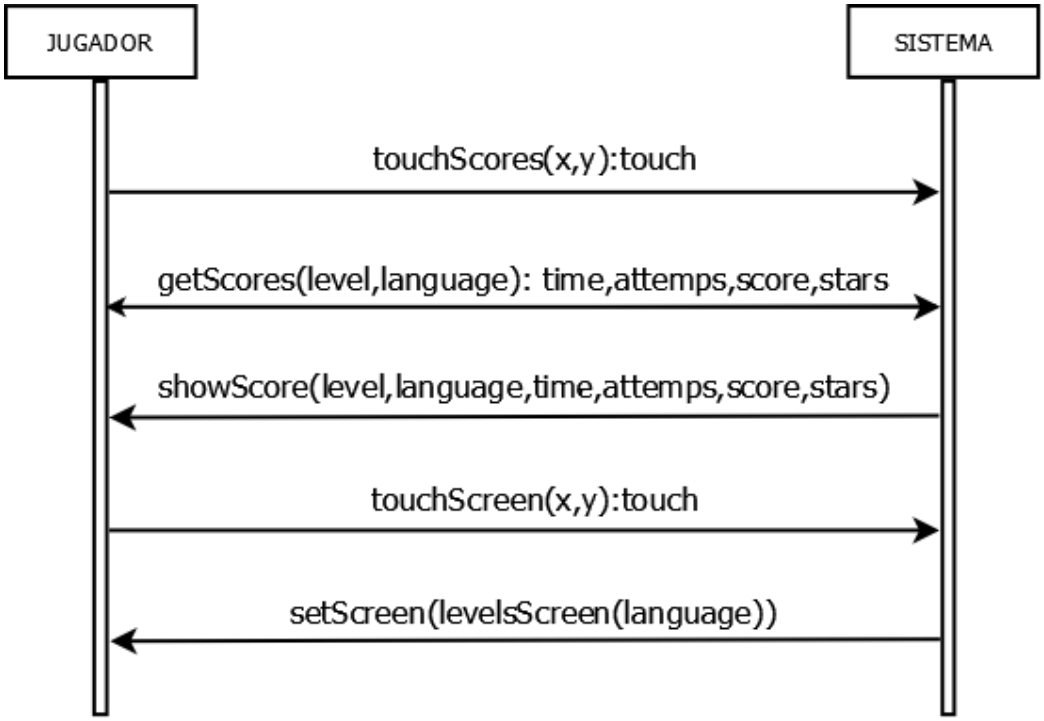


Figura 7.8: Diagrama de secuencia del caso de uso “Obtener puntuaciones”

7.8.1. Contrato: touchScores

Nombre: touchScores(x,y):touch

Responsabilidades: Se encarga de controlar si el usuario ha pulsado en el botón de puntuaciones, y de haberlo hecho y completado el nivel, le mostrará la pantalla correspondiente.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente al botón de puntuaciones.

7.8.2. Contrato: getScores

nombre: getScores(level,language): time,attemps,score,stars

Responsabilidades: Se encarga de obtener las puntuaciones correspondientes al nivel *level*, y al lenguaje *language*.

Precondiciones: Los identificadores *level* y *language* no son nulos y existen.

Poscondiciones: (Ninguna)

Salida: Obtiene el tiempo total *time* y los intentos *attemps* empleados hasta el momento en el nivel seleccionado, con ello calcula la puntuación total *score* y el número de estrellas conseguidas *stars*.

7.8.3. Contrato: showScore

nombre: showScore(level,language,time,attemps,score,stars)

Responsabilidades: Se encarga de mostrar la pantalla de puntuaciones con los datos obtenidos en la función anterior.

Precondiciones: Los identificadores *level*, *language*, *time*, *attemps*, *score* y *stars* no son nulos y existen.

Poscondiciones: Muestra las puntuaciones correspondientes al nivel y lenguaje escogidos.

Salida: (Ninguna)

7.8.4. Contrato: touchScreen

Nombre: touchScreen(x,y):touch

Responsabilidades: Se encarga de controlar si el usuario ha pulsado en cualquier parte de la pantalla de las puntaciones, y de haberlo hecho, le mostrará la pantalla correspondiente.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado en cualquier parte de la pantalla.

7.8.5. Contrato: setScreen(levelsScreen)

nombre: setScreen(levelsScreen(language))

Responsabilidades: Establece cual será la pantalla de niveles que se pintará depués.

Precondiciones: El identificador *language* no es nulo y existe.

Poscondiciones: Carga los niveles.

Salida: (Ninguna)

7.9. Caso de uso "Pausar/Reanudar juego"

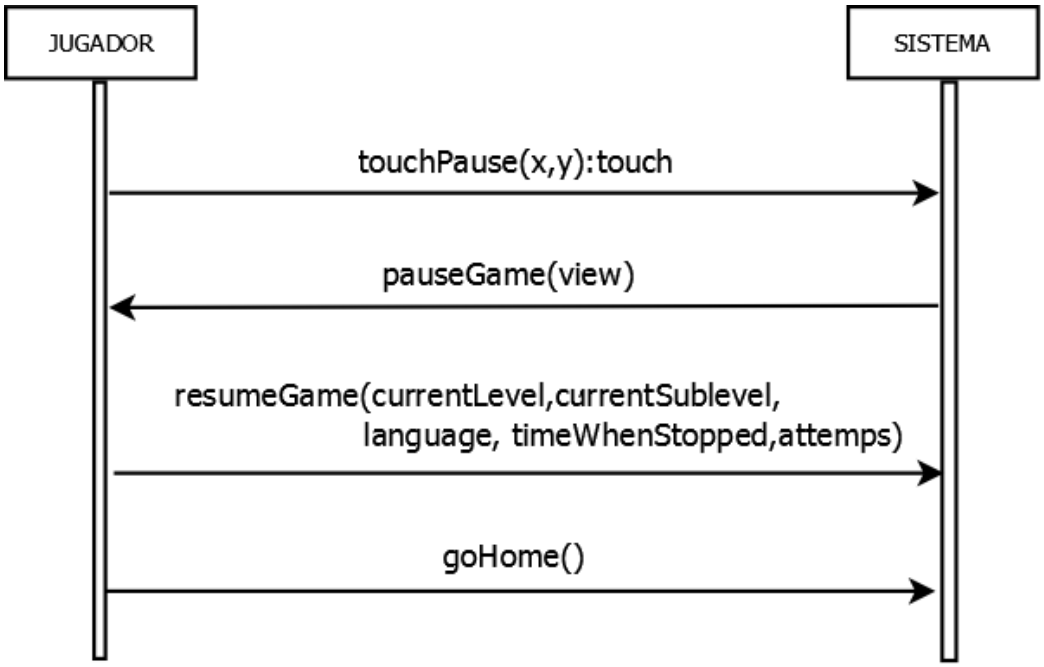


Figura 7.9: Diagrama de secuencia del caso de uso "Pausar/Reanudar juego"

7.9.1. Contrato: touchPause

Nombre: touchPause(x,y):touch

Responsabilidades: Se encarga de controlar cuando el usuario toca el botón de pausa.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente del botón de pausa.

7.9.2. Contrato: pauseGame

nombre: pauseGame(view)

Responsabilidades: Se encarga de poner en pausa el juego y de mostrar la pantalla correspondiente.

Precondiciones: El identificador *view* no es nulo y existe.

Poscondiciones: Muestra la pantalla de pausa y además guarda el estado actual del juego.

Salida: (Ninguna)

7.9.3. Contrato: resumeGame

nombre: resumeGame(currentLevel,currentSublevel,language, timeWhenStopped,attemps)

Responsabilidades: Se encarga de poner en marcha de nuevo el juego donde el Jugador lo dejó.

Precondiciones: Los identificadores *currentLevel*, *currentSublevel*, *language*, *timeWhenStopped* y *attemps* no son nulos y existen.

Poscondiciones: Retoma el juego en el nivel *currentLevel*, subnivel *currentSublevel*, con el lenguaje *language* y estableciento el tiempo *timeWhenStopped* y los intentos *attemps* empleados.

Salida: (Ninguna)

7.9.4. Contrato: goHome

nombre: goHome()

Responsabilidades: Cuando el jugador selecciona la opción de volver al menú principal, se encarga de volver a la pantalla del menú principal y de almacenar el estado actual del juego.

Precondiciones: El juego está en Pause.

Poscondiciones: Almacena en el sistema las puntuaciones, en concreto el tiempo *timeWhenStopped* y los intentos *attemps* por cada nivel *currentLevel* y lenguage *language* .

Salida: (Ninguna)

7.10. Caso de uso “Comprobar”

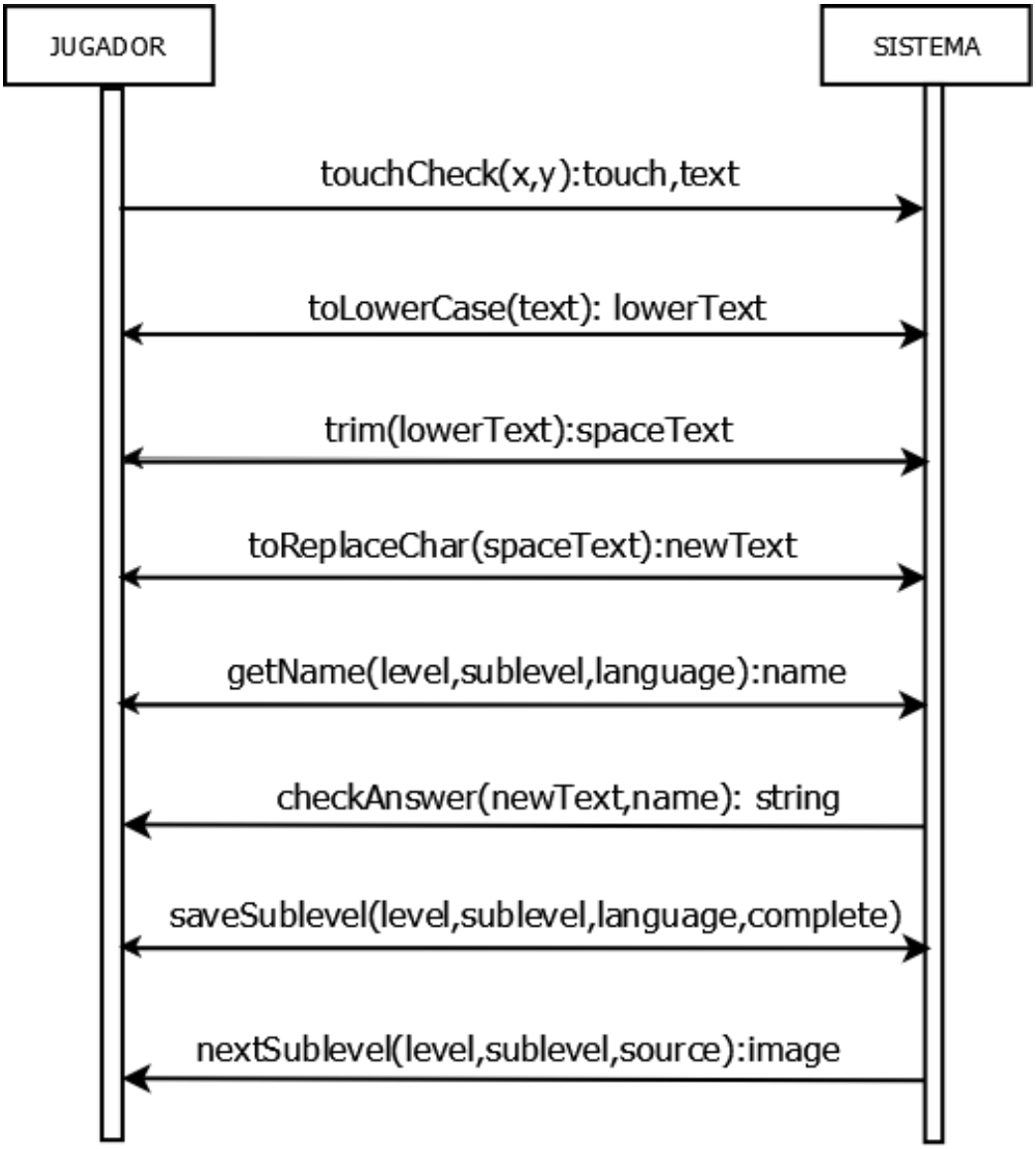


Figura 7.10: Diagrama de secuencia del caso de uso "Comprobar"

7.10.1. Contrato: touchCheck

Nombre: touchCheck(x,y):touch,text

Responsabilidades: Se encarga de controlar si el usuario ha pulsado en el botón de comprobar, y de haberlo hecho, pasará a comprobar su respuesta *text*.

Precondiciones: El evento del toque del dedo debe de ser del tipo UP.

Poscondiciones: (Ninguna)

Salida: touch = Boolean que será cierto en el caso de que se haya tocado la zona correspondiente al botón de puntuaciones.
Text = String que se corresponde con la respuesta introducida por el jugador.

7.10.2. Contrato: toLowerCase

nombre: toLowerCase(text):lowerText

Responsabilidades: Reemplaza las mayúsculas de las palabras introducidas, *text*, por minúsculas.

Precondiciones: El identificador *text* no es nulo y existe.

Poscondiciones: (Ninguna)

Salida: lowerText = String que representa la respuesta introducida por el usuario en minúscula.

7.10.3. Contrato: trim

nombre: trim(lowerText):spaceText

Responsabilidades: Elimina los espacios del comienzo y del final que pueda haber en la palabras introducida, *lowerText*.

Precondiciones: El identificador *lowerText* no es nulo y existe.

Poscondiciones: (Ninguna)

Salida: spaceText = String que representa la respuesta introducida por el usuario sin espacios.

7.10.4. Contrato: toReplaceChar

nombre: toReplaceChar(spaceText):newText

Responsabilidades: Reemplaza las palabras introducidas, *spaceText*, con tilde y ñ para que a la hora de comprobarlo con la respuesta correcta no de error.

Precondiciones: El identificador *spaceText* no es nulo y existe.

Poscondiciones: (Ninguna)

Salida: newText = String que representa la respuesta introducida por el usuario en código ascii.

7.10.5. Contrato: getName

nombre: getName(level,sublevel,language):name

Responsabilidades: Si el jugador ha pulsado sobre coomprobar su respuesta, esta función se encargará de obtener la solución correspondiente al nivel *level*, al subnivel *sublevel* y al lenguaje *language*.

Precondiciones: Los identificadores *level*, *sublevel* y *language* no son nulos y existen.

Poscondiciones: (Ninguna)

Salida: name = String que representa la solución correcta del subnivel.

7.10.6. Contrato: checkAnswer

nombre: checkAnswer(newText,name):string

Responsabilidades: Dado la solución *name* y la respuesta introducida por el jugador *newText*, se encarga de comprobar si es corresta o no y de comunicárselo al usuaruio.

Precondiciones: Los identificadores *name* y *newText* no son nulos y existen.

Poscondiciones: (Ninguna)

Salida: string = String que contiene el mensaje informativo para el jugador, correcto, incorrecto.

7.10.7. Contrato: saveSublevel

nombre: saveSublevel(level,sublevel,language,complete)

Responsabilidades: Se encarga de guardar los subniveles completados. *Complete*, es un int que tomará el valor de 0 cuando el subnivel no este completado y por tanto, 1 cuando se haya completado.

Precondiciones: Los identificadores *level*, *sublevel*, *language* y *source* no son nulos y existen.

Poscondiciones: Almacena en el sistema el subnivel completado.

Salida: (Ninguna)

7.10.8. Contrato: nextSublevel

nombre: nextSublevel(level,sublevel,source):image

Responsabilidades: Se encarga de obtener la imagen del siguiente subnivel *sublevel* en caso de que la respuesta haya sido correcta.

Precondiciones: Los identificadores *level*, *sublevel* y *source* no son nulos y existen.

Poscondiciones: (Ninguna)

Salida: image = Ruta en la que se encuentra la imagen deseada.

Capítulo 8

Diseño

En este capítulo se describen los diseños de las funciones definidas en el capítulo 7.

Se ha decidido representarlo mediante pseudocódigo para facilitar al lector su comprensión. Muchas de las funciones son iguales por lo que únicamente se explicará una de ellas, en concreto, la primera que aparezca. En el caso de utilizarse métodos y funciones de la librería no se explicarán debido a que no se tiene información sobre su implementación.

Se ha procurado que las clases sean lo más sencillas posible para un mejor mantenimiento y mayor facilidad de comprensión. Con esto, se obtendrá una alta cohesión entre ellas y se reducirá el tiempo para localizar los problemas.

8.1. Caso de uso "Iniciar/Salir juego"

8.1.1. Nombre: startGame()

Cuando el usuario inicia el juego pinchando sobre el icono del juego en sus aplicaciones, la clase *GuessThePicActivity* lo iniciará, cargará el fichero *.guesThePic* que contiene el último estado del sonido y además llamará a la clase *Loading*, la cual se ocupará de cargar todos los screen del juego y las nueve melodías diferentes del juego, así como de comenzar el juego llamando a la clase *MainMenuScreen*.

Carga el estado del sonido → Settings.load(getFileIO()) [*visto más adelante*]
Llama a la clase *Loading*
Carga los trece fondos de pantalla que utiliza el juego → g.newPixmap("backgroundOn.png", PixmapFormat.RGB565)
Carga las nueve melodías que se usan durante en el juego; la melodía principal y la de la cada nivel → game.getAudio().newMusic("nivel2.mp3")
Carga el sonido de enhorabuena que se utilizará cuando el usuario complete los 20 subniveles → game.getAudio().newSound(çongratulations.ogg")
Si el sonido está activado
  Entonces reproduce la melodía principal del juego
Fin Si
Llama a *MainMenuScreen* para iniciar el juego

8.1.2. Nombre: onBackPressed()

Este método está situado en todos los activity, es decir, en la clases *GuessThePicActivity*, *GameScreen*, *ReturnLevelsScreenActivity*, *ReturnMainMenuActivity* y *ScoresActivity*. Se encarga de detectar cuando el usuario ha pulsado el botón de atrás-salir de su teclado. En todos los casos mostrará un mensaje de confirmación, y en caso afirmativo, cada una de estas clases, parara la melodía que se esté reproduciendo en ese instante y llamará a la clase principal *GuessThePicActivity* que será la encargada de finalizar el juego. En el caso en el que el Jugador desee salir del juego una vez que ya ha comenzado la partida, es decir, si se encuentra en la clase *GameScreen*, además de lo anterior, almacenará los intentos, el tiempo empleado, la puntuación total y las estrellas conseguidas. Para representar el caso más complejo y completo, se hará el diseño de la clase *GameScreen*.

Si el Jugador ha pulsado en el botón de salir
Crea una alerta en la que le preguntará al usuario si está seguro de que desea salir
Si elegir la opción salir
Si el sonido está activado
Para el sonido del nivel correspondiente
Fin Si
Guarda los intentos → putInt(level + "-" + language + "intentos", attempTotal)
Guarda el tiempo → putLong(level + "-" + language + "tiempo", crono. getBase() SystemClock.elapsedRealtime())
Guarda la puntuación total → putInt(level + "-" + language + "puntuacion", scoreFinal)
Guarda las estrellas conseguidas → putString(level + "-" + language + "estrellas", starsFinal)
Llama a la clase *GuessThePicActivity* indicándole que cierre el juego
Si recibe el Flag "Exit" *GuessThePicActivity* de la clase *GameScreen*
Entonces finaliza el juego → GuessThePicActivity.this.finish()
Fin Si
Sino Si pincha sobre no salir
Entonces muestra la pantalla del juego
Fin Si
Fin Si

8.2. Caso de uso "Jugar"

8.2.1. Nombre: touchPlay(x,y):touch

Es una de las acciones principales, la cual se necesita para seguir con el desarrollo del juego. Se evalua la pulsación del usuario en la pantalla. La clase *mainMenuScreen* se encarga de saber si las coordenadas de la pulsación están dentro del rango, y en caso afirmativo, crea un juego con los parámetros iniciales.

Crea una lista de eventos → <TouchEvent>touchEvents en la cual se almacenarán todos los eventos detectados.
Por cada evento:
  Si el evento es de tipo TOUCH_UP
    Calcula la zona dónde se encuentra el rectángulo del botón ayuda → Con x =11 e y=731, calcula la esquina superior izquierda del rectángulo y con el ancho=72 y el alto=57, calcula la esquina inferior derecha del rectángulo
    Si la zona en la que ha tocado la pantalla está en las coordenadas especificadas
      Entonces llama al método setScreen(languageScreen(game))
    Fin Si
  Fin Si
Final For

8.2.2. Nombre: setScreen(languageScreen(game))

Este método se encuentra definido en la clase *AndroidGame*. Por lo tanto, si el método es llamado por la función anterior desde la clase *MainMenuScreen*, se llamará al método de la clase *AndroidGame* el cual se encargará de establecer la nueva pantalla a pintar, languageScreen.

Si languageScreen no es nula
  Pausa la pantalla actual → MainMenuScreen
  Libera la pantalla actual → MainMenuScreen para dejar sitio a la nueva instancia
  Si el intervalo de tiempo es 0
    Entonces se retoma la actividad y se actualiza la pantalla
  Fin Si
  Establece la nueva pantalla a pintar
Fin Si

8.3. Caso de uso “Configurar sonido”

8.3.1. Nombre: load(guessThePic)

El método load situado en la clase *Settings* carga el fichero *guessThePic*, para conocer el estado del sonido cuando el jugador inicia el juego, es decir, la

clase *GuessThePicActivity* llama a este método y cuando obtiene el estado de la variable reproduce o no la melodía principal del juego.

