UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

APLICACIÓN ANDROID PARA AUTOEVALUACIÓN DE CONOCIMIENTOS EN ELECTRÓNICA DIGITAL

Iván Leal Cabrera
Tutor: Eduardo Boemo Scalvinoni

Junio 2018

APLICACIÓN ANDROID PARA AUTOEVALUACIÓN DE CONOCIMIENTOS EN ELECTRÓNICA DIGITAL

AUTOR: Iván Leal Cabrera
TUTOR: Eduardo Boemo Scalvinoni

Digital System Laboratory
Dpto. Tecnología Electrónica y de Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio de 2018

Resumen (castellano)

Este Trabajo de Fin de Grado tiene como objetivo el desarrollo de una aplicación para dispositivos con Sistema Operativo Android. Se trata de una aplicación dedicada a la autoevaluación de conocimientos en Electrónica Digital.

Para ello, se genera un examen tipo test compuesto de una colección de preguntas, con cuatro posibles respuestas y la posibilidad de incorporar imágenes.

La aplicación sólo permite la selección de una posible respuesta y posee la capacidad de autocorrección. A su vez, dispone de un solucionario a partir del cual se puede obtener el resultado de la nota final del test, así como el número de preguntas acertadas, las falladas y las que no se han contestado.

Como características adicionales, dispone de sonidos y también de diferentes modelos de examen.

La aplicación es compatible con diferentes tamaños y densidades de pantalla, se puede ejecutar tanto en smartphones como en tabletas.

Está diseñada para su utilización en entornos educativos, en empresas o para un uso particular. Además, cuenta con una serie de interfaces de ayuda entre las diferentes interacciones entre el usuario y el dispositivo que facilitan y simplifican su utilización.

Además, la aplicación permite la selección de dos modos diferentes para realizar el examen; el modo estudiante, el cual dispone de tiempo ilimitado para responder, y el modo profesor habilitado con un contador de tiempo descendente en el que se puede elegir la duración del mismo.

Además, la aplicación dispone de diferentes grupos de preguntas:

- **Full Test**: Contiene todas las preguntas de la colección.
- **FPGA**: Contiene preguntas sobre “field-programmable gate array”.
- **EE Basic Concepts**: Contiene preguntas sobre conceptos básicos de la electrónica.
- **Asic Design:** Contiene preguntas sobre circuitos integrados para aplicaciones específicas.
- **EE Culture:** Contiene preguntas de cultura general de la electrónica.
- **Logic Design:** Contiene preguntas de diseño lógico.

La aplicación se llamará EETest y estará disponible para su descarga en la plataforma Google Play Store de forma gratuita.

Abstract (English)

This Project focus on the development of an application for devices with Android Operating System. This purpose of this application is the evaluation of knowledge in Digital Electronic.

In order to do so, a test has been created with a series of questions and four potential answers as well as the possibility of adding images.

This application only allows selecting one answer and it is able to correct itself. Moreover, it contains the answers in which the final score is obtained from as well as the number of questions correctly answered, the failed ones and the ones unanswered.

As additional features, this application provides different sounds and different types of exam.

This application is compatible with screen of different sizes and thickness and it can be used on smartphones and tablets.

This application can be used in an education environment, professional or personal one. Moreover, it contains varius help settings between user and aplication that facilite and simplify its use.

Moreover, this application allows the selection of two settings to perform the exam; student mode, which provides unlimited time to answer the questions, and teacher one, which has a stopwatch to select the time.

Last but not least, the application has the following types of questions:

- **Full test**: contains all the questions available in the app.
- **EPGA**: contains questions about “field programmable gate array”
- **EE Basic Concepts**: contains questions about basic electronic concepts.
- **Asic Design**: contains questions about integrated circuits for specific applications
- **EE Culture**: contains general questions about digital culture.
- **Logic Design**: contains questions about logic design.

This application will be called EETest and will be available for download in Google Play Store free of charge.