Crea el Buffer de lectura in
Try
  Carga el fichero GuessThePic indicado por la clase *GuessThePicActivity*
  Almacena en el buffer in lo leido en el fichero GuessThePic
  Guarda en la variable booleana soundEnabled el estado del sonido
catch
  Cierra el fichero buffer in si hay alguna excepción
Fin Try

8.3.2. Nombre: save(guessThePic)

El método save situado en la clase *Settings*, guarda en el fichero *guessThePic* el estado del sonido cada vez que el jugador lo modifica en la clase *MainMenuScreen*. Esto es para que en las posteriores partidas se cargue el estado del sonido deseado por el Jugador, es decir, si el Jugador desactivó en su última partido el sonido, la siguiente partida se iniciará el juego con el sonido desactivado para evitar así que tenga que hacerlo de nuevo en las siguientes partidas.

Crea el Buffer de escritura out
Try
  Carga el fichero GuessThePic indicado por la clase *MainMenuScreen*
  Escribe en el buffer out el estado de la variable soundEnabled que se le ha indicado en *MainMenuScreen* cada vez que el Jugador active o desactive el sonido
  Cierra el buffer out si hay alguna excepción
Fin Try

8.4. Caso de uso "Obtener ayuda"

8.4.1. Nombre: drawPixMap(HS,x,y)

La clase *HelpScreen* se encarga de dibujar la pantalla correspondiente a la ayuda. Para ello, llama a este método definido en la clase *AndroidGraphics* que lo único que hará será pintar todo el Pixmap en el framebuffer en las coordenadas 0,0.

Se obtiene el miembro Bitmap de HelpScreen →
Convierte las coordenadas 0,0 en la esquina superior izquierda del rectángulo donde se quiere empezar a pintar.
Llama al método drawBitmap de la clase *Canvas* implementada en la librería utilizada
Dibuja HelpScreen dentro del framebuffer

8.4.2. Nombre: back(x,y): touch

El método back(x,y) no deja de ser un método touch(x,y). En este caso la clase *helpScreen* se encarga de saber si las coordenadas de la pulsación están dentro del rango, y en caso afirmativo, retrocede a la clase *mainMenuScreen*.

Crea una lista de eventos → <TouchEvent>touchEvents en la cual se almacenarán todos los eventos detectados.
Por cada evento:
  Si el evento es de tipo TOUCH_UP
    Calcula la zona dónde se encuentra el rectángulo del botón ayuda → Con x =11 e y=731, calcula la esquina superior izquierda del rectángulo y con el ancho=72 y el alto=57, calcula la esquina inferior derecha del rectángulo.
    Si la zona en la que ha tocado la pantalla está en las coordenadas especificadas
      Entonces llama al método setScreen(MainMenuScreen(game))
    Fin Si
  Fin Si
Final For

8.5. Caso de uso "Seleccionar idioma"

8.5.1. Nombre: setScreen(levelsScreen(game,language))

Este método se encuentra definido en la clase *AndroidGame*. Es similar al explicado en el Caso de Uso "Jugar" pero esta vez se necesita un parámetro más por lo que es conveniente que se explique. Por lo tanto, este método se encargará de establecer la nueva pantalla a pintar, levelsScreen.

Si levelsScreen no es nula
  Pausa la pantalla actual → LanguageScreen
  Libera la pantalla actual → LanguageScreen para dejar sitio a la nueva instancia
  Si el intervalo de tiempo es 0
    Entoncs retoma la actividad y se actualiza la pantalla
  Fin Si
  Establece la nueva pantalla a pintar indicando además el idioma seleccionado (1- Castellano, 2- Inglés, 3- Euskera)
Fin Si

8.6. Caso de uso "Seleccionar nivel"

8.6.1. Nombre: getScore(level,language): time,attemps

Cuando el jugador selecciona un nivel en la clase *levelsScreen*, esta automáticamente obtiene la puntuación correspondiente a ese nivel de la clase *AndroidGame*.

Por cada nivel e idioma
Está almacenado el tiempo, intentos, puntuación y estrellas en un archivo dentro de la carpeta de instalación del juego
Al seleccionar nivel llama al método getScore(level,language) de la clase *AndroidGame*
(El valor del nivel es un entero entre el 1 y el 8, siendo uno el primer nivel y 8 el último nivel y el valor del lenguaje otro entero entre el 1 y el 3 siendo uno para castellano, dos para inglés y tres para euskera)
Por cada nivel y lenguage
Obtiene los intentos → getInt(level + “-” + language + “intentos”, 0) y lo guarda en la variable myAttemps
Obtiene el tiempo → getString(level + “-” + language + “tiempo”, “”) y lo almacena en la variable myTime
Fin For
Fin For

8.6.2. Nombre: forth(x,y): touch

El método forth(x,y) es igual que los métodos *back(x,y)* y *touch(x,y)*. En este caso la clase *levelsScreen* se encarga de saber si las coordenadas de la pulsación están dentro del rango, y en caso afirmativo, avanza a la clase *levelsScreen2*.

Crea una lista de eventos → <TouchEvent>touchEvents en la cual se almacenarán todos los eventos detectados.
Por cada evento:
Si el evento es de tipo TOUCH_UP
Calcula la zona dónde se encuentra el rectángulo del botón ayuda → Con x =396 e y=726, calcula la esquina superior izquierda del rectángulo y con el ancho=72 y el alto=63, calcula la esquina inferior derecha del rectángulo.
Si la zona en la que ha tocado la pantalla está en las coordenadas especificadas
Entonces llama al método setScreen(LevelsScreen2(game,id))
Fin Si
Fin Si
Final For

8.6.3. Nombre: saveScores(level, language, attemps, time, score, stars)

Cuando se completa el nivel, la clase *GameScreen* llama al método putX, siendo X el tipo de dato que se quiere almacenar, de la clase *SharedPreferences*, implementada por Android, para almacenar la información correspondiente.

Si el subnivel es igual a 20
  Por nivel y lenguage
    Se guardan los intentos → putInt(level + "-" + language + "intentos", attempTotal)
    Se guarda el tiempo → putLong(level + "-" + language + "tiempo", crono. getBase() SystemClock.elapsedRealtime())
    Se guarda la puntuación total → putInt(level + "-" + language + "puntuacion", scoreFinal)
    Se guardan las estrellas conseguidas → putString(level + "-" + language + "estrellas", starsFinal)
  Fin For
Fin Si

8.7. Caso de uso "Seleccionar subnivel"

8.7.1. Nombre: getLast(level,language):last

Antes de dibujar la pantalla de subniveles, la clase *SublevelsScreen* utiliza este método de la clase *AndroidGame* para obtener el último subnivel desbloqueado.

Obtiene el estado del primer subnivel getInt(level+-subLevel+-language, 0) y lo almacena en la variable inMyValue *(0 si no está completado y 1 si lo está)*
Mientras inMyValue sea igual a 1
  subnivel++
  Obtiene el estado del subnivel
Fin Mientras
Devuelve el último subnivel desbloqueado

8.7.2. Nombre: getImage(level,sublevel):image

Por cada nivel, en la clase *GameScreen* hay declarado un array del tipo Image el cual ha sido definido en la clase *Image* que contiene la ruta de la imagen, el nombre y la pista, ambos en cada idioma. Este método, se encarga de obtener la ruta de la imagen a adivinar.

Llama desde la clase *GameScreen* al método getImage() de la clase *Image* indicando el nivel y el subnivel
Devuelve la ruta de la imagen

8.7.3. Nombre: getTip(level,sublevel,language):tip

Este método es similar al anterior, pero en este caso se encarga de obtener la pista en el idioma seleccionado de la imagen a adivinar.

Si ha sido pulsado el botón de pista
  Si el idioma seleccionado es el castellano
    Llama al método getTipEs() de *Image*
    Devuelve la pista en castellano
  Sino Si el idioma seleccionado es el inglés
    Llama al método getTipEn() de *Image*
    Devuelve la pista en inglés
  Sino si el idioma seleccionado es el euskera
    Llama al método getTipEu() de *Image*
    Devuelve la pista en euskera
  Fin Si
Fin Si

8.7.4. Nombre: timeToSeconds(chrono):secondsTime

Esta función situada en la clase *GameScreen* se encarga de convertir el string del cronómetro que indica el tiempo empleado en el nivel a segundos.

Declara e inicializa secondsTime a 0
Almacena en un array de Strings el chrono eliminando : con el método de java split().
Si el array es de longitud 2
    Entonces Posición 0 del array (minutos) * 60 + posición 1 el array (segundos).
Sino Si el array es de longitud 3
    EntoncesPosición 0 del array (horas) * 60 * 60 + posición 1 el array * 60(minutos) + posición 2 del array (segundos).
Fin Si
Devuelve secondsTime correspondiente a los segundos empleados para superar el nivel

8.7.5. Nombre: calculateScore(attemps,secondsTime): score

Esta función de la clase *Scores*, es la encargada de calcular la puntuación total obtenida en un nivel a partir de los intentos y el tiempo empleado en dicho nivel indicados desde la clase *GameScreen*.

Declara scoreAttemps
Declara scoreTime
Declara score
Almacena en scoreAttemps 200.000/attemps
Almacena en scoreTime 2.550.000/secondsTime
Almacena en score scoreAttemps+scoreTime
Devuelve score, es decir, la puntuación final conseguida en el nivel

8.7.6. Nombre: calculateStars(scoreFinal):stars

Esta función de la clase *Scores*, se encarga de calcular las estrellas que se corresponden con la puntuación final conseguida y a partir del string que devuelva, la clase *GameScreen* se encargará de pintar las estrellas correspondientes.

```
Si score <= 2784
   Devuelve “a”
Sino Si (2785<=score)&&(score<=4819)
   Devuelve “b”
Sino Si (4820<=score)&&(score<=6854)
   Devuelve “c”
Sino Si (6855<=score)&&(score<=8889)
   Devuelve “d”
Sino Si (8890<=score)&&(score<=10924)
   Devuelve “e”
Sino Si (10925<=score)&&(score<=12959)
   Devuelve “f”
Sino
   Devuelve “g”
Fin Si
```

8.7.7. Contrato: showScore(level, language, time ,attemps, score, stars)

Este método situado en la clase *GameScreen*, se encarga de mostrar las puntuaciones del nivel al Jugador. Para ello, utiliza antes las tres funciones explicadas previamente. Una vez tiene todos los datos, le mostrará al usuario una pantalla en la que aparece un título en el lenguaje correspondiente indicando el nivel completado, los intentos, el tiempo, la puntuación total y las estrellas conseguidas.

Dibuja el tiempo con la función de java → setText(chronoText)
Dibuja los intentos con la función de java → setText(attempTotal)
Dibuja la puntuación total con la función de java → setText("Puntos: -SscoreFinal)
Si el string devuelto por la función calculateStars() es una "a"
  Dibuja cero estrellas completadas con la función de java → setImageResource(R.drawable.a))
Sino Si el string devuelto por la función calculateStars() es una "b"
  Dibuja media estrella completada con la función de java → setImageResource(R.drawable.b))
Sino Si el string devuelto por la función calculateStars() es una "c"
  Dibuja una estrella completada con la función de java → setImageResource(R.drawable.c))
Sino Si el string devuelto por la función calculateStars() es una "d"
  Dibuja una estrella y media completadas con la función de java → setImageResource(R.drawable.d))
Sino Si el string devuelto por la función calculateStars() es una "e"
  Dibuja dos estrellas completadas con la función de java → setImageResource(R.drawable.e))
Sino Si el string devuelto por la función calculateStars() es una "f"
  Dibuja dos estrellas y media completadas con la función de java → setImageResource(R.drawable.f))
Sino
  Dibuja tres estrellas completadas con la función de java → setImageResource(R.drawable.g))
Fin Si

8.8. Caso de uso "Obtener Puntuaciones"

8.8.1. Contrato: getScores(level,language): time, attemps, score, stars

Este método sito en la clase *ScoresActivity*, se encarga de obtener las puntuaciones relacionadas con el nivel y el lenguaje especificado.

Si el nivel ha sido completado (=1)
Obtiene en myAttemps los intentos con el método de java → getInt(level + “-” + language + “intentos”, 0)
Obtiene en myTime el tiempo empleado con el método de java → getString(level + “-” + language + “tiempo”, “ ”)
Obtiene en myScore la puntuación total con el método de java → getInt(level + “-” + language + “puntuacion”, 0)
Obtiene en myStars las estrellas conseguidas con el método de java → getString(level + “-” + language + “estrellas”, “ ”)
Sino Si el nivel no ha sido completado (=0)
Si es el lenguage en castellano
Entonces muestra el mensaje “Aún no se han completado los 20 niveles”
Si es el lenguage en inglés
Entonces muestra el mensaje “20 levels are not complete yet”
Si es el lenguage en euskera
Entonces muestra el mensaje “Oraindik ez dira 20 mailak bukatu”
Fin Si
Fin Si

8.8.2. Contrato: showScore(level, language, time, attemps, score, stars)

Este método es el mismo que el del caso de uso de “Seleccionar subnivel” pero a diferencia del anterior, este está situado en la clase *ScoresActivity*. Además tampoco utiliza las tres funciones, timeToSeconds(), calculateScore() y calculateStars() que utilizaba el método anterior ya que todos esos datos se encuentran almacenados en un fichero dentro de la carpeta de instalación del juego.

8.9. Caso de uso “Pausar/Reanudar juego”

8.9.1. Nombre: pauseGame()

En Android hay dos posibles formas de dibujar la pantalla. Una de ellas es la que hemos visto más arriba, con el metodo setScreen() y drawPixmap() o con un nuevo método, setContentView() que lo hace mediante un archivo .XML definido en el directorio *Layout*. Por lo tanto cuando el Jugador pulsa

el botón de Pause del juego, el sistema sabe que debe llamar a este método, el cual pausará el juego mostrando la correspondiente pantalla y guardará el estado actual del juego.

Carga la pantalla de pause mediante el método definido en java → setContentView(R.layout.pause), donde pause es el archivo .XML. **Almacena** el tiempo empleado en el nivel en la variable timeWhenStopped **Para** el cronómetro **Almacena** el nivel actual en currentLevel **Almacena** el subnivel actual en currentSublevel

8.9.2. Nombre: resumeGame(currentLevel, currentSublevel, language, timeWhenStopped, attemps)

Este método situado en la clase *GameScreen*, simplemente lo que hace es poner en marcha de nuevo el juego. Para ello, carga nuevamente la Activity *GameScreen* con los datos guardados con el anterior método.

Llama a la clase *GameScreen* indicándole el nivel, el subnivel, el idioma seleccionado, el tiempo transcurrido y los intentos.

8.9.3. Nombre: goHome(): mainMenuScreen

La otra posibilidad que tiene el Jugador cuando está el juego en pausa es volver al menú principal. En este caso, el juego deberá guardar todos los datos en el fichero correspondiente y al mismo tiempo llamar a la clase *MainMenuScreen* para que pinte la pantalla del menú principal. goHome() utiliza una Activity auxiliar *secondIntermediateActivity* para llamar a esta clase.

Llama a la Activity *secondIntermediateActivity*
Si el sonido está activado
    Para el sonido del nivel correspondiente
Fin Si
Guarda los intentos empleados hasta ese momento en el nivel y lenguaje con el método de java visto también anteriormente → putInt(level + “-” + language + “intentos”, attempTotal)
Guarda el tiempo transcurrido hasta ese momento en el nivel y lenguaje con el método de java visto también anteriormente → putLong(level + “-” + language + “tiempo”, crono.getBase() SystemClock.elapsedRealtime())
Para el cronómetro
Guarda la puntuación total conseguida hasta ese momento en el nivel y lenguaje → putInt(level + “-” + language + “puntuacion”, scoreFinal)
Guarda las estrellas conseguidas hasta ese momento en el nivel y lenguage → putString(level + “-” + language + “estrellas”, starsFinal)
Llama a la clase *MainMenuScreen*

8.10. Caso de uso “Comprobar”

8.10.1. Nombre: touchCheck(x,y):touch,text

Este método de la clase *GameScreen*, evalua la pulsación del usuario sobre el botón de comprobar. Este tipo de detectar si el usuario ha tocado la pantalla está programado de manera diferente al resto de los touch() explicados. En este caso, está implementado con un listener, el cual está relacionado con el archivo main.xml del directorio Layout. La clase *GameScreen* se encarga de saber con el listener si se ha producido la pulsación del botón, y en caso afirmativo, pasara a comprobar la respuesta.

Si ha sido pulsado el botón btnShowMessage
    Entonces llama al método checkAnswer()
    Suma uno al número de intentos → attempTotal++;
    Muestra de manera actualizada los intentos con el método de java → setText(attempTotal)
Fin Si

8.10.2. Nombre: toReplaceChar(spaceText): newText

Este método, situado en la clase *GameScreen*, se encarga de covertir la palabra introducida por el Jugador a código ascii debido a que a la hora de comprobarla con la solución correcta se producen errores. Esto tiene lugar una vez que en la palabra introducida han sido reemplazadas las mayúsculas por minúsculas y eliminados los caracteres en blanco tanto del comienzo como del final de la palabra. Esto se hace mediante dos métodos de la librería de Java ya explicados en el capítulo de análisis. Dichos métodos son: toLowerCase y trim().

Declara e inicializa en la variable original el siguiente string "áàäéèëîïóòöúùuñÁÀÄÉÈËÍÌÏÓÒÖÚÙÜÑçÇñ"
Declara e inicializa en la variable ascii el siguiente string "aaaeeeiiiooouuunAAAEEEIIIOOOUUUNcCn"
Por cada caracter del string original
    Reemplaza cada caracter del string ascii por cada caracter del string original y lo almacena en la variable newText
Fin Por
Devuelve newText

8.10.3. Nombre: getName(level,sublevel,language): name

Este método, situado en la clase *Image*, es similar a getImage() y getTip(). Cuando el Jugador pulsa sobre la opción de comprobar respuesta de la clase *GameScreen*, se encarga de obtener la respuesta correcta del subnivel en el lenguaje seleccionado.

Si el idioma seleccionado es el castellano
  Llama al método getNameEs() de *Image*
  Devuelve la respuesta correcta en castellano
Sino Si el idioma seleccionado es el inglés
  Llama al método getNameEn() de *Image*
  Devuelve la respuesta correcta en inglés
Sino si el idioma seleccionado es el euskera
    Llama al método getNameEu() de *Image*
  Devuelve la respuesta correcta en euskera
Fin Si

8.10.4. Nombre: checkAnswer(text,name):string

Este método de clase *GameScreen* se encarga de comprobar si la respuesta introducida por el Jugador es correcta, para ello tendrá que utilizar más métodos explicados anterior o posteriormente.

Si ha sido pulsado el botón btnShowMessage
  Si el idioma seleccionado es el castellano
    Llama al método getNameEs() explicado anteriormente
    Comprueba la respuesta de getNameES() con la introducida por el usuario correcta en castellano
  Sino Si el idioma seleccionado es el inglés
    Llama al método getNameEn() explicado anteriormente
    Comprueba la respuesta de getNameEn() con la introducida por el usuario correcta en inglés
  Sino si el idioma seleccionado es el euskera
    Llama al método getNameEu() explicado anteriormente
    Comprueba la respuesta de getNameEu() con la introducida por el usuario correcta en euskera
  Fin Si
Fin Si
Si la respuesta es correcta
  Guarda el subnivel completado → saveSublevel()
  Si el subnivel es distinto de 20
    Entonces muestra en pantalla el mensaje: {CORRECTO/RIGHT/-ZUZENA} dependiendo del idioma seleccionado
    Pasa al siguiente subnivel → nextSublevel()
  Sino si el subnivel es igual que 20
    Muestra las puntuaciones → showScores()
    Muestra en pantalla un mensaje de enhorabuena dependiendo del idioma seleccionado
    Almacena las puntuaciones → saveScores()
  Fin Si
Sino si la respuesta no es correcta
  Entonces muestra en pantalla el mensaje: {INCORRECTO/WRONG/EZ DA ZUZENA} dependiendo del idioma seleccionado
Fin Si

8.10.5. Nombre: saveSublevel(level, sublevel, language, complete)

Cada vez el que el jugador completa un subnivel, se llama a este método de la clase *GameScreen*. Se encarga de guardar en el fichero, qué subnivel de qué nivel y en qué idioma ha sido completado escribiendo un 1.

Si la respuesta es correcta
Entonces crea un objeto del fichero
Guarda en el fichero el subnivel completado indicando también el nivel y el lenguaje, con la función de java → putInt(level+“-”+(sublevel+1)+“-”+language, 1)
Fin Si

8.10.6. Nombre: nextSublevel(level,sublevel,source): image

Al igual que el anterior método, cada vez el que el jugador completa un subnivel, se llama a este método de la clase *GameScreen*. Se encarga de mostrarle al usuario el siguiente subnivel a completar.

Si la respuesta es correcta
Si no es el subnivel 20
Entonces actualiza el subnivel → sublevel++
Llama al método getImage() de la clase *Image* explicado anteriormente para obtener la nueva imagen y mostrarla con el método de java → setImageResource()
Borra el texto de la respuesta introducida por el usuario en el subnivel anterior
Fin Si
Fin Si

Capítulo 9

Implementación

En este capítulo se incluyen diversos detalles de la implementación del sistema detallando la forma en la que se decidió desarrollar la aplicación. Se comienza explicando el funcionamiento de las aplicaciones Android en general para poner al lector en situación. Incluye la descripción de las clases implementadas y un diagrama con todas ellas, se explica la estructura de todos los ficheros utilizados y demás aspectos que han llevado a la consecución del producto final.

9.1. Bloques de construcción

Las aplicaciones en Android se construyen basándose en bloques de construcción básicos. Evidentemente, no todas las aplicaciones tienen que tener todos. A continuación se explican los bloques más importantes:

9.1.1. Actividades

Un *Activity* puede ser considerado como uno de los bloques de construcción más utilizado. Se trata de una pantalla de la interfaz de usuario. Una aplicación puede definir una o más actividades para controlar el flujo del programa manejando sus distintas fases. Cada actividad tiene su propio ciclo de vida por lo que esto facilita la recuperación de cada *activity* cuando mejor convenga.

Pero en este punto surge un problema. Si el dispositivo Android considera oportuno puede priorizar o quitar una actividad de la pila de memoria por cuestión de falta de recursos de memoria. Esto quiere decir, que si por ejemplo, se minimiza la aplicación y no se han guardado las puntuaciones o el subnivel completado hasta el momento cabe la posibilidad que se pierda dicha información.

Para solucionar esto, cuando el Jugador minimice el juego se guardará el subnivel en el que se encuentra, el tiempo empleado, los intentos y la puntuación. De esto modo, si el dispositivo cerrara la aplicación, todos los cambios estarían guardados.

9.1.2. Intenciones

Un *Intent* es un mecanismo para describir una acción específica. En Android, prácticamente todo funciona por medio de intenciones.

Las intenciones han sido utilizadas para comunicar las diferentes actividades de las que consta el juego. Por ejemplo, para pasar de la pantalla de los subniveles a lo que es el juego en sí, se utiliza un *Intent* para realizar la acción de ejecutarlo, indicandole el idioma en el que se encuentra así como el nivel y el subnivel correspondiente.

9.1.3. AndroidManifest

Se trata de un archivo XML que controla el funcionamiento del sistema. Es el que le dice al sistema que debe de hacer con los componentes de alto nivel creados, por ejemplo con las Actividades e Intentos explicados previamente. Es decir, *AndroidManifest.xml* es considerado como la unión entre ambos ya que indicará qué *Intent* ha sido recibido por qué *Activity*.

En el caso de Guess the Pic, se han definido en este fichero las siguientes actividades.

- **SplashActivity**: Que es la encargada de lanzar una pantalla inicial con el mensaje "cargando" mientras se cargan los recursos al iniciar el juego.
- **GuessThePicActivity**: Se encarga de iniciar el juego cargando la pantalla de menú principal.

- **GameScreen**: Es el activity principal. Se encarga de mostrar la pantalla en la que el usuario juega, pintando la imagen que el usuario tiene que adivinar.
- **ReturnLevelsScreenActivity**: Es un activity auxiliar utilizado para volver a la pantalla de selección de nivel desde la pantalla de puntuaciones.
- **ReturnMainMenuActivity**: Se trata de otro activity auxiliar encargado de volver al menú principal, al pulsar el botón de home cuando el usuario pausa el juego.
- **ScoresActivity**: Por último, esta actividad es la encargada de mostrar las puntuaciones de cada nivel.

9.2. Ciclo de vida de una actividad

Una vez explicados los bloques de construcción básicos de Android conviene describir cómo es el ciclo de vida de una actividad[1] para entender el funcionamiento de una aplicación.

En la siguiente figura 9.1 se muestra dicho ciclo de vida.

[1] `http://androideity.com/2011/07/06/ciclo-de-vida-de-una-actividad/`

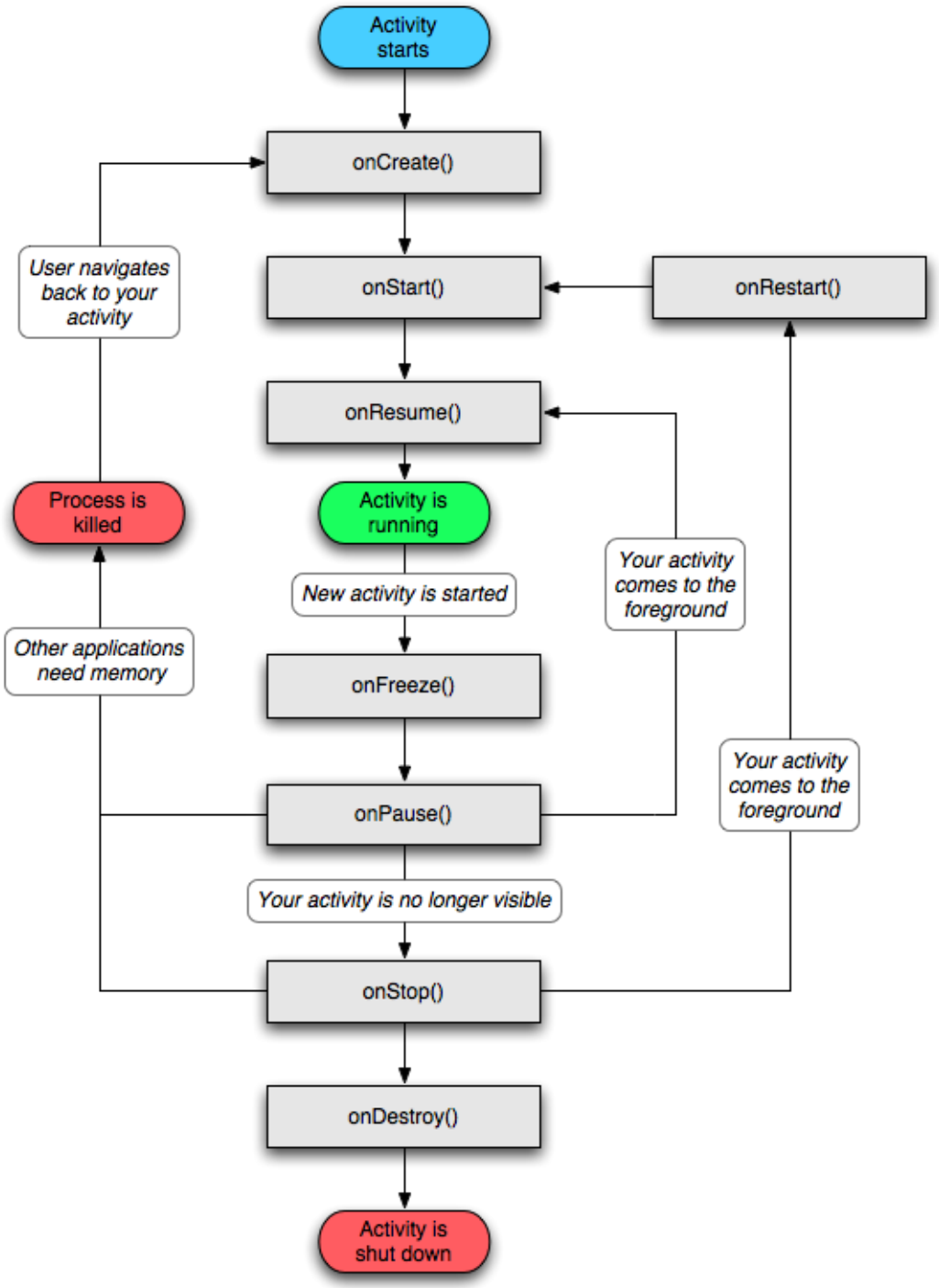


Figura 9.1: Ciclo de vida de una actividad Android

A continuación, se explican cada uno de los métodos.

- **onCreate():** Se dispara cuando la *Activity* es llamada por primera vez. Aquí es donde se debe crear la inicialización normal de la aplicación, crear vistas, etc. Este método da acceso al estado de la aplicación cuando se cerró. Después de esta llamada siempre se llama al onStart(). Este método se ha utilizado para realizar todo lo que queríamos que se hiciera en el Activity. Por ejemplo, en el método onCreate de GameScreen, es donde se realiza toda la lógica del juego, la comprobación de las palabras, cambiar de subnivel, gestionar el tiempo, etc.
- **onRestart():** Se ejecuta cuando la *Activity* ha sido parada, y se desea volver a utilizarla.
- **onStart():** Se ejecuta cuando la *Activity* se está mostrando en la pantalla del dispositivo del usuario.
- **onResume():** Se ejecuta una vez que la *Activity* ha terminado de cargarse en el dispositivo y el usuario empieza a interactuar con la aplicación. Cuando el usuario ha terminado de utilizarla es cuando se llama al método onPause(). Este método se ha utilizado básicamente para reanudar la música una vez que se maximizaba el juego.
- **onPause():** Se ejecuta cuando el sistema arranca una nueva *Activity* que necesitará los recursos del sistema centrados en ella. Hay que procurar que la llamada a este método sea rápida ya que hasta que no se termine su ejecución no se podrá arrancar la nueva *Activity*. Después de esta llamada puede venir un onResume() si la *Activity* que haya ejecutado el onPause() vuelve a aparecer en primer plano o un onStop() si se hace invisible para el usuario. Del mismo modo, en Guess the Pic, se ha utilizado este método para pausar el sonido cuando se minimizaba el juego.
- **onStop():** Se ejecuta cuando la *Activity* ya no es visible para el usuario porque otra *Activity* ha pasado a primer plano. Si se observa el diagrama, después de que se ha ejecutado este método quedan tres opciones: ejecutar el onRestart() para que la *Activity* vuelva a aparecer en primer plano, que el sistema elimine este proceso porque otros procesos requieran memoria o ejecutar el onDestroy() para apagar la aplicación.
- **onDestroy():** Esta es la llamada final de la *Activity*, que quedará totalmente destruida despues de llamar al método. Esto pasa por los requerimientos de memoria que tenga el sistema o porque de manera explícita el usuario manda a llamar este método. Si se quisiera volver

a ejecutar la *Activity* se arrancaría un nuevo ciclo de vida.

9.3. Estructura de ficheros

En este apartado se describe como se encuentra estructurado el código. Para ello, se explica cómo están distribuidos los ficheros dentro del workspace de la plataforma Eclipse, qué tipo de archivos se encuentran en cada carpeta y cómo y dónde se pueden encontrar esas carpetas.

En la siguiente figura 9.2, se puede observar cómo está estructurado el proyecto.

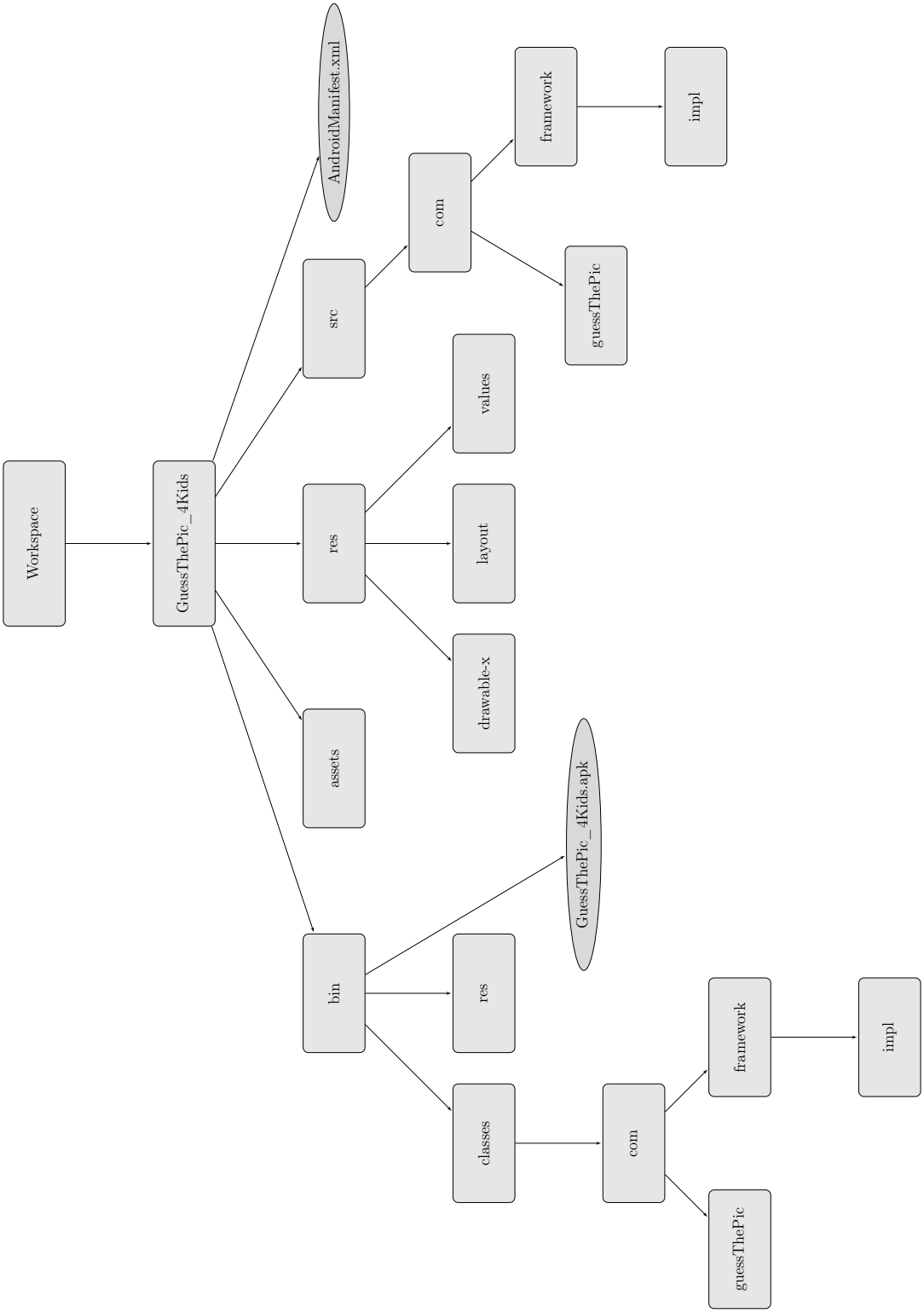


Figura 9.2: Estructura de los ficheros del proyecto

A continuación, se explica qué son cada una de ellas, que contienen y cuál es su función, en caso de tenerla.

Dentro del **Workspace**, se encuentra el proyecto creado para implementar esta aplicación, **GuessThePic_4Kids**. A su vez, dentro de esta carpeta, se encuentran cuatro directorios importantes y un archivo .xml.

El primero de ellos es **bin**. Esta carpeta es importante porque en ella se encuentra un archivo fundamental, el .apk del juego, es decir, el instalador. Cada vez que se compile la aplicación se guarda en esta carpeta. Dentro, se encuentran otras dos carpetas que son generadas automáticamente. En **classes** se encuentra la carpeta **com** y las subcarpetas *guessThePic* y *framework*. En ambos subdirectorios se dividen todos los archivos binarios que componen el juego. En **guessThePic** están todos los archivos binarios de las clases del modelo vista mientras que en **framework**, están los archivos binarios de las interfaces usadas en la capa de control del juego. Dentro de *framework*, hay otra carpeta, **impl**, que contiene los archivos binarios de las implementaciones de estas interfaces. Por otro lado, en **res**, están las imágenes de los iconos. No obstante, el contenido de esta carpeta se explica más adelante cuando se hable del directorio con el mismo nombre pero en el nivel superior.

El segundo directorio es **assets**, que es la carpeta en la que se guardan los archivos que necesita la aplicación, como por ejemplo, los de configuración, audio o imágenes. Todos ellos se empaquetan junto con la aplicación.

Por otro lado, se encuentra el directorio **res**, que contiene todos los recursos necesarios, como iconos o diseños para la interfaz del usuario que se definen a través de XML. Esta carpeta, cuenta a su vez con diferentes directorios, llamados **drawable-hdpi**, **drawable-ldpi**, etc, uno por cada tamaño de resolución posible; en los que se incluyen dichos iconos e imágenes con tamaños diferentes para usarlos en dispositivos con la resolución de la pantalla distinta. También contiene la carpeta **layout**, que como su propio nombre indica incluye los distintos layouts de la aplicación. Los layouts son elementos no visibles que permiten indicar la distribución, posicionamiento y dimensión de los diferentes elementos que se encuentran en su interior. Cada uno de estos elementos tiene un identificador único que se debe especificar, y a partir de él se adjudicará un entero único. Este procedimiento será aplicado a cada uno de los elementos guardados en esta carpeta. Al generar un proyecto se nos crea una actividad, y junto a ello un layout. Los identificadores de elementos dentro de un layout deben tener la sintaxis que se muestra en la figura 9.3:

Del mismo modo, para cargar un layout se debe hacer una llamada al método

```
1        <ImageView
2                android:id="@+id/clock"\backslash>
```

Figura 9.3: Declaración de identificadores en un Layout

setContentView() al crear una actividad (en el método onCreate()). A este método se le debe pasar la referencia de ID del layout a instanciar. Por ejemplo, si el nombre del archivo de layout es main.xml, se deberá hacer lo mostrado en la figura 9.4:

```
1        public void onCreate(Bundle savedInstanceState) {
2            super.onCreate(savedInstanceState);
3            setContentView(R.layout.main);
4        }
```

Figura 9.4: Ejemplo de cómo instanciar un Layout

Pero si lo que se quiere es obtener una referencia del botón cuando se ejecuta la aplicación, se debe usar la función findViewById y pasarle el id del elemento, como se muestra a continuación, en la figura 9.5:

```
1        public void onCreate(Bundle savedInstanceState) {
2            super.onCreate(savedInstanceState);
3            setContentView(R.layout.main);
4            ImageView clock = (ImageView) findViewById(R.id.clock
                 );
5        }
```

Figura 9.5: Ejemplo de cómo instacionar un elemento del Layout

Para esta aplicación, han sido necesarios cinco layout diferentes:

- **activity_guess_the_pic.xml:** Se trata del layout de la actividad inicial del juego.
- **main.xml:** Es el layout de la pantalla principal del juego. El que dibuja la imagen a adivinar, el tiempo, los intentos, el botón de pause, etc.
- **pause.xml:** Es el layout correspondiente a la pantalla de pause del juego.
- **score.xml:** Se trata del layout encargado de pintar la pantalla de las puntuaciones.

- **splash.xml:** Es el layout que se encarga de pintar la pantalla de la actividad *SplashActivity*.

Y por último, dentro del directorio *res*, se encuentra **values**. En esta carpeta, es donde se predefinen los valores de la aplicación en formato de archivos XML, definiendo los nombres de las variables y sus valores, que serán referenciados más tarde en el código. Estos valores, por ejemplo, pueden ser cadenas de caracteres e incluso constantes. Se puede entender esta carpeta como el único lugar que contiene todos los valores constantes de una aplicación. De esta manera, si necesitan ser ajustados durante las fases de desarrollo y de pruebas, puede hacerse en un solo lugar. Los tres archivos que han sido definidos en la carpeta values son:

- **dimens.xml:** Archivo XML que define las dimensiones, tanto alturas como tamaños de fuentes de la interfaz de usuario.
- **strings.xml:** Archivo XML que define cadenas de texto usadas en la aplicación. Por ejemplo, para colocar los títulos de las ventanas o el nombre de la aplicación.
- **styles.xml:** Archivo XML que define los estilos usados en la aplicación. Estos estilos pueden ser aplicados a los elementos de la interfaz de usuario, de modo que esté separado la plantilla de las funcionalidades.

El cuarto directorio es **scr**. Contiene todos los archivos fuente de Java. Tiene el mismo contenido que el *src* de *bin*. La única diferencia es que estos archivos son de tipo .java y los de *bin* son de tipo .class.

Por último, a la misma altura que los directorios explicados anteriormente, se encuentra el fichero **AndroidManifest.xml**. Este archivo describe la aplicación. Es decir, define las actividades y servicios que la componen, la versión mínima de Android en la que se supone que funcionará y los permisos que necesitará, se declara la orientación de la pantalla de la aplicación, etc.

9.4. Manual de implementación

En este apartado se explican las diferentes clases de las que consta el juego.

9.4.1. Clases del controlador

En la figura 9.6, se aprecia el diagrama de clases de la parte controlador.