Palabras clave (castellano)

Android, pantalla, test, electrónica, digital, examen, aplicación, Java, Smartphone, tableta, dispositivo, interfaz.

Keywords (inglés)

Android, screen, test, electronics, digital, exam, application, Java, Smartphone, Tablet, device, interface.

Agradecimientos

Finalizado el presente proyecto, quisiera dedicar este apartado para agradecer el apoyo y la colaboración recibida por parte de todas las personas que han hecho posible llevarlo a cabo.

En primer lugar, agradezco a mi tutor de este trabajo, Eduardo Boemo Scalvinoni por la oportunidad de desarrollar el proyecto en el DSLab, por su dedicación y su interés.
Agradecer también a todos los profesores de la escuela Politécnica superior de la UAM por su profesionalidad, su sabiduría y por conducirme hasta el punto donde me encuentro hoy.

A todos mis compañeros de facultad con los he compartido alegrías y tristezas.
En especial a mi pareja, Irene, por el apoyo incondicional, por acompañarme en los momentos más duros, desafiándome y consiguiendo sacar lo mejor de mí, por tus consejos y por haberme impulsado a alcanzar este sueño.

Por último, a mi familia, ya que sin ellos nada de esto habría sido posible. Gracias por el apoyo y por estar siempre conmigo.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

1 Introducción

1.1 Motivación

En la actualidad observamos cómo los dispositivos móviles están cada vez más presentes en nuestra vida, nos aportan un sinfín de posibilidades debido a que son una herramienta cuyo desarrollo es continuo e ilimitado. Debido a las características de la sociedad en la que vivimos, marcadas por factores como el uso y disfrute de un dispositivo móvil por parte de casi la totalidad de la población y la necesidad en el mercado de aplicaciones especializadas en electrónica digital, surge la idea de lanzar una aplicación que reúna conocimientos en distintas áreas de la electrónica digital para poder ayudar a cualquier usuario a evaluar sus conocimientos sobre las diferentes materias que la aplicación aborda, así como aprender de las correcciones del mismo.

Está orientada tanto para su uso educativo como para su uso empresarial.

Uso educativo:

- Permite que el/la estudiante realice una serie de auto-test en un tiempo limitado.
- Mide conocimientos con una nota numérica de 0 a 10.
- Permite elegir sub-temas de evaluación.
- Permite al profesor evaluar a los/las estudiantes y ver cuáles son las carencias (por ejemplo, no saber las Leyes de Kirchoff).
- Nivel de educación universitario, aunque puede valer para centros de formación profesional.

Uso profesional en empresas:

- Procesos de selección.
- Procesos de promoción.

Esta aplicación será complementada por otras dos acciones:

- Un libro de electrónica digital, con la teoría relacionada con las preguntas de los temas.
- Una app de soluciones detalladas de todas las preguntas (de pago).
- Versiones iOS.

El esquema se repetirá sobre una serie de nuevas apps que condensan cerca de 30 años de docencia, preparación de material y exámenes de electrónica digital. Cada una de ellas estará compuestas por 255 nuevas preguntas.

Mi entusiasmo por la electrónica digital y la visión de ayuda y aprendizaje que proporcionará la tecnología mediante el desarrollo de una Aplicación (App) para los usuarios, hacen que surja este desafío y me lance a hacer realidad la idea planteada.

Algunas de las ventajas destacables son:

- No se necesita conexión a internet, es accesible en cualquier momento y lugar.
- Corrección de los exámenes de forma automática.
- Experiencia de usuario fácil e intuitivo.
- Eliminación del formato físico en papel.
- Comodidad y visualización incluso en entornos poco iluminados.

1.2 Objetivos

El objetivo de este Trabajo de Fin de Grado es desarrollar una aplicación Android que permita a los usuarios autoevaluar sus conocimientos en electrónica digital, así como utilizarla como herramienta de aprendizaje en las diferentes áreas de la electrónica que la aplicación aborda.