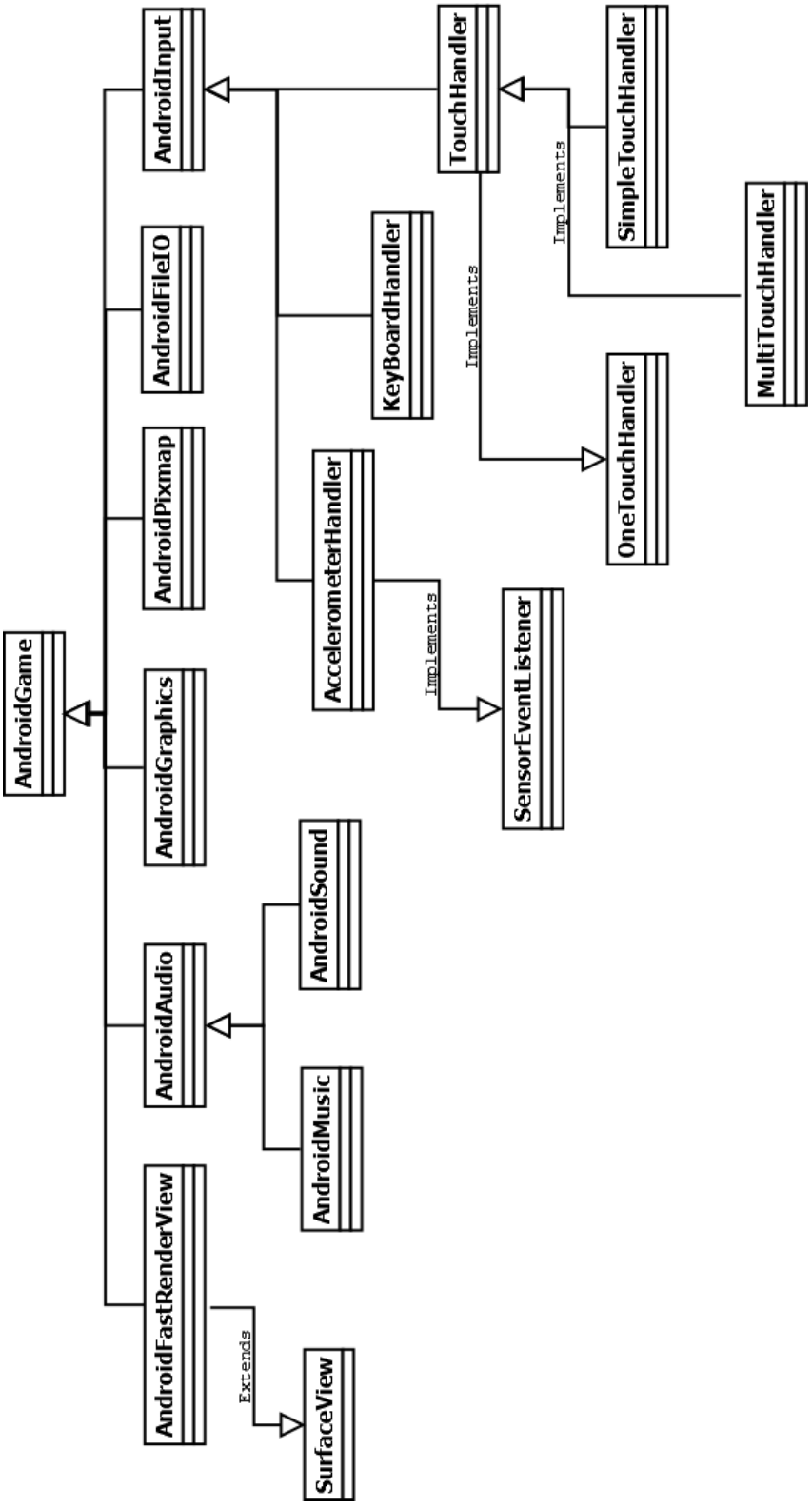


Figura 9.6: Diagrama de clases de la parte controlador

La primera clase que se ha creado es **AndroidGame**, la cual extiende de la clase *Activity*, de este modo hereda los métodos onCreate(), onResume() y onPause() explicados en el apartado anterior. La clase *AndroidGame*, es el motor del juego y la que gestiona todas las clases de control de imágenes, sonidos y demás. La clase *AndroidGame* implementa la interfaz *Game*. Se encargará de la gestión de la ventana, esto implica que hay que configurar una actividad y una vista **AndroidFastRenderView** y hay que hacerse 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 e Input a las partes interesadas, gestiona los Screen y los integra en el ciclo de vida de la actividad.

En toda aplicación, el objetivo es tener una única clase *AndroidGame*, de la cual se pueda obtener todos los elementos que sean necesarios.

Por otro lado están las clases **AndroidAudio**, **AndroidSound** y **AndroidMusic** las cuales 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 guarda en SoundPool y obtiene una instancia de *AndroidSound*. En *AndroidAudio* se almacena 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. Si el teléfono no tiene acelerómetro, el controlador mostrará que la aceleración de todos los ejes es 0.

La clase **SingleTouchHandler** implementa **TouchHandler**, la cual es una pequeña clase que a su vez implementa a **OnTouchListener**, y hace de elemento cohesionador entre *SingleTouchHandler* y *MultiTouchHandler*. A continuación, se toma nota del estado de la pantalla táctil para un dedo y de dos listas que obtienen los eventos TouchEvent. También están los métodos ScaleX y ScaleY responsable de adaptar la lectura de los eventos del dedo a todas las resoluciones de pantalla posibles.

Por otro lado, la clase **MultiTouchHandler**, al igual que *SingleTouchHandler*, implementa a *TouchHandler*, por lo que tiene la posibilidad de almacenar

datos de los dedos y los eventos. En vez de almacenar el estado de un único puntero, lo hace hasta de 20. También contiene los eventos de ScaleX y ScaleY, explicados en el párrafo anterior. El método es parecido al *SingleTouchHandler*, pero en este caso también se guarda 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.

El controlador del teclado **KeyboardHandler** se encarga de un par de tareas. En primer lugar, debe hacerse cargo de la vista View a través de la cual se recibirán los eventos del teclado. Luego, deberá tomar nota del estado de cada tecla guardándola en un pool. También ha de llevar un registro de las instancias KeyEvent que se hayan usado. Por último, deberá sincronizarlo todo porque estará recibiendo eventos desde la interfaz del usuario mientras los almacena en el pool del juego, que se estará ejecutando en otro hilo diferente.

Una vez definidos estos cuatro últimos controladores, se puede explicar la clase **AndroidInput**. Esta clase reune 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 única responsabilidad que se tiene a la hora de implementar el juego es la de *TouchHandler*, que usando como referencia la versión de Android del dispositivo, delegará el control a *SingleTouchHandler* o a *MultiTouchHandler*. *SingleTouch* se utilizará si el dispositivo en el que se está ejecutando el juego sólo puede procesar un evento a la vez. *MultiTouch* será utilizado en caso contrario.

La clases encargadas de los gráficos del juego son **AndroidGraphics** y **AndroidPixmap**. La primera de ellas dibuja pixels, líneas, rectángulos y pixmap en el framebuffer. Se puede recurrir al Bitmap como un framebuffer virtual al que dirigir todas las llamadas relacionadas con el dibujo a través de la clase *Canvas*. En mayor medida se usa para generar instancias de pixmap procedentes de archivos de recursos. Estos pixmap se dibujan mediante el método drawPixMap(), el cual dibuja todo el pixmap en el framebuffer virtual en las coordenadas que se le especifiquen. Las clases getWidth() y getHeight() devuelven el tamaño del frame buffer virtual de la instancia de *AndroidGraphics*. En cuanto a *AndroidPixmap*, almacena una instacia de Bitmap y ofrece información sobre el alto, el ancho y el formato de un Pixmap.

La clase **AndroidFastRenderView**, la cual extiende de la clase **SurfaceView**, se encarga de usar un hilo de ejecución independiente y dibujar de forma continua. Este hilo ayuda a la ejecución del bucle principal del juego, y se puede configurar para que utilice la clase *Canvas* para dibujar constantemente en *SurfaceView*. Esta clase guarda una referencia a la instancia Game, que se utiliza para obtener el elemento Screen activo. Se llama cons-

tantemente a Screen.update() y Screen.present(), desde el hilo de ejecución *FastRenderView*. Guarda también un registro del intervalo de tiempo que transcurre entre fotogramas, para llevar un control sobre la velocidad del juego, para asegurar que es siempre la misma en cualquier terminal en el que se instale. Y toma también el framebuffer virtual, el cual dibuja la instancia *AndroidGraphics* en *SurfaceView* para modificar su escala si fuera necesario.

La parte controlador del juego está ya casi descrita, sólo queda explicar la clase **AndroidFileIO**. Para el juego es esencial que se pueda escribir y leer desde un archivo, por ejemplo para poder guardar la configuración del sonido. Es muy sencillo, consta de dos métodos diferentes, load() y save() en los que especificándole el nombre de un archivo se obtiene una serie de información. El archivo APK lee todos los recursos y los archivos se leerán y escribirán en la tarjeta SD del dispositivo en el que se esté ejecutando.

Una vez acabado de explicar todas las clases controladoras, se observa que utilizando esta arquitectura, se pueden reutilizar perfectamente todas estas clases para futuros juegos a desarrollar.

9.4.2. Clases de la 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 si misma dependiendo de la pantalla en la que se encuentren.

En la figura 9.7, se puede observar el diagrama de clases de esta parte.

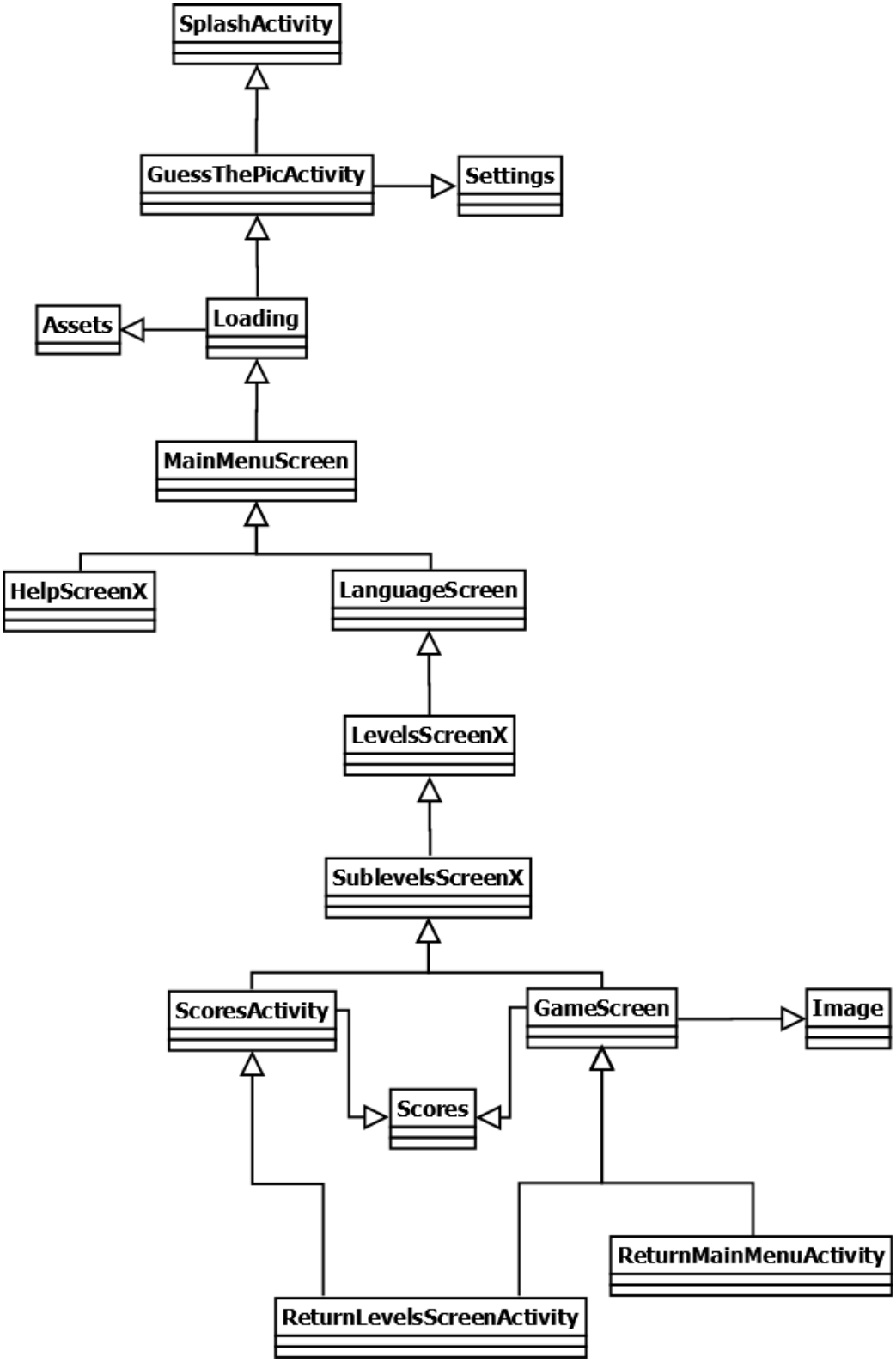


Figura 9.7: Diagrama de clases de la parte vista

En primer lugar, antes de comenzar el juego, la clase **SplashActivity** lanza una pantalla de carga con el propósito de darle tiempo a la aplicación para que cargue los recursos (imágenes, música, etc). Al mismo tiempo, la clase **GuessThePicActivity**, que extiende de la clase *AndroidGame* explicada en el apartado de "Clases del controlador", se encarga de iniciar el juego. Para ello, llama a la clase **Loading**, que es la que carga todos los datos de imágenes y sonido utilizando sus respectivos controladores y que han sido definidos en la clase **Assets**. A continuación, en la figura **??** se puede ver parte del código de la clase *Loading*.

```
1   //Pantalla de selección de lenguaje
2   Assets.languageScreen = g.newPixmap("languageScreen.png",
        PixmapFormat.RGB565);
3
4    //Pantalla de ayuda
5   Assets.helpScreen = g.newPixmap("helpScreen.png",
        PixmapFormat.RGB565);
6
7   //Sonido de felicitación al completar los 20 subniveles
8   Assets.congratulations = game.getAudio().newSound("
        congratulations.ogg");
9
10  //Música principal del juego
11  Assets.song = game.getAudio().newMusic("music.mp3");
```

Figura 9.8: Parte del código de la clase Loading

Una vez cargados todos los recursos, se llama a la clase **MainMenuScreen** que al igual que todas las clases acabadas en "Screen" extiende de la clase *Screen*. Esta clase es la encargada de pintar el menú principal del juego y desde donde se da la posibilidad al usuario de comenzar a jugar. Cada vez que se llame a una clase de este tipo se utilizará el método game.setScreen() que cambiará el estado del Screen actual al screen que se le especifique. Cada una de estas clases tiene sus métodos update() y present() para la 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. Por otro lado, en el método present(), se compone el pixmap que pinta el controlador gráfico.

Desde esta clase, hay tres posibilidades para continuar. La primera de ellas, es la de activar o desactivar el sonido. Para ello, lo único que hará es llamar al método stop() o play() de la clase *Music* y a su vez, cada vez que se modifique se almacenará el estado del sonido en la variable *soundEnabled* en el

archivo de configuración *.guessThePic*, a través de la clase **Settings**. El único propósito de esta clase, es cargar al comienzo del juego el estado del sonido y de guardarlo cuando es modificado. La segunda opción, es la de acceder a la ayuda. Para ello, se cargarán las clases **HelpScreen**, **HelpScreen2**, **HelpScreen3** y **HelpScreen4** respectivamente. Se han definido cuatro clases, una por cada pantalla de ayuda. Estas clases dan la posibilidad de volver a atrás en todo momento, y además, una vez finalizada la explicación de cómo jugar, da la opción de volver al menú principal.

A continuación, en la figura 9.9, se puede ver un ejemplo del código para continuar a la siguiente pantalla o a la anterior pantalla. En concreto, se trata de la tercera pantalla de ayuda, en la que podemos acceder a la segunda o a la cuarta.

```
1   public void update(float deltaTime) {
2       List<TouchEvent> touchEvents = game.getInput().
            getTouchEvents();
3       game.getInput().getKeyEvents();
4
5       int len = touchEvents.size();
6
7       for (int i = 0; i < len; i++) {
8           TouchEvent event = touchEvents.get(i);
9           if (event.type == TouchEvent.TOUCH_UP) {
10
11              //Siguiente
12              if (inBounds(event, 395 ,728 ,72 ,58 )) {
13                  game.setScreen(new HelpScreen4(game));
14                  return;
15              }
16              //Atrás
17              if (inBounds(event, 11, 731, 72, 57)) {
18                  game.setScreen(new HelpScreen2(game));
19                  return;
20              }
21
22          }
23      }
24  }
```

Figura 9.9: Ejemplo de código para continuar a la siguiente o a la anterior pantalla

Por último, se tiene la posibilidad de comenzar el juego. Para ello, antes hay que elegir el idioma en el que se desea jugar. Esto se hace desde la clase **LanguageScreen**. Una vez seleccionado el idioma se cargarán los niveles sitos en las clases **LevelsScreen**, la cual contiene los niveles del 1 al 4,y **LevelsScreen2**, que contiene los niveles del 5 al 8, a los cuales se les ha pasado mediante la variable *language* el idioma seleccionado en la clase *LanguageScreen*. Desde estos dos últimos screen también se puede navegar hacia adelante y atrás. Para ello, lo único que se hace es repintar los backgrounds correspondientes una y otra vez. Cuando se ha seleccionado el nivel se carga la pantalla de los subniveles correspondiente a este nivel. Es decir, hay ocho clases de subniveles, una por cada posible nivel. Para ello, la clase *LevelsScreen* pasará el idioma y el nivel seleccionado a una de las siguientes clases: **SublevelsScreen**, **SublevelsScreen2**, **SublevelsScreen3**, **SublevelsScreen4**, **SublevelsScreen5**, **SublevelsScreen6**, **SublevelsScreen7**, **SublevelsScreen8**. Desde esta clase se puede acceder a las puntuaciones del nivel o seleccionar el subnivel que se desea jugar.

En este punto, se puede decir que es cuando comienza el juego. La clase que tiene toda la lógica es **GameScreen**, la cual extiende de *activity*. En ella se cargan las imágenes a adivinar, se hace la comprobación de las respuestas, se gestionan los subniveles completados, los métodos que pausan y reanudan el juego, etc. Esta clase utiliza otra clase, **Image**, la cual relaciona las imágenes a ser adivinadas con su nombre y pista en los distintos idiomas.

Por otro lado, hay dos clases relacionadas con las puntuaciones. La clase **ScoresActivity** que extiende de *activity* y que es la encargada de mostrar las puntuaciones cuando se accede a través de la clase *SublevelsScreenX*. Y la otra clase, es la de **Scores**, que es la encargada de calcular la puntuación total a partir de los intentos y el tiempo empleado.

Existen otras dos actividades que han sido explicadas en el apartado "Bloques de construcción" y que serán recordadas. La primera de ellas es la de **ReturnLevelsScreenActivity**, que permite volver a la pantalla de selección de nivel al tocar en la pantalla de puntuaciones. Y la otra actividad es **ReturnMainMenuActivity**, que se encarga de volver al menú principal al pulsar el botón de home cuando el usuario pausa el juego en la clase *GameScreen*.

Capítulo 10

Pruebas

Las pruebas son un componente fundamental en todo proceso de desarrollo de software. Su finalidad, más allá de ser la de comprobar que todo funciona correctamente, es la de descubrir errores.

Para ello, en este apartado se diseñarán unos casos de prueba para comprobar el correcto funcionamiento del juego. Además, también se realiza un checklist con el que se observará el comportamiento de 10 niños de distintas características, como por ejemplo su edad, lenguas nativas, sexo, etc.

A continuación, aparecen los diferentes tipos de pruebas que se han llevado a cabo. En todos ellos se han buscado casos de prueba con alta probabilidad de fallo para una depuración lo más completa posible del juego.

10.1. Pruebas unitarias

Se trata de una serie de pruebas que se han ido realizando en cada nivel y subnivel del juego, de tal manera que se pudiera validar si el comportamiento de cada acción se correspondía con el esperado, es decir, pruebas de caja negra.

10.1.1. Pulsar botón "back" del teclado

Resultado esperado

Pulsando en este botón debería permitirnos salir del programa en cualquier momento, parando el sonido y guardando las puntuaciones. El juego se debe cerrar y la única forma de volver, es ejecutándolo de nuevo.

Resultado obtenido

El juego se cierra correctamente, se para el sonido y guarda las puntuaciones.

10.1.2. Pulsar botón "home" del teclado

Resultado esperado

Pulsando en este botón debería minimizar la aplicación dejándola en un segundo plano, parar el sonido y si está en el juego, debe parar también el cronómetro.

Resultado obtenido

El juego se minimiza correctamente, para el sonido y pausa el cronómetro.

10.1.3. Volver al juego

Resultado esperado

Tanto pulsando en el icono del juego, como en la opción de "aplicaciones recientes", la respuesta debería ser la conservación del estado en el que se dejó el juego y la puesta en marcha de nuevo del cronómetro y de la música.

Resultado obtenido

Pulsando sobre el icono del juego, no siempre conserva el estado en el que se dejó, sino que en ocasiones es finalizado por el propio dispositivo. Esto se debe a que el sistema operativo Android cierra las aplicaciones en el caso de

necesitar más recursos, por lo que no se puede hacer nada para solucionarlo. Si se pulsa sobre la aplicación reciente se vuelve al estado en el que se dejó el juego reanudando la música. Se ha detectado que el cronómetro multiplica por dos el tiempo cuando no se ha pausado anteriormente.

Solución

Para solucionar el fallo del cierre de la aplicación no hay solución posible ya que eso depende del sistema operativo en si. Mientras que para solucionar lo de las puntuaciones, cuando se vuelve al juego y por tanto, se carga el tiempo y los intentos, se le resta al tiempo que se carga el tiempo guardado ya que por un motivo desconocido lo suma dos veces.

10.1.4. Pulsar otros botones del teclado

Resultado esperado

Al pulsar cualquier otro botón del juego no debería realizarse ninguna acción que afectara al juego. No obstante, botones como subir o bajar volumen deberían tener un comportamiento normal aunque sin afectar el funcionamiento del mismo.

Resultado obtenido

Al tocar cualquier otro botón no ocurre nada en el juego. Si se sube o baja el volumen del mismo, se observa un correcto funcionamiento.

10.1.5. Pulsar en la pantalla

Resultado esperado

Al pulsar sobre cualquier parte de la pantalla que no tenga ninguna función, a no ser que sea en la de puntuaciones, no deberá realizar ninguna acción.

Resultado obtenido

No ocurre nada, el juego sigue en funcionamiento correctamente.

10.1.6. Pulsar sonido

Resultado esperado

Si el sonido está activado y el Jugador toca sobre este botón el sonido debería desactivarse y viceversa. En ningún momento del juego, el comportamiento del sonido debería cambiar. Es decir, si el sonido está desactivado, no debería sonar ninguna melodía a lo largo del juego.

Resultado obtenido

El sonido se activa y desactiva correctamente cuando se pulsa sobre este botón y no se vuelve a reproducir en el caso de desactivarse. En el caso contrario, no se silencia en ningún momento del juego.

10.1.7. Pulsar "back"

Resultado esperado

A lo largo del juego hay muchas veces en la que aparece está opción. El comportamiento de cada una de ellas debería ser el siguiente:

- En ayuda: En la primera pantalla, debería volver al menú principal, mientras que en las otras tres debería volver a la pantalla anterior de ayuda.
- En la selección de idioma: Debería volver al menú principal.
- En la selección de nivel: En la primera pantalla, debería volver a la selección de idioma y en la segunda, a la primera pantalla de selección de nivel.
- En la selección de subnivel: Debería volver a la pantalla correspondiente de selección de nivel. Es decir, si se elige del nivel 1 al 4, a la primera pantalla; mientras que si se elige del nivel 5 al 8, a la segunda pantalla. Además, deberá cambiar a la melodía principal del juego.

Resultado obtenido

En todos los casos, los resultados obtenidos son los deseados.

10.1.8. Pulsar “next”

Resultado esperado

Del mismo modo, a lo largo del juego también hay muchas veces en la que aparece la opción de navegar hacia adelante. El comportamiento de cada una de las opciones debería ser el siguiente:

- En ayuda: En todas las pantallas de ayuda, exceptuando en la última, deberían ir a la pantalla siguientes de la ayuda. En la última, debería volver al menú principal.
- En la selección de nivel: En la primera pantalla, debería ir a la segunda pantalla mientras que en esta, debería volver al menú principal.

Resultado obtenido

En ambos casos, los resultados obtenidos son los deseados.

10.1.9. Pulsar en un subnivel bloqueado

Resultado esperado

El juego no debe hacer nada, únicamente mostrar un mensaje indicándole al jugador que el subnivel está bloqueado.

Resultado obtenido

Cuando se pulsa sobre un subnivel bloqueado, el juego se comporta de manera correcta, mostrando un mensaje informativo.

10.1.10. Pulsar “pista”

Resultado esperado

Dependiendo del nivel seleccionado y subnivel seleccionado el juego debería mostrar imágenes distintas y al pinchar sobre pista deberíamos obtener la pista correspondiente a ese subnivel.

Resultado obtenido

Se comprueba que la pista obtenida en cada subnivel se corresponde con la imagen y además se obtiene en el idioma seleccionado.

10.1.11. Pulsar "puntuaciones"

Resultado esperado

Cuando el usuario está en la pantalla de selección de subniveles, si pincha sobre la opción de obtener puntuaciones el juego debe comportarse de manera diferente según la situación que se dé:

- Si el jugador no ha completado aún los 20 niveles: Deberá mostrar un mensaje indicándolo sin cambiar de pantalla.
- Si el jugador ha completado los 20 niveles: Deberá mostrarle la mejor puntuación conseguida hasta el momento sin tener en cuenta la actual.

Resultado obtenido

- Si el jugador no ha completado aún los 20 niveles: Comportamiento deseado.
- Si el jugador ha completado los 20 niveles en alguna ocasión: Le muestra la mejor puntuación.

10.1.12. Comprobar las soluciones introducidas

Resultado esperado

Cuando el Jugador introduce su respuesta con las siguientes características:

1. Caracteres en mayúscula: Si la respuesta es correcta, deberá darla por buena.
2. Tildes: Si la respuesta es correcta, deberá darla por buena.
3. En blanco: Respuesta incorrecta.
4. Carácter en blanco al final de la palabra: Respuesta incorrecta.

Resultado obtenido

1. Caracteres en mayúscula: Resultado esperado.
2. Tildes: Resultado esperado.
3. En blanco: Resultado esperado.
4. Carácter en blanco al final de la palabra: Da la solución como incorrecta.

Solución

Añadir el método trim(), el cual elimina cualquier carácter en blanco tanto al principio como al final de un string.

10.2. Pruebas de integración

Estas pruebas se realizan una vez comprobado cada componente por separado y para descubrir los errores que están relacionados con las dependencias existentes entre los diferentes componentes que forman la aplicación. Las pruebas para comprobar la cohesión entre las diferentes partes del juego están basadas en la misma metodología que las unitarias.

10.2.1. Idioma

Resultado esperado

Una vez que el Jugador selecciona el idioma, si se navega por el menú de niveles, subniveles, se completa el nivel y se vuelve a la selección de niveles, el idioma seleccionado debería permanecer.

Resultado obtenido

En todos los casos mencionados anteriormente, el idioma que ha sido seleccionado al comienzo permanece.

10.2.2. Relación entre nivel-subnivel

Resultado esperado

Cuando se selecciona un nivel, el juego debería mostrar los subniveles, desbloqueando aquellos que han sido completados y bloqueando los que no. En el caso de que el nivel haya sido completado, la siguiente vez que se accedería al juego, deberían bloquearse de nuevo los subniveles. Además deberá parar la melodía principal del juego y reproducir la correspondiente al nivel.

Resultado obtenido

El juego bloquea de manera correcta aquellos subniveles, correspondientes al nivel seleccionado, que no han sido completados y reproduce de manera correcta la música del nivel.

10.2.3. Pulsar "pausa"

Resultado esperado

Cuando se pincha sobre pausa, el juego debería mostrar una pantalla con las opciones de reanudar el juego y volver al menú principal, así como debería parar el crono del juego y guardar el estado actual del mismo.

Resultado obtenido

Se pausa tanto el juego como el cronómetro de manera correcta.

10.2.4. Pulsar "reanudar"

Resultado esperado

Una vez que el juego ha sido puesto en pausa, si el jugador desea reanuar el sonido, el juego debería mostrar la pantalla en la que se encontraba, y retomar el cronómetro.

Resultado obtenido

Se muestra de manera correcta la pantalla del juego en la que se encontraba el jugador, pero al igual que ocurría al retomar el juego cuando se minimizaba, si antes no se ha pulsado en pausa, el cronómetro es multiplicado por dos.

Solución

La solución es la misma que al retomar al juego cuando se minimiza. Es decir, se le resta al tiempo que se carga el tiempo guardado.

10.2.5. Pulsar "volver menú principal"

Resultado esperado

El juego debería cargar la pantalla del menú principal, guardar las puntuaciones obtenidas por el jugador hasta el momento y reproducir la melodía principal del juego.

Resultado obtenido

Cuando se pincha sobre la opción de volver al menú principal, el juego para la música que se está reproduciendo en ese momento y reproduce la melodía principal, así como muestra la pantalla de menú principal y almacena las puntuaciones del nivel en el que se encontraba el jugador.

10.2.6. Relación entre nivel-puntuaciones

Resultado esperado

Cuando el jugador completa el subnivel 20, el juego muestra las puntuaciones obtenidas en el nivel a partir de los intentos necesitados y el tiempo empleado. El cálculo que se muestra tiene que corresponderse con la ponderación ya explicada anteriormente en este documento. Además, cuando el jugador pulse sobre esta pantalla el juego le deberá reconducir a la pantalla de selección de nivel.

Resultado obtenido

Cuando se completa el nivel, la puntuación mostrada es correcta (comprobado haciendo cálculos manuales) y al pulsar sobre cualquier parte de la pantalla, se redirige a la pantalla de selección de nivel.

10.3. Pruebas de implantación

Para llevar a cabo este tipo de pruebas, se ha instalado la aplicación en distintos dispositivos Android con diferentes versiones, resolución de pantalla, etc. Las características de estos dispositivos eran las siguientes:

- Smartphone BQ Aquaris 5.0 con Android 4.2.1 y resolución 5"
- Smartphone LG G2 con Android 4.4 y resolución 5.2"
- Smartphone Motorola Moto G con Android 4.4 y resolución 4.5"
- Smartphone BQ Aquaris 4.5 con Android 4.1.1 y resolución 4.5"
- Smartphone Samsung Galaxy SCL con Android 2.3.6 y resolución 4"
- Tablet Airis OnePad 700 con Android 2.2 y resolución 7"
- Tablet Nexus 7 con Android 4.4 y resolución 7"

En todos ellos se ha comprobado que cada imagen e icono se adopta perfectamente a las pantallas de los siete dispositivos a pesar de tener una resolución distinta.

Estas pruebas fueron llevadas a cabo por un grupo de ingenieros informáticos ajenos al desarrollo del juego. Más tarde se profundizará en esta cuestión en el apartado de Implantación. Este grupo de testers reportaron los siguientes fallos en el juego:

1. Problemas con el sonido del juego
   El sonido del juego experimenta un comportamiento extraño.
   - Cuando se minimiza en la primera parte del juego, antes de seleccionar el subnivel, y se retoma el juego, si se minimiza y reanuda de nuevo cuando se ha iniciado la partida en si, el sonido no se reanuda.
   - Lo mismo ocurre al revés. Es decir, cuando se minimiza el juego una vez se ha iniciado la partida y se retoma el mismo, si se vuelve

a la primera parte del juego (menú principal, ayuda, selección de idioma, nivel y subnivel) y se minimiza de nuevo, una vez que se reanuda, el sonido no lo hace.

2. Problemas con las puntuaciones
   Este problema ya había sido localizado con anterioridad por la desarrolladora del juego al realizar las pruebas unitarias y las de integración.
   - Al minimizarse el juego y reanudarlo, el tiempo del cronómetro se multiplica por dos.
   - De igual modo, al poner en pausa el juego y reanudarlo, el tiempo del cronómetro también se multiplica por dos.

10.4. Pruebas de explotación

Estas pruebas se realizan con los futuros usuarios del juego para observar su comportamiento. La finalidad de estas pruebas es saber si hay niveles demasiado difíciles, si el juego es intuitivo, si hay que realizar algún cambio y en qué, etc. Para ello, se ha realizado un checklist, el cual se puede ver en anexos, para comprobar la interacción del niño con el juego.

Las pruebas se realizaron en dos partes. Por un lado, el día 5 de junio de 2014, se realizaron con nueve niños diferentes, de los cuales cinco eran niños y cuatro niñas. Sus edades comprendían entre los seis y los nueve años, es decir, cursaban desde primero hasta tercero de primaria.

Estas pruebas se realizaron de manera individual. Mientras los niños estaban entrenando al tenis, se escogieron nueve al azar cuya edad estuviera comprendida entre los seis y los nueve años como se ha mencionado en el apartado anterior y que además estuvieran dispuestos a probar el juego.

En primer lugar, además de preguntarles el nombre, la edad y su lengua materna, me presentaba y les contaba el motivo de la realización del juego. Una vez sentados, les daba la tablet para que fueran ellos los que interactuaran con el juego y les explicaba su objetivo. Cuando se iniciaba el juego la primera pregunta era si eran capaces de saber dónde se activaba o desactivaba el sonido. A continuación, les dejaba que tomaran las riendas para observar si accedían o no a la ayuda. Evidentemente, algunos de ellos sí lo hicieron mientras que otros, no. A los que accedían les preguntaba si les parecía de utilidad esta ayuda y con cuánta la puntuarían del uno al diez siendo el diez

la máxima puntuación. Si no accedían a la ayuda, era yo quien les explicaba lo que debían de hacer (seleccionar idioma, nivel y subnivel).

Mientras seleccionaban yo les observaba y anotaba sus elecciones. Cuando comenzaban a jugar apuntaba cuánto tardaban en realizar cada subnivel para más tarde hacer la media de lo que tardaban en cada subnivel y nivel, hecho de gran importancia a la hora de calcular las puntuaciones. Más o menos, todos jugaron a diez subniveles de cada nivel escogido. Después de esto les mandaba seleccionar un segundo nivel para poder seguir observando su fluidez, comprensión y facilidad de aprendizaje con el juego.

Una vez finalizada la prueba con cada niño, les preguntaba su opinión sobre las imágenes que habían visto, las músicas que habían escuchado y su grado de satisfacción con el juego en general y les pedía que valoraran la dificultad del juego, de nuevo, del uno al diez, siendo diez muy difícil.

Cuando se acababa con un niño, se esperaba un tiempo para poder apuntar todas las observaciones, las preguntas realizadas por alguno de los niños así como cualquier comentario suyo o hecho significante. Después de esto, se pasaba al siguiente.

Por otro lado, el día 6 de junio se realizó la prueba a otra niña, cuya lengua materna es el euskera. Esta prueba se realizó para realizar la comprobación de este idioma debido a que ninguno de los otros nueve niños lo había escogido. El procedimiento fue el mismo que con los otros nuevo niños.

En las gráficas que se muestran a continuación, se pueden observar los datos obtenidos.

En primer lugar, en la figura 10.1, se muestran los datos obtenidos en general. Exceptuando el tiempo empleado en resolver el nivel, medido en minutos, todos los demás tienen una puntuación máxima de diez y mínima de uno. Todos ellos, comprenden de manera rápida el funcionamiento y el objetivo del juego, no presentando una dificultad elevada para ellos. Del mismo modo, muestran un fácil aprendizaje del juego. Se desenvuelven muy bien tanto con el juego como con el dispositivo, manejándolo con fluidez. El grado de satisfacción del juego, de las imágenes y la música es elevado. En general, no encuentran muy útil la ayuda ya que la mitad de ellos ni siquiera pincho sobre ella antes de empezar a jugar. En cuanto al tiempo medio empleado en resolver un nivel, es de ocho minutos y medio.

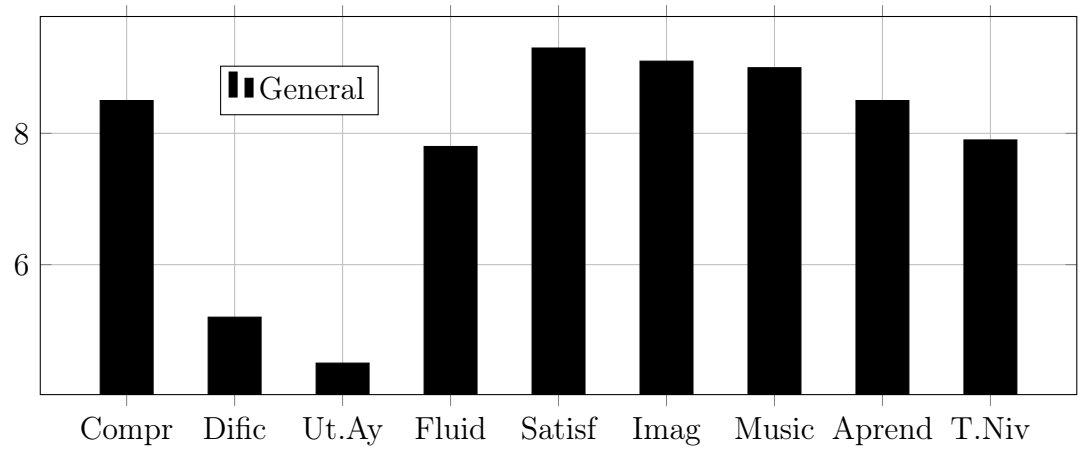


Figura 10.1: Datos obtenidos de las pruebas con los niños

En la figura 10.2, se estudia los datos obtenidos diferenciando el sexo de los niños, cinco niñas y cinco niños. Se puede observar que los niños tienen una mayor facilidad de comprensión y aprendizaje, pero les resulta más difícil a la hora de jugar y lo hacen con menor fluidez lo que les lleva a necesitar más tiempo a la hora de resolver un nivel. En general, el juego, sus imágenes y música han tenido mayor aceptación entre el sexo masculino. Hay gran diferencia entre la utilidad que encuentran los niños a la ayuda frente a la que encuentran las niñas, siendo la de estas mayor.

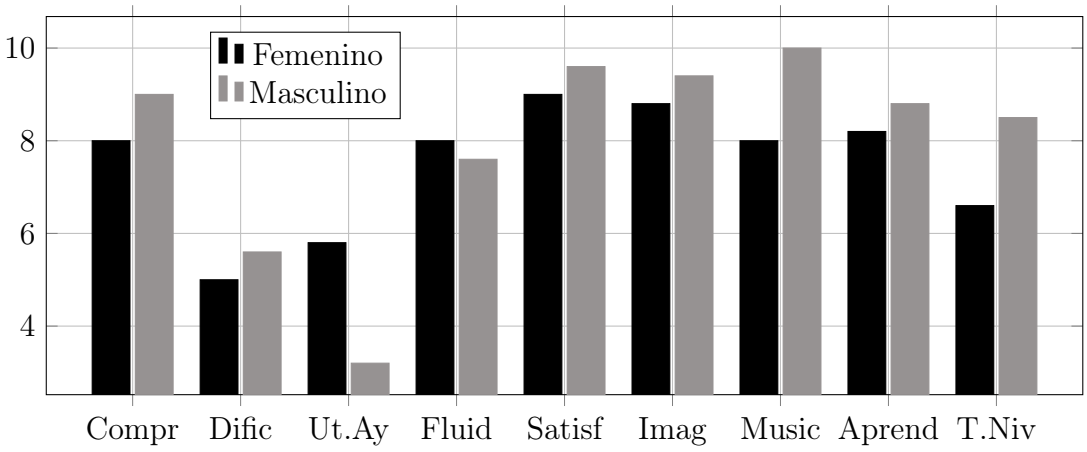


Figura 10.2: Datos obtenidos diferenciando el sexo de los niños

Y por último, en la figura 10.3, se estudia los datos obtenidos en relación a la edad de los niños, comprendida entre los 6 y los 9 años. Con 6 años hay dos niños, con 7 hay cuatro niños, con 8 años hay dos niños, y por

último, con 9 años hay otros dos niños. No hay grandes diferencias entre los niños de 6 y 7 años, al igual que entre los de 8 y 9 pero si entre ellos. Se puede observar que los niños de 7 y 8 años tienen una mayor facilidad de comprensión y aprendizaje que los de 6 y 7 como es lógico. Los de 7 presentan mayor comprensión y facilidad de aprendizaje que los de 8, teniendo relación esto con lo analizado en el caso anterior. Tres de los cuatro niños con 7 años, son chicos, costándoles a estos menos. Como es evidente, a los niños de 6 y 7 años se les presenta mayor dificultad que a los de 8 y 9; y por lo tanto también juegan con menos fluidez. En cuanto al grado de satisfacción del juego, de sus imágenes y música es similar en todas las edades, muy alto. Lo único que se podría destacar, es que la menor aceptación de la música se ha dado en los niños de 8 años. La utilidad de la ayuda y el tiempo empleado va en progresión de la edad. Es decir, los más pequeños son los que le encuentran la menos utilidad y los que más tiempo emplean en resolver un nivel, mientras que los más mayores, son los que más utilidad le encuentran a la ayuda y los que menos tiempo emplean.

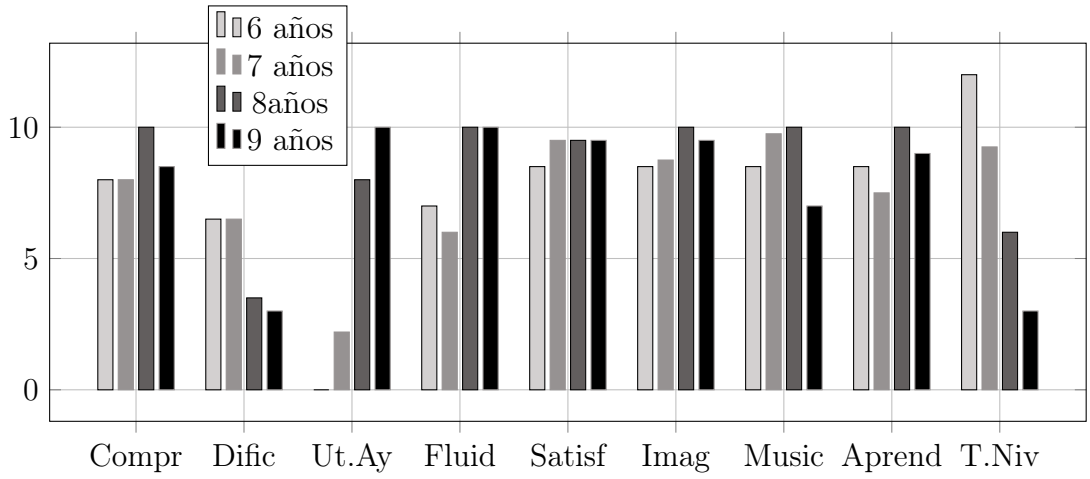


Figura 10.3: Datos obtenidos diferenciando la edad de los niños

A partir de todo esto, se ha podido concluir lo siguiente:

En un principio se dudaba a partir de que edad podría ir destinado el juego. Gracias a estas pruebas, se considera que 6 años es la edad adecuada, al menos en castellano para niños cuyo idioma materno sea este y lo mismo para euskera, para aquellos niños cuya lengua materna sea castellano y euskera.

Exceptuando a uno de los niños, el resto selecciona jugar en castellano. Se cree que esto puede ser porque los niños con los que se ha hecho las pruebas son aún pequeños y la mayoría de ellos está comenzando a escribir, por lo

que se concluye que el nivel de inglés será para niños mayores de 10 años.

En cuanto a las pistas, son revisadas todas teniendo en cuenta la carencia de conocimiento demostrada por alguno de los niños. Por ejemplo, en la palabra "bombero", se añade la M como pista para hacerles saber que antes de B siempre va M. En la palabra "ojos", se añade la J debido a que se ha detectado problemas con la G y la J.

Del mismo modo, algunas de las palabras son sustituidas. "Pantalón", pasa a ser "pantalones" ya que los dos niños que respondieron a ese subnivel, lo escribieron de esa manera.