La aplicación ingresará una lista de preguntas y respuestas en formato estándar y generará un examen tipo test de respuesta múltiple con capacidad de autocorrección. Contará con un contador de tiempo, sonidos, diferentes materias en las que examinarse, posibilidad de incluir imágenes y una tabla de resultados.

Irá dirigida a cualquier persona interesada en evaluar sus conocimientos en electrónica digital, ya sea para un fin docente, empresarial o para un interés particular del usuario.

En cuanto a los propósitos personales este proyecto será útil para desarrollar mis conocimientos en Java, en programación XML, aprender a utilizar el entorno de desarrollo Android Studio, el uso del emulador de Android AVD Manager, manejo de herramientas de diseño gráfico y adaptar aplicaciones Android a diferentes dispositivos.

1.3 Organización de la memoria

Este documento tiene como objetivo exponer el desarrollo del Trabajo de Fin de Grado sobre una aplicación para el sistema operativo Android, con el fin de obtener una herramienta de aprendizaje para todos los usuarios, donde a partir de sus resultados, servirá de mejora en sus conocimientos en electrónica digital.

Para ello, la memoria consta de los siguientes capítulos:

- Capítulo 1: Introducción.
- Capítulo 2: Estado del Arte.
- Capítulo 3: Diseño.
- Capítulo 4: Desarrollo.
- Capítulo 5: Integración, pruebas y resultado.
- Capítulo 6: Conclusiones y trabajo futuro.

2 Estado del arte

Como se menciona anteriormente, el objetivo de este proyecto es el desarrollo de una aplicación móvil para autoevaluación de conocimientos en electrónica digital.

El primer paso ha sido realizar un estudio en la plataforma Google Play Store para conocer diferentes aplicaciones disponibles que puedan ser similares y que aborden aspectos parecidos.

2.1 Stuck At

Es una aplicación gratuita disponible en Google Play Store que permite a los usuarios realizar una serie de ejercicios y un test sobre el modelo de fallos "stuck-at" desde su dispositivo móvil.

La aplicación permite comprobar errores cometidos y resolver de forma automática los problemas, pudiendo guardar soluciones, además de permitir enviarlas a través del correo electrónico.

Esta App se centra únicamente en el modelo de fallos "stuck-at" dentro de la electrónica digital y además dispone de test, pero se puede catalogar como carente ya que solo dispone de 10 preguntas.

2.2 Digital Electronics 101

Es una aplicación disponible tanto de manera gratuita como en versión de pago que se puede descargar desde Google Play Store. Se trata de una aplicación que revisa los fundamentos de la electrónica digital a través de preguntas de opción múltiple, las cuales incluyen solución y explicación teórica. Dispone de una amplia colección de preguntas que abordan temas como son los circuitos lógicos, conceptos básicos, multiplexores, puertas lógicas, "flip flops", etc.

Ésta App es la más completa respecto al contenido en electrónica digital, y su dinámica en cuanto a preguntas con opción múltiple. Todo ello, hace que sea la que más similitud guarda con respecto a este proyecto.

Una posible desventaja encontrada en la aplicación, es que en la versión gratuita cuenta con publicidad lo cual incomoda su utilización. Además no es muy fluida, ya que cada pregunta que se responde va informando de si se ha acertado o se ha fallado, no da opción a completar el examen y obtener las soluciones al final del mismo. Además si se falla una pregunta solo da la opción de volver a responder la misma o salir al menú principal anterior, no deja continuar con el test.

2.3 Learn Digital Electronics

Es una aplicación gratuita también disponible en Google Play Store que reúne conocimientos teóricos sobre electrónica digital divididos en capítulos.

Es una App con un temario muy completo, se divide en diferentes secciones como son los sistemas numéricos, los códigos binarios, puertas lógicas, transistores, constructor de circuitos, tablas de verdad, "flip flops", multiplexores, decodificadores etc.

Es una herramienta de aprendizaje muy útil ya que tiene un temario muy amplio, aunque como desventaja, no dispone de ningún método para resolver ejercicios ni para resolver preguntas tipo test, por lo que no podemos evaluar nuestros conocimientos con el uso de ésta aplicación.