Se observa que unos niveles gustan más que otros. Los dos más seleccionados son el 1, los animales, y el 3, los alimentos. Por otro lado, el menos elegido es el 8, acciones de la vida cotidiana, habiendo sido escogido por sólo un niño. Igualmente, y cómo es lógico hay niveles y subniveles que les resultan más difíciles. En concreto, el nivel 8, a la niña que lo seleccionó le produjo dificultad a la hora de resolverlo. Se considera normal, por lo que en lo relacionado con este tema no se llevará a cabo ningún cambio.

Del mismo modo, no se hará ningún cambio en los iconos ni en la distribución de las opciones, debido a que, en general, les queda bastante claro todo ello y es intuitivo.

Todos ellos, entienden de manera rápida el funcionamiento del juego y su objetivo. Manejan hábilmente y con fluidez tanto los dispositivos como el propio juego. La gran mayoría de las preguntas de los niños están relacionadas con el teclado y dónde escribir. Esto es cuestión de práctica por lo que no se realiza ningún cambio.

En general, tanto las imágenes como la música y el juego, les produce un nivel de satisfacción muy elevado. Únicamente una de las imágenes, "leer", ha tenido que ser reemplazada, debido a que daba lugar a confusión y no les quedaba muy claro de que se trataba. El resto, imágenes y músicas, se dejan como estaban.

Por último, se redefine el valor de ponderación de las puntuaciones. En un principio, se estableció que se dividiría 450.000 entre el tiempo empleado, ya que se estimó que sería necesario un minuto y medio (90 segundos) para completar el nivel y por ello se podría obtener una puntuación máxima de 5.000. Visto que se necesita más tiempo, una media de ocho minutos y medio (510 segundos), el valor a ponderar se sustituye por 2.550.000.

10.5. Pruebas de documentación

Estas pruebas no están relacionadas con el producto en sí, sino que tienen que ver con la memoria del proyecto.

Para comprobar que no haya errores ortográficos y no haya ningún apartado incompleto se ha ido leyendo cada uno de ellos cuando se han ido finalizando. Del mismo modo, una vez finalizada se ha releído por completo el documentos corrigiendo frases mal redactadas y algún que otro fallo de ortografía. Además había un hito marcado para el 16 de junio para entregar la memoria para ser revisada por última vez por el director del proyecto. Finalmente, se entregó el 7 de junio. Con el feedback del director se llevó a cabo pequeñas remodelaciones de la estructura de la memoria, la corrección de algún error ortográfico no detectado previamente, la adición de los prototipos diseñados para la captura de requisitos, se eliminaron partes que se creyeron que no eran necesarias, así como se ampliaron otras que se consideraron demasiado escuetas.

Como se puede observar, la batería de pruebas de cada uno de los apartados ha servido para detectar diversos fallos de baja consideración, que fueron resueltos todos ellos. A parte de estos pequeños problemas y de los típicos errores por descuido a la hora de implementar el código, no han ocurrido incidentes reseñables durante el desarrollo de estas pruebas. En general, todo el módulo ha necesitado ajustes tras realizarse las pruebas.

Capítulo 11

Implantación

En este apartado se habla sobre cómo se ha llevado a cabo la implantación del juego en diferentes dispositivos a través de la tienda online Google Play y los resultados que se han obtenido. Por ello, también se explica brevemente qué es esta tienda online.

11.1. Google Play

En octubre de 2008, Google publicó Android Market[1], el cual paso a llamarse Google Play en marzo del 2012 al fusionarse con Google Music.

Se trata de una plataforma de distribución digital de aplicaciones móviles para los dispositivos con sistema operativo Android, así como una tienda en línea desarrollada y operada por Google. Esta plataforma permite a los usuarios navegar y descargar aplicaciones, música, libros, revistas y películas.

Para acceder a ella se puede hacer desde la aplicación instalada en el propio dispositivo móvil o desde la pagina web de Google Play: `https://play.google.com/store`.

Las aplicaciones de Google Play pueden ser gratuitas o de pago.

[1] `http://es.wikipedia.org/wiki/Google_Play`

La gran novedad que aporta Google Play, y a la que a nosotros nos interesa, hace referencia a los desarrolladores. Estos, tienen la posibilidad de subir sus aplicaciones en un servicio abierto. Para ello hay que seguir tres pasos:

1. Registrarse como desarrollador en "Google Play Developer". Para lo cual hay que pagar una cuota de $25
2. Subir la aplicación
3. Describir su contenido y publicarlo

Del mismo modo, para registrarse como desarrollador y poder vender las aplicaciones, el desarrollador debe registrarse también en "Google Checkout", siendo esta vez gratuito.

Como se puede ver, Google Play no tiene ningún proceso de aprobación de las aplicaciones, sino que se basa en un sistema de permisos, al contrario que otras plataformas de distribución como por ejemplo AppStore de Apple en la que las aplicaciones si son revisadas antes de validarlas. En Goolge Play, por tanto, la aplicación muestra al usuario todos los permisos necesarios para proceder a su instalación. Estos controlaran el acceso a los servicios del teléfono, a la conexión de redes, etc.

11.2. Implantación de Guess the Pic

Para implantar Guess The Pic, lo primero que ha habido que hacer, como se ha explicado en el apartado anterior, ha sido registrarse como desarrollador. Una vez hecho esto, ya se puede subir el juego desde el nuevo usuario, el cual se puede modificar a tu gusto añadiendo el nombre del desarrollador, su dirección de correo electrónico, el sitio web, el número de teléfono, etc.

11.2.1. Versión Beta

Para crear una aplicación de calidad, es importante obtener la opinión de los usuarios lo antes posible y de forma frecuente. Para ello, el día 30 de mayo, se subió una versión beta (1.0) para probarla con un grupo de usuarios. Por este motivo, hubo que dar de alta a un grupo reducido de testers, formado por ingenieros informáticos, los cuales eran los únicos que tenían acceso al juego.

Gracias a este lanzamiento se detectaron una serie de fallos en el sonido y las puntuaciones, todo ello explicado en el apartado de Pruebas de Implantación.

11.2.2. Última versión

Una vez finalizado el juego, el día 20 de junio, se subió la versión definitiva (2.0). Aunque se prevé que en unas semanas haya otra actualización del juego (2.1) y se suba al Google Play. En dicha actualización se harán mejoras en el icono del juego, se arreglarán una serie de fallos relacionados con el sonido, las puntuaciones, las pistas y las soluciones en euskera.

Google Play posibilita el seguimiento del rendimiento del juego mediante estadísticas. Por ejemplo, se pueden ver los gráficos de rendimiento de la instalación de la aplicación a lo largo del tiempo, las instalaciones actuales por dispositivo, al igual que las desinstalaciones, las valoraciones, etc. y además, consultar los datos según la versión de Android, el dispositivo, el país del usuario, el idioma del usuario, la versión, etc. Con todo esto, se ha realizado un estudio sobre la evolución de las instalaciones y desinstalaciones de Guess the Pic en Google Play desde el 20 de junio hasta el 28 de junio y estos son los resultados que se han obtenido.

En la figura 11.1, se puede observar la gráfica con las instalaciones actuales por dispositivo desde que la aplicación está activa.

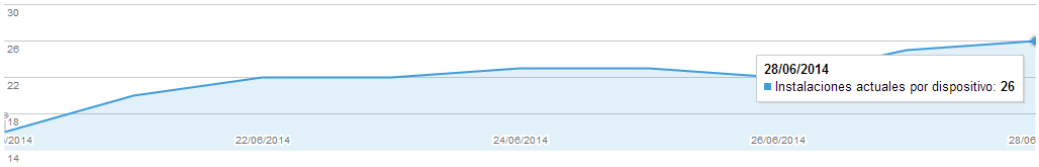


Figura 11.1: Gráfica del progreso de las instalaciones actuales

En esta gráfica, se puede ver, que a día 28 de junio de 2014 hay un total de 26 instalaciones en diferentes dispositivos. A excepción del día 27, se puede observar que el número de instalaciones es ascendente y el día en el que hubo un mayor número de instalaciones que aún permanecen, fue el día en el que se subió el juego al Google Play, en concreto, 16 instalaciones.

A su vez, este número de instalaciones se pueden desglosar por dispositivo, versión de Android, país y operador entre otros. En la figura 11.2, se pueden observar estos datos.

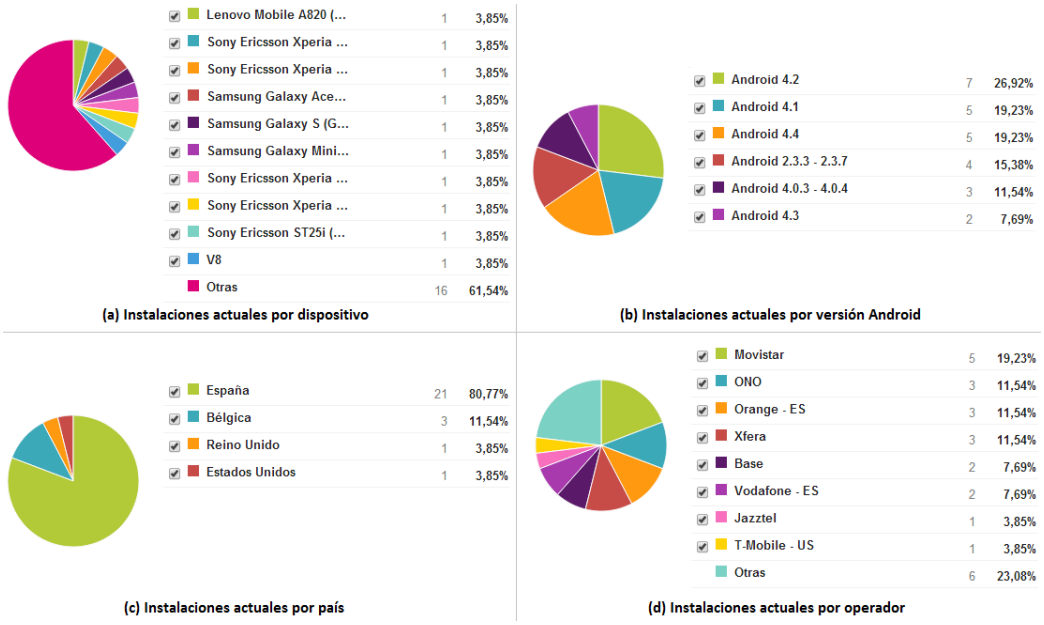


Figura 11.2: Gráficas de las instalaciones actuales por dispositivo

En la gráfica (a) Instalaciones actuales por dispositivo, se puede ver que hay un total de 26 dispositivos distintos en los que el juego está instalado. Por otro lado, en la gráfica (b) Instalaciones actuales por versión Android, se observa que las versiones están comprendidas entre la 2.3.3 y la 4.4, ambas incluidas. La que más instalaciones tiene es la versión 4.2, con siete, mientras que la es la que menos número de instalaciones tiene es la 4.3, con sólo dos. En cuanto a las instalaciones por países, se puede ver en la gráfica (c) Instalaciones actuales por país, que en estos momentos está instalada en dispositivos de cuatro países diferentes, España, Bélgica, Estados Unidos y Reino Unido. Evidentemente, en España es donde hay un mayor número de dispositivos con el juego instalado. Y por último, en la gráfica (d) Instalaciones actuales por operadores, hay 14 compañías diferentes, destacando Movistar, con cinco instalaciones.

Frente a las instalaciones actuales, se pueden obtener las instalaciones totales. En la figura 11.3 se muestra la gráfica con dichas instalaciones.

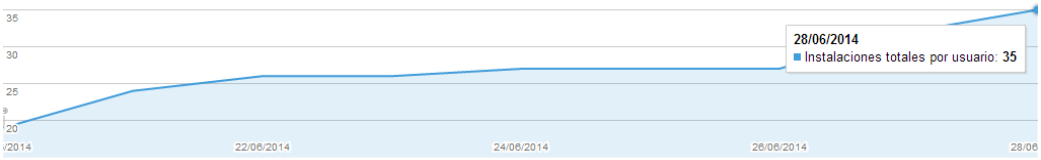


Figura 11.3: Gráficas de las instalaciones totales por dispositivo

En esta gráfica, se puede ver, que a día 28 de junio de 2014, el juego ha sido instalado en un total de 35 dispositivos diferentes frente a los 26 en las que está actualmente instalado. El primer día, el 20 de junio, hubo 19 instalaciones de las cuales, perduran 16. Al igual que la gráfica de instalaciones actuales, esta gráfica también muestra una progresión linealmente ascendente.

Del mismo modo que anteriormente, en la figura 11.4, se pueden observar las instalaciones totales en detalle según el dispositivo, la versión de Android, país y operador.

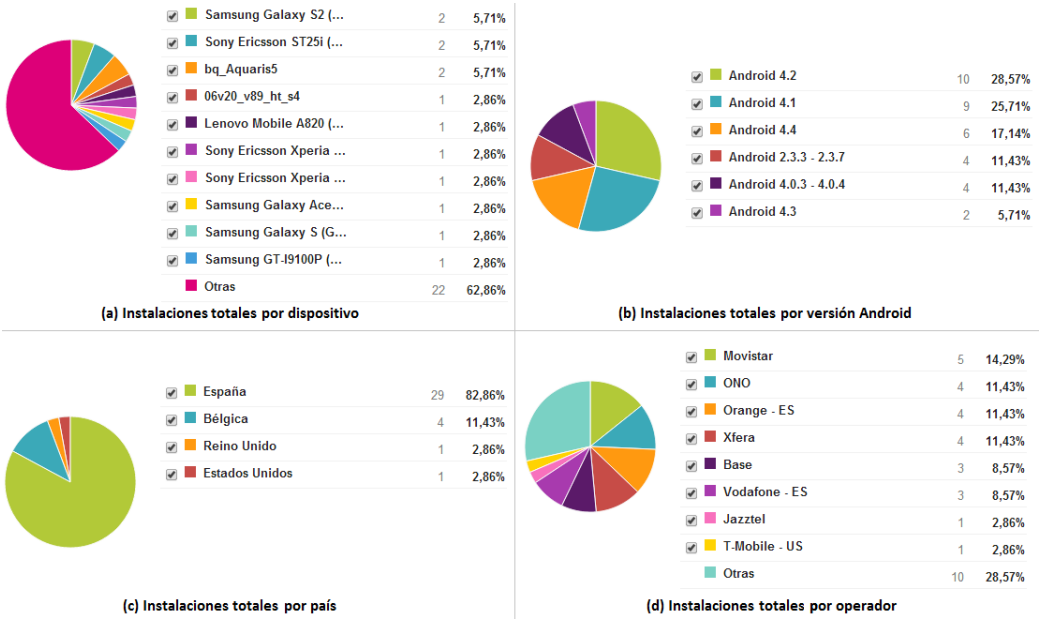


Figura 11.4: Gráficas de las instalaciones totales por dispositivo

En la gráfica (a) Instalaciones totales por dispositivo, se puede ver que el juego se ha instalado en 32 dispositivos distintos. Comparándola con la gráfica anterior de las instalaciones actuales, se pueden ver algunos móviles de los cuales ha sido desinstalada. Por ejemplo, el juego ha sido removido de dos Samsung Galaxy S2 y de un Sony Ericsson ST25i ya que antes estaba

instalado en dos, etc. En cuanto a las versiones en las que se ha instalado el juego, en la gráfica (b) Instalaciones totales por versión Android, se puede ver que son las mismas en las que todavía aparece el juego. Las versiones de Android en las que más desinstalaciones se han producido es en la 4.2 pasando de diez a siete y la 4.1 pasando de nueve a cinco. Por otro lado, en lo relacionado a las instalaciones por países, no ha sido instalado el juego en más de los que se encuentra actualmente. En la gráfica (c) Instalaciones totales por país se puede observar lo comentado. Y por último, en la gráfica (d) Instalaciones totales por operadores, pasa exáctemente lo mismo que en el caso anterior, no hay más operadores que en la actualidad.

También se puede observar la gráfica con las desinstalaciones actuales por dispositivo desde que la aplicación está activa. En la figura 11.5, aparece dicha gráfica.

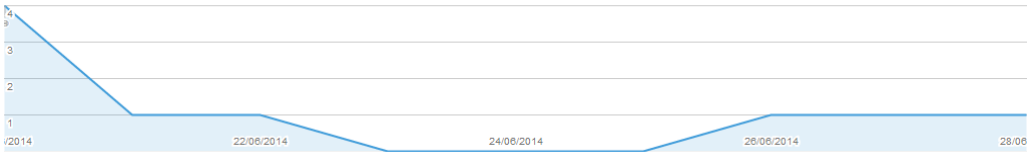


Figura 11.5: Gráficas de las desinstalaciones actuales por dispositivo

En esta gráfica, se observa el número total de desinstalaciones que se han producido, nueve. En el día 20 de junio, hubo un total de cuatro desinstalaciones siendo este día en el que mayor número se registraron. Durante los días 23, 24 y 25 no se produjo ninguna; y el resto de los días únicamente una.

Capítulo 12

Gestión

En este apartado se recoge y reflexiona sobre los sucesos acontecidos y los cambios que se han dado a lo largo de la vida del proyecto con respecto a lo inicialmente planificado en el documento de objetivos del proyecto.

12.1. Objetivos

Se marcó como objetivo principal el desarrollo de una aplicación educativa para Android que permitiera a sus usuarios, en este caso, niños, desarrollar sus habilidades de escritura. Dicho objetivo se ha cumplido con el desarrollo de Guess the Pic. Más adelante, en el apartado relativo al *alcance* se enumeran y detallan las características concretas de dicha aplicación, y su comparación con las previsiones iniciales.

Por otro lado, se marcaron como objetivos indirectos la familiarización de la desarrolladora con las aplicaciones móviles y por lo tanto con Android, la adquisición de experiencia en los procedimientos para el desarrollo de software, incluyendo las herramientas de control de versiones, y en la edición de documentos con LaTeX.

La competencia (o al menos familiarización) resultante en el campo de las aplicaciones móviles se considera que ha sido satisfecha ya que se ha logrado terminar el juego con un resultado óptimo.

En el terreno de los procedimientos para el desarrollo de software, se han llevado a cabo diversas acciones que se consideran buenas prácticas, tales como el haber realizado una planificación para posteriormente seguirla, la

elaboración de pruebas unitarias o el uso de sistemas modernos de control de versiones.

Por último, la pericia adquirida en la edición de documentos con LaTeX puede apreciarse en esta misma memoria, teniendo en cuenta que al inicio del proyecto no se tenía ningún conocimiento de dicho sistema.

12.2. Alcance

En cuanto al alcance, se fijó como requisitos mínimos la elaboración de un juego educativo, aspecto que se ha cumplido; que el juego constara de tres idiomas diferentes, ocho niveles y 160 subniveles distintos, así como de una pista por subnivel, que del mismo modo se ha cumplido. Se fijó también el requisito de que fuera compatible con los dispositivos Android con sistema operativo cuya versión estuviera comprendida entre la 2.2 y la 4.4, así como el almacenamiento offline por nivel de las puntuaciones y otras funcionalidades del juego, las cuales han sido todas llevadas a cabo.

Por otro lado, originalmente se planteaba como línea de ampliación la adicción de un mayor número de niveles y subniveles, incluir un cuarto idioma, sonidos a adivinar y el almacenamiento de las puntuaciones de manera online. Al final esto no se llevó a cabo por falta de tiempo y queda como ampliaciones futuras. En la falta de tiempo, parte se debe a que hubo muchos problemas con las puntuaciones y los sonidos. Ya se ha comentado en las pruebas, los fallos que se encontraron en estas dos características y costó más de lo previsto buscar el error y solucionar esos comportamientos extraños.

12.3. Planificación temporal

En esta sección se evalúa el cumplimiento de los hitos marcados al inicio del proyecto y se analiza el coste temporal real que han tenido las diversas actividades realizadas.

12.3.1. Hitos

En la siguiente tabla 12.1, se puede ver la comparación entre los hitos planificados y los reales. Para comenzar, cabe destacar que, aunque inicialmente

se había planificado comenzar con la memoria una vez se finalizara la implementación del juego, finalmente se decidió realizarlo de manera simultánea para ir adelantado la memoria.

Hito	F. estimada	F. real
Finalización de la planificación	21 de febrero	20 de febrero
Finalización del prototipo	16 de abril	11 de abril
Decisión de ampliación del alcance inicial	21 de abril	16 de abril
Finalización de la implementación	4 de junio	19 de junio
Finalización de la memoria	13 de junio	7 de junio
Entrega del proyecto	1 de julio	*1 de julio*
Cierre del proyecto	15 de julio	*15 de julio*

Tabla 12.1: Hitos planificados frente a los reales

A continuación se expone el grado de cumplimiento de los hitos establecidos:

- El primero, que estaba relacionado con la **finalización de la planificación** se cumplió a raja tabla ya que estaba fechado para el 21 de febrero y para el 20 de este mes se envió dicha planificación al director.
- El segundo, que preveía la **presentación de un prototipo funcional** el 16 de abril sobre el que recibir críticas y proponer cambios, también se cumplió a rajatabla. El día **11 de abril**, tuvo lugar una reunión en la que se presentó dicho prototipo.
- El hito relacionado con la **decisión de ampliación del alcance** se tomó el **20 de abril**, estando fijado para el día siguiente. Se decidió que no se ampliaría el alcance debido a que hasta el 19 de mayo no se iba a poder continuar con el proyecto y se consideró que no había tiempo suficiente.
- El siguiente hito, era la **finalización de la implementación** y el **inicio del proceso de documentación**, fechado para el 4 de junio. Como se ha comentado anteriormente, finalmente se decidió que la documentación se empezaría antes, en concreto el **19 de mayo**, una vez fuera retomado el proyecto. En cuanto a la finalización de la última versión del juego, no tuvo lugar hasta el **19 de junio** retrasándose 15 días según lo previsto.

- En lo relativo al hito relativo a la **finalización de la memoria**, el **7 de junio** se entregó una versión para ser corregida no estando completada al cien por cien, dejando por lo tanto este hito hasta la última posible fecha de entrega.
- Por último, en cuanto al hito de **finalización de todos los entregables** y **cierre del proyecto**, no puede concretarse el día en el que tendrán lugar pero si se prevé que estén dentro del margen de los días en los que se establecieron, el **1 de julio** y el **15 de julio** respectivamente.

12.3.2. Actividades

En cuanto a las desviaciones temporales en las actividades, en el tabla 12.2 se recogen los costes temporales originalmente previstos frente a los reales. Se planificó que se necesitarían 360 horas de trabajo y finalmente han sido necesarias 414 horas.

Tarea	T. estimado	T. real
Planificación	**15 h.**	**10 h.**
Planificación inicial	10 h.	10 h.
Replanificaciones	5 h.	0 h.
Formación	**65 h.**	**56 h.**
Análisis de mercado	10 h.	15 h.
Formación en SDK Android	45 h.	30 h.
Formación en control de versiones	5 h.	2 h.
Formación en LaTeX	5 h.	9 h.
Desarrollo	**190 h.**	**230 h.**
Captura de requisitos	10 h.	7 h.
Análisis	15 h.	13 h.
Diseño	15 h.	18 h.
Implementación	130 h.	171 h.
Pruebas	20 h.	21 h.
Documentación	**70 h.**	**103 h.**
Memoria	40 h.	55 h.
Manual de usuario	10 h.	6 h.
Revisión y corrección de errores	10 h.	32 h.
Defensa	10 h.	*10 h.*
Gestión	**20 h.**	**15 h.**
Seguimiento y control	15 h.	10 h.
Reuniones	5 h.	5 h.
Total	360 h.	414 h.

Tabla 12.2: Coste temporal estimado frente al coste real

Cabe hacer algunos comentarios al respecto sobre las desviaciones más considerables:

- A pesar de los cambios que se fueron dando a lo largo del proyecto, no se vio la necesidad de hacer una nueva planificación, por lo que no se consumió ninguna de las horas previstas para esa tarea.
- La formación en la SDK de Android consumió bastantes menos horas de las previstas (30 frente a 45) principalmente por la documentación existente y porque finalmente no resultó ser tan diferente de programar cualquier otra aplicación en Java como se pensaba.
- La formación en control de versiones ya se sabía de antemano que sería una tarea que consumiría poco tiempo. Sin embargo, al haber realizado todas las estimaciones en módulos de 5 horas, se decidió asignarle a la tarea un módulo completo, en vez de las 2-3 horas que realmente se creía que se iba a tardar. Y así ha sido, finalmente el aprendizaje de su uso resultó trivial, consumiendo poco más de 2 horas.
- En la formación en LaTeXse necesitaron 9 horas frente a las 5 previstas, ya que hubo que investigar no sólo en como crear documentos, sino en gráficas, tablas, etc, y llevó más tiempo del previsto.
- En cuanto a la implementación, se estimó que se necesitarían 130 horas. Finalmente se han necesitado 171, habiendo una desviación temporal considerable. Esto se debe a la inexistente experiencia de la desarrolladora en este campo, por lo que era imposible saber cuánto tiempo llevaría cada tarea de la implementación.
- En cuanto a las pruebas, se previeron 20 horas en total. La cifra de este apartado depende mucho de la forma de cómputo, ya que sólo se han tenido en cuenta las sesiones finales de pruebas y no los pequeños intervalos que se realizaban según se iba desarrollando el juego. El total de horas se puede dividir en dos partes. Por un lado, el tiempo invertido en las pruebas con los niños más el análisis de estas, que fueron tres+cuatro horas y por otro, el de las pruebas unitarias y de integración que fueron 14, haciendo un total de 21 horas.
- En la realización de la memoria finalmente se han necesitado más horas de las previstas. Esto es, porque en un principio no se tuvo la idea de hacer un documento tan completo, pero al añadir varia información sobre Android, la implantación en Google Play, y pequeñas desviaciones en lo si previsto, han hecho que se necesiten 55 horas frene a las 40 estimadas.

- En la redacción de manuales se incluía la redacción de los manuales de instalación de la SDK de Android y el propio manual de usuario, aunque finalmente el primero se ha excluido debido a que se ha considerado que había suficiente información en la web. Es por este motivo que se han necesitado 4 horas menos de las que se previó.
- En cuanto a la revisión y corrección de errores, se han necesitado más de las 10 horas previstas, en concreto 32 horas. Esto se debe a que únicamente en leer la memoria completa y corregir fallos, se tardó casi 6 horas y se ha llevado a cabo dos veces. En esta tarea, también se incluyen los cambios solicitados por el director, los cuales han llevado más tiempo del que se previó.
- Las tareas asociadas a la defensa del proyecto (preparación de la presentación, grabación del vídeo, ensayos y la propia defensa final) no se han llevado a cabo a la hora de entregar esta memoria, por lo que las 10 horas que se indican continúan siendo una estimación. Por eso están marcadas en cursiva en la columna relativa al tiempo real invertido.
- En lo relativo al seguimiento y control del proyecto, se estimó que fueran necesarias 15 horas, habiendo utilizado finalmente 10. Este seguimiento ha consistido en llevar un control de la horas invertidas en cada tarea anotándolas al finalizar cada jornada en un documento excel diseñado para ello, controlar el cumplimiento del alcance y los hitos establecidos, en realizar una comparación entre el esfuerzo planificado y lo real, solventar los riesgos que se han materializado
- Y por último, en cuanto a la reuniones, se establecieron 5 horas, pensando que se llevarían a cabo cinco reuniones de una hora de duración cada una de ellas. Esta previsión fue totalmente correcta. Se produjeron desvíos insustanciales en todas, por lo que no se han tenido en cuenta.

12.3.3. Diagrama de Gantt

Para poder analizar la desviación de los días, se realiza de nuevo un diagrama de Gantt, el cual se muestra en la figura 12.1.
En él, se pueden observar las desviaciones que se ha sufrido a lo largo del proyecto. Siempre se ha intentado cumplir con la planificación inicial, pero a pesar de esto, diversos aspectos a la hora de desarrollar el proyecto han exigido que se alargue durante más tiempo del previsto.
Las desviaciones y sus motivos ya están explicados, pero se quiere destacar

aquellos que se han prolongado considerablemente frente a las fechas planificadas. Muchas de las tareas, se han comenzado antes o acabado después pero al fin y al cabo han tenido la misma duración o un desviación de una semana.
La que llama considerablemente la atención, es la de implementación. Se estimó que se comenzaría con esta tarea en la semana 7 y se finalizaría en la 14. La realidad, en cambio, ha sido que se comenzó con ella en la semana 5 y se alargó hasta la 19, abarcando 7 semanas más de lo establecido y con ello, un mayor número de horas, hecho ya comentado anteriormente.

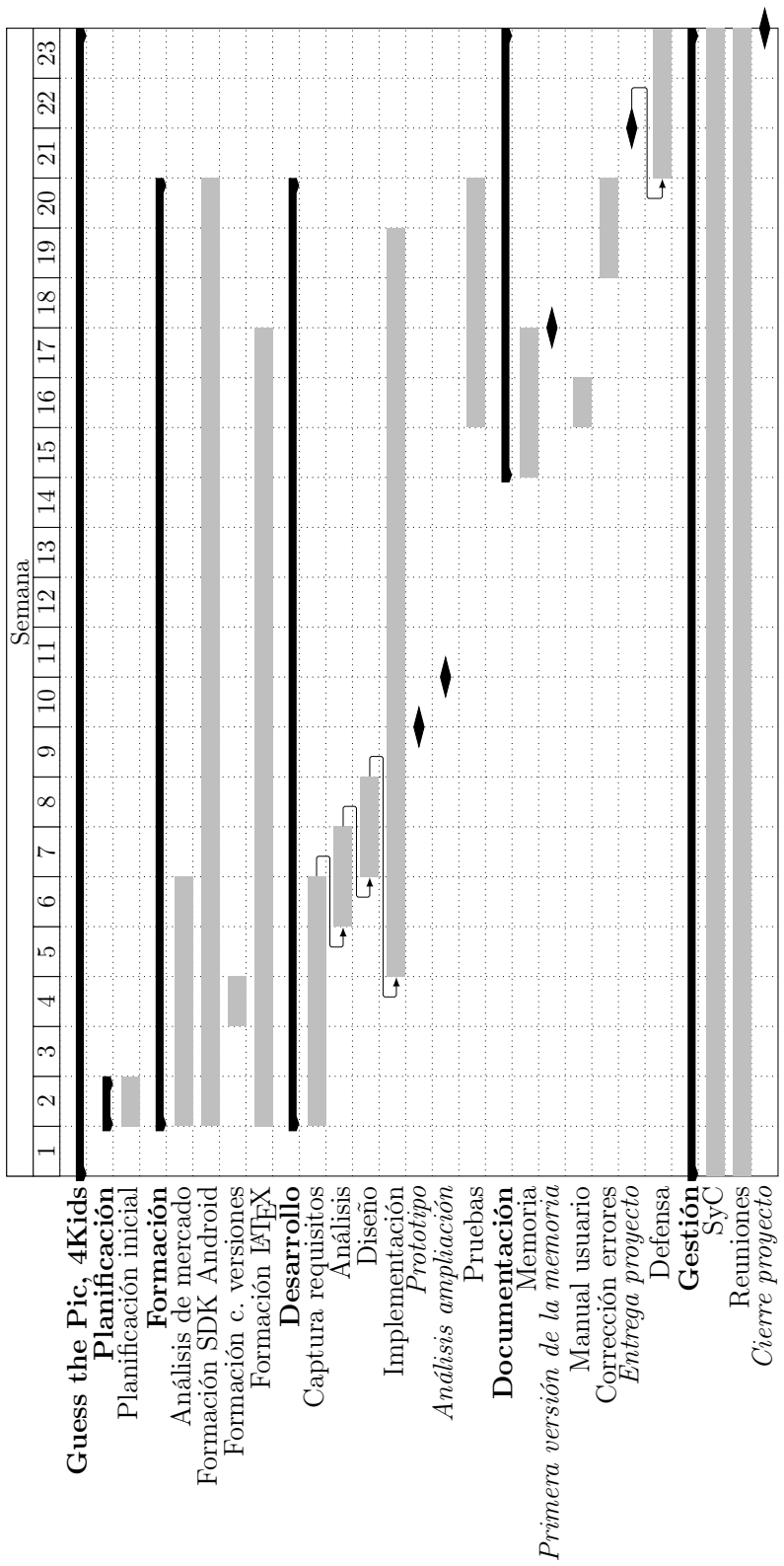


Figura 12.1: Diagrama de Gantt real

12.4. Comunicaciones

12.4.1. Canales de comunicación

En cuanto a los canales de comunicación, se han utilizado los dos que se planificaron:

1. **Conversación en persona.** Se han realizado las reuniones en persona estrictamente necesarias, cinco, para poner al coordinador al tanto del proyecto.
2. **Correo electrónico** Cuando no se ha podido llevar a cabo reuniones en persona, el correo electrónico ha sido el canal principal de comunicación, siendo la mayoría de las veces utilizado para establecer reuniones de seguimiento.

12.4.2. Métodos de comunicación

Se celebraron las siguientes reuniones de seguimiento entre los dos interesados en el proyecto, la desarrolladora y el director del proyecto. Todos las fechas que se indican corresponden al año 2014:

- **14 de febrero** Inicio del proyecto.
- **14 de marzo** Definición del alcance del juego, se establece el índice y los temas que se trataran en la memoria.
- **11 de abril** Muestra de la primera versión funcional del juego, comentarios y propuestas de mejora.
- **9 de mayo** Consulta sobre varias cuestiones de la memoria.
- **23 de junio** Revisión de la memoria y propuestas de mejora.

12.5. Riesgos

A continuación se enumeran los riesgos que se especificaron en el plan inicial, comentando hasta qué punto sucedieron o se consiguieron evitar:

- **Retrasos:** Se dieron importantes retrasos, sobre todo en mitad del proyecto, cuando la dedicación a él tuvo que dejar de ser completa

durante un mes. Sin embargo, se compensó con una mayor dedicación al comienzo y al final, por lo que se logró reconducir la situación y cumplir en mayor parte con los hitos originalmente fijados, tal y como se ha expuesto anteriormente.

- **Planificación inicial inadecuada:** La planificación resultó bastante más acertada de lo que en un principio se esperaba, por lo que este riesgo no supuso ningún problema real.
- **Requisitos definidos incorrectamente:** Los cambios en los requisitos fueron incrementales y no dramáticos, por lo que no hubo mayores problemas derivados de este riesgo.
- **Diseño inadecuado:** Al igual que en el caso anterior, el diseño fue variando a lo largo del desarrollo del proyecto, las necesidades fueron detectadas y atajadas a tiempo, por lo que no se dio ningún caso en el que se tomasen decisiones incorrectas que afectaran en gran medida al proyecto.
- **Incumplimiento del alcance deseado:** El alcance mínimo propuesto ha sido cumplido satisfactoriamente. Al comienzo, se hizo una reunión con el director y se trato este tema. Gracias a esto se estableció un alcance factible. En cuanto a las líneas de ampliación, ninguna de ellas ha podido llevarse a cabo por falta de tiempo pero no han supuesto ningún problema para el proyecto.
- **Pérdida de información:** El sistema de copias de seguridad fue estricto (copias diarias tanto en el caso del código de la aplicación como en el de la memoria), por lo que, en caso de catástrofe, se habría perdido como máximo el trabajo de un día. En cualquier caso, la situación no se dio y no hubo que hacer uso de dichas copias.
- **Baja por enfermedad:** Afortunadamente no hubo bajas por enfermedad.

12.6. Recursos externos

Ha sido necesario llevar a cabo el uso de los siguientes recursos externos durante el ciclo de vida del proyecto.

Por un lado, se han utilizado nueve melodías diferentes, todas ellas con licencias libre de uso.

- http://incompetech.com/music/royalty-free
- http://soundbible.com/1700-5-Sec-Crowd-Cheer.html

En lo relativo a los wallpapers utilizados como fondos en el juego, todos ellos han sido obtenidos de la misma página.

- http://subtlepatterns.com/

Los iconos usados para los botones de pausa, atrás, siguiente, play, sonido, etc, han sido cogidos del siguiente enlace.

- http://images.tdm.info/main.php?g2_itemId=877

Y por último en cuanto al producto, las imágenes de los niveles han sido descargadas de las siguientes páginas.

- http://pixabay.com/
- http://actividadesinfantil.com/
- http://www.gifs-animados.es/
- http://maestraerikavalecillo.blogspot.com.es/

Si se habla de los recursos externos relacionados con el proyecto, hay que hablar de la de la plantilla de LaTeXpara la memoria. Dicha plantilla se obtuvo del proyecto de Alberto Calvo y posteriormente ha sido modificada al gusto de la desarrolladora.

Capítulo 13

Conclusiones y líneas futuras

En este último capítulo se resumen los objetivos conseguidos, se hace una breve valoración personal respecto a la aplicación realizada, y se presentan las líneas principales sobre las que se puede continuar trabajando para extender y mejorar el juego.

13.1. Objetivos conseguidos

El objetivo principal del proyecto era la realización de una aplicación para el sistema operativo Android. En concreto, se trataba de desarrollar un juego educativo para niños. Por lo tanto, la obtención de resultados se corresponde con lo planteado inicialemente.

La realización del proyecto también ha alcanzado otros objetivos derivados, como por ejemplo la asquisición de conocimientos para programar en Android. Se ha podido aprender a desarrollar aplicaciones para Android en un periodo corto de tiempo gracias a la gran cantidad de información que hay disponible, tanto en la red como físicamente. Como es lógico, Google ha facilitado tutoriales y ha publicado una API para acelerar la creación de programas por parte de los usuarios. La instalación de los programas es rápida y al utilizar un lenguaje de programación tan conocido como Java han facilitado aun más las cosas. Otros dos de los objetivos derivados y que también se han conseguido, son los del aprendizaje de control de versiones, algo muy útil para cualquier proyecto y que servirá para el futuro profesional de la alumna, y el segundo, el aprendizaje del uso de LaTeXpara la edición de documentos, también muy útil debido al toque de profesionalidad que aporta.

Debido a que finalmente la alumna no ha podido dedicarle tanto tiempo como en un principio se estimó, no ha sido posible realizar ninguna de las ampliaciones que se propuso en un principio. Pero lo positivo, es que lo fundamental e importante del juego si ha podido ser llevado a cabo.

En resumen, las espectativas del proyecto se han cumplido con creces habiendo realizado una aplicación funcional. Otro factor de gran importancia para calificar de éxito este proyecto es la adquisición de conocimiento en Android por parte de la alumna ya que tanto los móviles como el sistema operativo elegido, son dos campos con gran importancia en el presente y siempre viene bien tener conocientos sobre una apuesta segura para el futuro.

13.2. Experiencia personal

En cuanto a mi valoración personal en general del proyecto, estoy muy satisfecha ya que se han cumplido todos los objetivos propuestos inicialmente y además he cumplido el reto que me propuse y que tantas ganas tenía de llevar a cabo, el de realizar un juego para Android. Es por esto, por lo que tengo motivos para estar orgullosa.

Por otro lado, el desarrollo para Android ha sido, a mi parecer, muy interesante dado el gran abanico de posibilidades que ofrece, siendo imposible recogerlas todas en un proyecto y menos en una sola aplicación. Aún a pesar de las limitaciones que ofrece el emulador frente a un dispositivo móvil real, es un buen punto de partida para familiarizarse con el entorno y probar prototipos de aplicaciones de una manera sencilla aunque no muy rápidamente.

En cuanto al juego realizado, estoy muy satisfecha con el resultado final ya que considero que la mejor forma para que los niños aprendan es jugando. Tradicionalmente se hacía con juegos físicos pero es sabido que cada vez más, lo hacen a través de dispositivos móviles. Por lo que con todo ello, opino que mi juego será de gran ayuda para ellos pudiendo aprender diferente idiomas y competencias básicas como lo son los colores, números, profesiones, etc. La utilización de imágenes animadas y la forma en la que se ha planteado el juego, creo que les ayudará a abstraerse del real próposito del juego, que no es otro que el de que aprendan, y únicamente se crentrarán en jugar e intentar conseguir completar los niveles en el menor tiempo posible y realizando el menor número de intentos. Según mi parecer, esto es bueno ya que cabe la posibilidad de que al plantear a un niño el realizar una actividad concreta para aprender, su reacción pueda ser negativa mientras que si se le plantea la

opción de jugar, seguramente su respuesta sea menos reacia que en la primera proposición. Por otro lado, también creo que puede ser de gran ayuda para los padres de los futuros usuarios del juego, ya que habrá ocasiones en las que no sepan como enseñarles a sus hijos ciertas cosas, como por ejemplo, los alimentos y con esta aplicación podrán aprender eso y muchas otras cosas.

Respecto a opiniones ajenas al desarrollo de la aplicación, como lo son de compañeros de clases, amigos o incluso las de Google Play, todas coinciden en la gran idea que he tenido para enseñar a los niños a escribir. También coinciden en la utilidad que tendrá el juego en el futuro. En general, todas las críticas han sido positivas, alabando lo bien que está estructurado y lo sencillo que resulta jugar con él, así como el toque animado que aportan las imágenes. Algunos de estos comentarios podemos verlos en Google Play:

1. David Perez Conde: "Excelente aplicación!! Muy intuitiva para los más pequeños!!"

2. Adrián Fernandez: "Es estupenda!! Tengo un primito pequeño que la usa para aprender. Es genial, muy buena y didactica!!"

3. Marcos Martínez Díez: "Muy didáctica. Genial que este en inglés."

4. Gema Mayordomo: "Es una aplicación genial! Tengo una hija de 5 años y es un juego muy apropiado para ella. Además de tener diferentes dificultades e idiomas, es un juego 100 % didáctico y divertido. A mi marido y a mí nos encanta y sobre todo a nuestra hija.

5. Ana Gorrochategui: "Realmente buena. Todo un reto para los peques. Mi enano está con los animales a tope. Gracias por una aplicación así, hacía falta :)"

6. Pedro Estébanez: "Excelente y a la vez practica aplicacion. Muy didactica y amena."

Y lo importante, la opinión de los futuros usuarios. Sus reacciones y opiniones al realizarles las pruebas también aportaron un grado de satisfacción personal ya que todos ellos, se interesaron por el nombre del juego y por cuándo estaría disponible para poder descargárselo. Y además, lo valoraron entre 8 y 10 siendo 10 la puntuación más alta.

13.3. Líneas futuras

Tras finalizar la implementación del juego, se considera que el resultado final del mismo es completo pero aun así hay una serie de mejoras que perfeccionarían y mejorarían el juego.

La principal mejora sería la de hacer una versión on-line del juego. De este modo, no haría falta que todas las imágenes relacionadas con los subniveles estuvieran en el propio APK, sino que cada vez que hubiera que cargar una imagen la obtendría de Internet reduciendo así el tamaño del juego de manera considerable.

Además, un punto fuerte de esta mejora es la de poder comparar las puntuaciones obtenidas en los distintos niveles con las de otros usuarios. Para ello, habría que crear una base de datos on-line en la que se fueran almacenando, por ejemplo, las 10 mejores puntuaciones por nivel y se actualizaran en el móvil del usuario cada vez que pinchara en "Ver puntuaciones". Además, Android incorpora de serie todas las herramientas necesarias para la creación y gestión de bases de datos SQLite, y entre ellas una completa API para llevar a cabo de manera sencilla todas las tareas necesarias. La puesta en marcha de las puntuaciones on-line conllevaría al Jugador a tener que registrarse en el juego con su cuenta de Google para así poder acceder a la base de datos.

Otra mejora que podría realizarse, y que en un principio se pensó en hacerlo por recomendación del director, es la redistribución aleatoria de los subniveles una vez que se complete el nivel, para evitar así, que los niños se aprendan las respuestas de cada subnivel. Se considera que esta mejora no llevaría mucho tiempo. En este momento, hay un array por nivel con las posiciones fijas de cada subnivel. Podrían asociarse unos números a cada uno de ellos y utilizar arrays auxiliares para formar cada nivel de manera aleatoria cuando el nivel fuera completado, es decir, cuando todos los subniveles tuvieran el valor 1.

Han quedado pendientes algunas funcionalidades que se establecieron como ampliación de alcance y que finalmente no se han llevado a cabo. Todas ellas son de menor complejidad que la propuesta anterior. Una buena opción de mejora sería la de añadir el alemán como cuarto idioma del juego ya que en la actualidad cada vez más gente lo está estudiando y es considerado el idioma de moda. También podrían añadirse mayor número de niveles y subniveles para hacer el juego más duradero. Y por último, para desarrollar el sentido del oido, se podrián añadir subniveles en los que en lugar de tener que adivinar una imagen, habría que adivinar un sonido.

Capítulo 14

Bibliografía

14.1. Libros

[8] Mario Zechner. *Desarrollo de Juegos para ANDROID*. Ediciones Anaya Multimedia, 2011.

14.2. Referencias en Internet (Implementación)

[1] *Blog de desarrollo de aplicaciones sobre Android*. URL: http://droideando.blogspot.com.es/.

[2] *Guía de Instalación de la API de Android en Eclipse*. URL: http://developer.android.com/sdk/index.html.

[3] *Sitio web oficial de Stackoverflow, foro sobre todo tipo de dudas de programación*. URL: http://stackoverflow.com/.

[4] *Sitio web oficial de Tex-Latex Stack Exchange, foro sobre todo tipo de dudas de Latex*. URL: http://tex.stackexchange.com/.

[5] *Sitio web oficial para desarrolladores en Android*. URL: http://developer.android.com/about/index.html.

[6] *Tutorial para desarrolladores de Android*. URL: http://saigeethamn.blogspot.com.es/2010/05/gallery-view-android-developer-tutorial.html.

[7] *Tutotorial de cómo descargar e instalar la SDK*. URL: http://www.comolohago.cl/como-programar-apps-para-android-1-descarga-instalacion-y-primera-app/.

Apéndice A

Checklist

Niño 1

- **Nombre:** Diana
- **Sexo:** Femenino
- **Edad:** 6
- **Curso escolar:** 1º de primaria
- **Lengua materna:** Castellano
- **Idioma del juego seleccionado:** Castellano
- **Niveles del juego seleccionados:** 4 y 3
- **Comprensión del juego:** 8
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Dificultad del juego:** 6
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Tiempo medio para resolver un subnivel:** 30"
- **Tiempo medio para resolver un nivel:** 9'
- **Utiliza el niño la ayuda previa del juego:** No
- **En caso afirmativo, utilidad de esta ayuda:** -
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Fluidez:** 8
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)

- **Preguntas hechas por el niño:** -
- **Nivel de satisfacción del niño con el juego:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de las imágenes:** 8
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de la música:** 8
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Facilidad de aprendizaje y uso del juego:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Observaciones:** No le es fácil saber para qué es cada botón

Niño 2

- **Nombre:** David
- **Sexo:** Masculino
- **Edad:** 7
- **Curso escolar:** 1º de primaria
- **Lengua materna:** Castellano
- **Idioma del juego seleccionado:** Castellano
- **Niveles del juego seleccionados:** 2 y 3
- **Comprensión del juego:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Dificultad del juego:** 6
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Tiempo medio para resolver un subnivel:** 45"
- **Tiempo medio para resolver un nivel:** 16'
- **Utiliza el niño la ayuda previa del juego:** No
- **En caso afirmativo, utilidad de esta ayuda:** -
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Fluidez:** 6
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)

- **Preguntas hechas por el niño:** ¿Cómo se borra?
- **Nivel de satisfacción del niño con el juego:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de las imágenes:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de la música:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Facilidad de aprendizaje y uso del juego:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Observaciones:** Diferencia unos botones de otros sabiendo para qué son cada uno de ellos

Niño 3

- **Nombre:** Diego
- **Sexo:** Masculino
- **Edad:** 8
- **Curso escolar:** 3º de primaria
- **Lengua materna:** Castellano y euskera
- **Idioma del juego seleccionado:** Castellano
- **Niveles del juego seleccionados:** 1 y 6
- **Comprensión del juego:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Dificultad del juego:** 4
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Tiempo medio para resolver un subnivel:** 20"
- **Tiempo medio para resolver un nivel:** 6'
- **Utiliza el niño la ayuda previa del juego:** Si
- **En caso afirmativo, utilidad de esta ayuda:** 6
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)

- **Fluidez:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Preguntas hechas por el niño:** ¿Cómo se comprueba?
- **Nivel de satisfacción del niño con el juego:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de las imágenes:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de la música:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Facilidad de aprendizaje y uso del juego:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Observaciones:** Diferencia unos botones de otros sabiendo para qué son cada uno de ellos

Niño 4

- **Nombre:** Andrea
- **Sexo:** Femenino
- **Edad:** 9
- **Curso escolar:** 4º de primaria
- **Lengua materna:** Castellano
- **Idioma del juego seleccionado:** Castellano
- **Niveles del juego seleccionados:** 5 y 3
- **Comprensión del juego:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Dificultad del juego:** 2
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Tiempo medio para resolver un subnivel:** 7"
- **Tiempo medio para resolver un nivel:** 3'
- **Utiliza el niño la ayuda previa del juego:** Si

- **En caso afirmativo, utilidad de esta ayuda:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Fluidez:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Preguntas hechas por el niño:** -
- **Nivel de satisfacción del niño con el juego:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de las imágenes:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de la música:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Facilidad de aprendizaje y uso del juego:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Observaciones:** En lugar de pantalón, escribe pantalones y no le fácil saber dónde se desactiva el sonido

Niño 5

- **Nombre:** Juan
- **Sexo:** Masculino
- **Edad:** 7
- **Curso escolar:** 1º de primaria
- **Lengua materna:** Castellano
- **Idioma del juego seleccionado:** Castellano
- **Niveles del juego seleccionados:** 1 y 7
- **Comprensión del juego:** 8
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Dificultad del juego:** 7
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Tiempo medio para resolver un subnivel:** 30"
- **Tiempo medio para resolver un nivel:** 10'

- **Utiliza el niño la ayuda previa del juego:** No
- **En caso afirmativo, utilidad de esta ayuda:** -
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Fluidez:** 4
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Preguntas hechas por el niño:** ¿Dónde se escribe?
- **Nivel de satisfacción del niño con el juego:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de las imágenes:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de la música:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Facilidad de aprendizaje y uso del juego:** 5
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Observaciones:** En lugar de maestra, escribe profesora. Diferencia perfectamente la función de cada botón

Niño 6

- **Nombre:** Clara
- **Sexo:** Femenino
- **Edad:** 7
- **Curso escolar:** 1º de primaria
- **Lengua materna:** Castellano
- **Idioma del juego seleccionado:** Castellano
- **Niveles del juego seleccionados:** 1 y 5
- **Comprensión del juego:** 7
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Dificultad del juego:** 6
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Tiempo medio para resolver un subnivel:** 18"

- **Tiempo medio para resolver un nivel:** 3'
- **Utiliza el niño la ayuda previa del juego:** Si
- **En caso afirmativo, utilidad de esta ayuda:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Fluidez:** 6
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Preguntas hechas por el niño:** ¿Qué hay que hacer?
- **Nivel de satisfacción del niño con el juego:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de las imágenes:** 8
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de la música:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Facilidad de aprendizaje y uso del juego:** 6
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Observaciones:** Le es fácil saber para qué sirve cada botón

Niño 7

- **Nombre:** Fernando
- **Sexo:** Masculino
- **Edad:** 8
- **Curso escolar:** 3º de primaria
- **Lengua materna:** Castellano
- **Idioma del juego seleccionado:** Castellano
- **Niveles del juego seleccionados:** 5 y 3
- **Comprensión del juego:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Dificultad del juego:** 3
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Tiempo medio para resolver un subnivel:** 20"

- **Tiempo medio para resolver un nivel:** 6'
- **Utiliza el niño la ayuda previa del juego:** Si
- **En caso afirmativo, utilidad de esta ayuda:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Fluidez:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Preguntas hechas por el niño:** -
- **Nivel de satisfacción del niño con el juego:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de las imágenes:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de la música:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Facilidad de aprendizaje y uso del juego:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Observaciones:** Utiliza de manera efectiva las pistas, escribe pantalones en lugar de pantalón, encuentra fácilmente el icono sonido del juego

Niño 8

- **Nombre:** Aiara
- **Sexo:** Femenino
- **Edad:** 6
- **Curso escolar:** 1º de primaria
- **Lengua materna:** Euskera y castellano
- **Idioma del juego seleccionado:** Euskera
- **Niveles del juego seleccionados:** 4 y 2
- **Comprensión del juego:** 8
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)

- **Dificultad del juego:** 7
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Tiempo medio para resolver un subnivel:** 50"
- **Tiempo medio para resolver un nivel:** 15'
- **Utiliza el niño la ayuda previa del juego:** No
- **En caso afirmativo, utilidad de esta ayuda:** -
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Fluidez:** 6
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Preguntas hechas por el niño:** -
- **Nivel de satisfacción del niño con el juego:** 8
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de las imágenes:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de la música:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Facilidad de aprendizaje y uso del juego:** 8
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Observaciones:** Está empezando a escribir

Niño 9

- **Nombre:** Carla
- **Sexo:** Femenino
- **Edad:** 9
- **Curso escolar:** 3º de primaria
- **Lengua materna:** Castellano
- **Idioma del juego seleccionado:** Castellano
- **Niveles del juego seleccionados:** 1 y 8
- **Comprensión del juego:** 7
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)

- **Dificultad del juego:** 5
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Tiempo medio para resolver un subnivel:** 8"
- **Tiempo medio para resolver un nivel:** 3'
- **Utiliza el niño la ayuda previa del juego:** Si
- **En caso afirmativo, utilidad de esta ayuda:** 4
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Fluidez:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Preguntas hechas por el niño:** -
- **Nivel de satisfacción del niño con el juego:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de las imágenes:** 9
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de la música:** 5
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Facilidad de aprendizaje y uso del juego:** 8
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Observaciones:** Le es fácil saber para qué sirve cada botón

Niño 10

- **Nombre:** Marcos
- **Sexo:** Masculino
- **Edad:** 7
- **Curso escolar:** 2º de primaria
- **Lengua materna:** Castellano
- **Idioma del juego seleccionado:** Castellano
- **Niveles del juego seleccionados:** 1 y 3
- **Comprensión del juego:** 8
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)

- **Dificultad del juego:** 7
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Tiempo medio para resolver un subnivel:** 28"
- **Tiempo medio para resolver un nivel:** 8'
- **Utiliza el niño la ayuda previa del juego:** No
- **En caso afirmativo, utilidad de esta ayuda:** -
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Fluidez:** 8
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Preguntas hechas por el niño:** ¿Cómo escribo? ¿Cómo se juega? ¿Dónde se borra?
- **Nivel de satisfacción del niño con el juego:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de las imágenes:** 7
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Opinión de la música:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Facilidad de aprendizaje y uso del juego:** 10
  (Del 1 al 10 siendo 1 el mínimo y 10 el máximo)
- **Observaciones:** El primer nivel seleccionado le resulta más fácil que el segundo, costándole este mucho más. Diferencia perfectamente los iconos del juego

Apéndice B

Manual de usuario

B.1. Introducción

Este manual pretende ser una guía para el usuario final en la que se muestran en funcionamiento las principales características de Guess the Pic desde un punto de vista no técnico. Las explicaciones se ilustran con numerosas capturas de pantalla con el fin de hacer hacer el proceso más visual y sencillo.

B.2. Requisitos

Para poder disfrutar del juego lo único que se necesita es lo siguiente:

1. Disponer de un dispositivo Android
2. Tener una cuenta en Google Play
3. Capacidad de 50MB libres en el dispositivo

B.3. Manual

Una vez está instalado el juego en el dispositivo, lo primero que hay que hacer es seleccionar el icono del juego, como el que se muestra en la figura B.1, situado en las aplicaciones.

Figura B.1: Icono del juego

Tras cargarse, se mostrará el menú principal, en el aparecen todas las opciones que ofrece el juego; Activar\Desactivar sonido, Jugar y Ayuda, de izquierda a derecha como se muestra en la figura B.2.

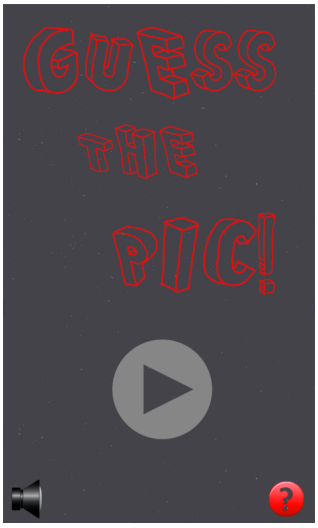


Figura B.2: Menú principal del juego

Si se desea activar o desactivar el sonido del juego, lo único que hay que hacer es seleccionar el icóno del altavoz para desactivarlo ( ) o activarlo ( ) y esta configuración se recordará para posteriores partidas.

Si lo que se desea es acceder a la ayuda, basta con pinchar en el icono ( ) de la parte inferior derecha del menú principal. Como consecuencia de esto, aparece la primera pantalla de ayuda, figura B.3.

Figura B.3: Primera pantalla de la ayuda

Para avanzar entre las pantallas de ayuda se debe pulsar la flecha ( ) situada en la parte inferior izquierda. Para retroceder, se debe pinchar la flecha ( ) situada al otro lado de la pantalla. Al llegar a la última pantalla de ayuda, figura B.4, se debe seleccionar el icono ( ), para volver al menú principal, al igual que si se encuentra en la primera pantalla de ayuda y se pincha sobre retroceder ( ).

Figura B.4: Última pantalla de ayuda

En esta ayuda, nos explica qué pasos se deben seguir para comenzar a jugar,

así como cuál es el objetivo del juego y los temas que se tratan en cada nivel.

La última opción que ofrece el juego es la de jugar. Para ello, habrá que pulsar en el icono ( ) del menú principal para comenzar con la partida.

La primera pantalla que aparece es la de selección de idioma, figura B.5. Es decir, a partir de este momento el juego se mostrará en el idioma elegido. Se tiene que elegir entre los tres idiomas disponibles, castellano ( ), inglés ( ) o euskera ( ).

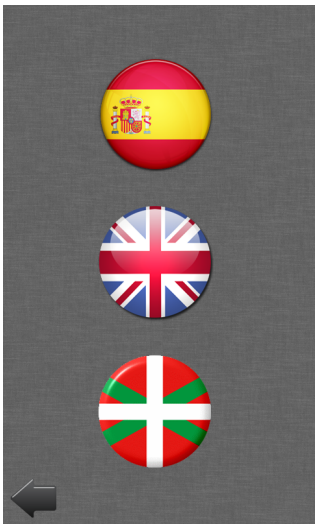


Figura B.5: Selección de idioma

Una vez seleccionado el idioma en el que se desea jugar, hay que seleccionar el nivel, figura B.6. Hay ocho niveles en total y cada uno de ellos se corresponde con una categoría diferente.

1. Animales
2. Números y colores
3. Alimentos
4. Meses, días de la semana y estaciones del año
5. Ropa y partes del cuerpo
6. Medios de transporte
7. Profesiones
8. Escenas de la vida cotidiana

Figura B.6: Selección de nivel

Se puede seleccionar el nivel que se desee independientemente de si se ha superado el anterior o no. Para ello, bastará con seleccionar el rectángulo de cada nivel ().

Cuando se selecciona el nivel, a continuación se cargan los subniveles pertenecientes a ese nivel. Por cada uno de ellos, hay 20 subniveles diferentes. En este caso, sí es importante el hecho de haber desbloqueado los subniveles anteriores. Es decir, como se puede ver en la figura B.7, cuando comienza la partida, sólo el subnivel 1 estará desbloqueado mientras que el resto de los subniveles permanecerán desbloqueados. A medida que se vayan completando, se irán desbloqueando.

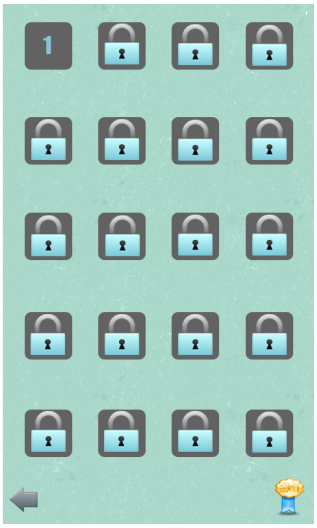


Figura B.7: Selección de subnivel

Desde esta pantalla se puede volver a la selección de niveles pinchando sobre la flecha ( ), que aparece en la parte inferior izquierda de la pantalla, obtener la mejor puntuación del nivel pinchando en el botón ( ) de la otra parte de la pantalla. En este punto pueden ocurrir dos cosas diferentes. Si es la primera partida del nivel, se mostrará un mensaje avisando de que no se han completado los 20 subniveles, figura B.8 de la izquierda, mientras que si ya se han completado, se mostrará la mejor puntuación conseguida en ese nivel, figura B.8 de la derecha. Para salir de esta pantalla, basta con tocar en cualquier parte de la misma y el juego se redigirá a la primera pantalla de selección de nivel.

Figura B.8: Obtener puntuaciones

Por último se puede comenzar con el juego. Para ello, se debe seleccionar el subnivel () en el que se desea jugar, siendo lo normal el último nivel desbloqueado. Si se da el caso y se pincha sobre cualquier subnivel bloqueado, como se ve en la figura B.9 el juego únicamente mostrará un mensaje indicándolo y no ocurrirá nada más.

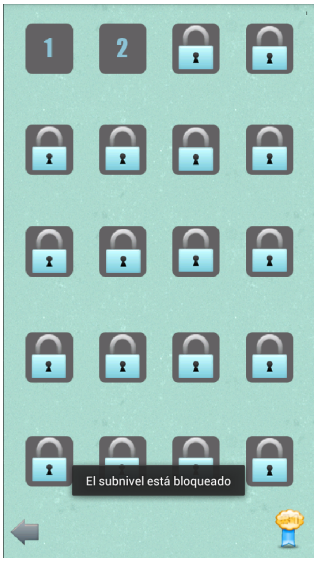


Figura B.9: Selección de un subnivel bloqueado

Una vez seleccionado el subnivel, se cargará la pantalla principal del juego, figura B.10. En la parte superior izquiera, está situado el botón de pausa ( ), en la parte superior derecha, el tiempo empleado por el Jugador y los intentos realizados. Mientras que en el centro de la pantalla se encuentra la imagen a adivinar. Justo debajo de ella, un cuadro de texto en el que se debe escribir la respuesta, el botón de comprobar que habrá que pincharlo para comprobarla, y por último un botón para obtener la pista de la solución.

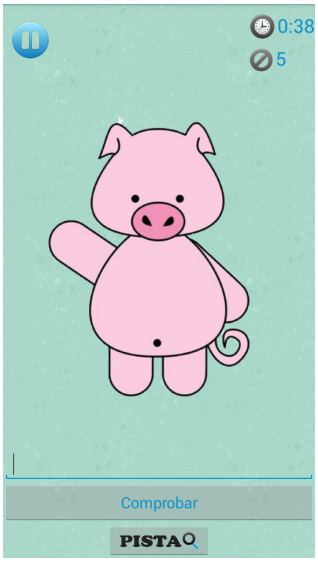


Figura B.10: Pantalla del juego

Para parar el juego, basta con seleccionar el botón de pausa ( ). Entonces aparecerá la siguiente pantalla, figura B.11.

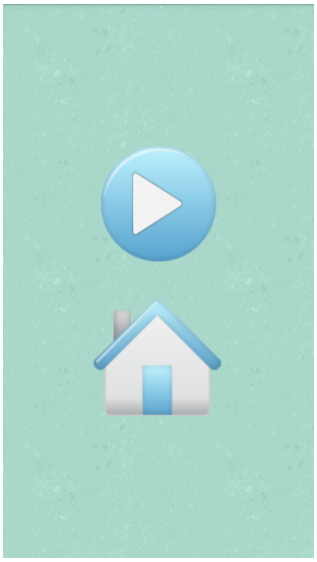


Figura B.11: El juego se encuentra pausado

Si se pulsa sobre el botón de play ( ), el juego se reanudará, mientras que si se pincha sobre el icono home ( ), el juego volverá a la pantalla del menú principal.

Cuando se completan los 20 subniveles, el juego muestra las puntuaciones del nivel y la próxima vez que se acceda a dicho nivel se encontrará de nuevo los subniveles bloqueados.