Además incluye publicidad lo cual dificulta bastante su uso en ciertos momentos.

2.4 Digital Electronics Quiz

Es una aplicación disponible de manera gratuita a través de la plataforma Google Play Store que contiene un cuestionario de más de mil preguntas sobre electrónica digital. Se puede ejecutar sin conexión a internet desde cualquier lugar.

La aplicación cuenta con publicidad y además su uso no es muy dinámico ya que cada pregunta que se responde, va informando si se ha contestado bien o mal y no te deja hacer un test completo para obtener la solución al final del mismo.

2.5 GoDiGi (Digital Electronics)

Es una aplicación disponible en la plataforma Google Play Store de forma gratuita. Es bastante completa ya que tiene contenidos teóricos sobre electrónica digital, dispone de conversor numérico de unas bases a otras y también simplifica expresiones booleanas en el mapa de Karnaugh, además de incluir un test para evaluar los conocimientos.

La aplicación tiene un límite de diez preguntas en el test de evaluación de conocimientos lo cual se queda algo corto y además también incluye publicidad.

2.6 Resumen

Una vez estudiadas las aplicaciones similares que hay en el mercado, se llega a la conclusión de que sólo hay una App (Digital en Electronics 101) con la que realmente se pueden evaluar conocimientos en electrónica digital en profundidad. El resto de aplicaciones o no tienen forma de evaluación mediante test o en alguna que si lo tiene el número de preguntas es escaso.

A continuación, se muestra una tabla comparativa entre las aplicaciones estudiadas y la aplicación desarrollada en el proyecto:

APP	Test	Contenido Teórico	Interfaz	Soluciones	Icono
Stuck At	Si	No	Simple e intuitiva	Si	
Digital Electronics 101	Si	Si	Simple pero poco fluida	Si	
Learn Digital Electronics	No	Si	Simple e intuitiva	No	
Digital Electronics Quiz	Si	No	Poco fluida	Si	
GoDiGi	Si	Si	Simple	Si	
EETest	Si	No	Simple e intuitiva	Si	

Figura 2-1: Tabla comparativa de aplicaciones similares. (Fuente: elaboración propia).

3 Diseño

3.1 Objetivos de diseño

Este proyecto tiene como finalidad el desarrollo de una aplicación Android para facilitar el aprendizaje a los usuarios de manera interactiva mediante la resolución de test de electrónica digital.

Cuenta con una colección de preguntas muy amplia dividida a su vez en 6 materias diferentes cuya pretensión será que cualquier persona sea capaz de aprender desde cualquier lugar de manera muy fácil con su dispositivo móvil, incluso si éste no dispone de conexión a internet.

Además dispone de diferentes modos de realizar el examen, posibilidad de incluir imágenes en las preguntas, temporizador de cuenta atrás y capacidad de autocorrección.

3.2 Requisitos de diseño

En éste apartado se determinarán las diferentes características y requisitos con los que deberán de contar los dispositivos móviles para que la aplicación pueda ser ejecutada correctamente.

3.2.1 Elección del sistema operativo

Lo primero que se debe considerar antes de comenzar con el desarrollo de la aplicación planteada será estudiar las diferentes posibilidades que hay en el mercado en cuanto al sistema operativo (SO). La elección irá ligada a un sistema operativo y por tanto estará disponible para un número determinado de dispositivos, se utilizará una plataforma de desarrollo diferente o se utilizará un lenguaje de programación distinto entre otras cosas.

Es por ello por lo que se analizarán los principales sistemas operativos que existen en la actualidad.

Lo primero en lo que estará basada su elección será en el número de móviles que utilizan un sistema operativo u otro. Se observa que el sistema operativo predominante en el mundo en abril de 2018 es Android con un 75,66% del total de dispositivos, después le sigue iOS con un 19,23% y por último Windows con un 0,75% según Statcounter.

A continuación, se muestra un gráfico del porcentaje de dispositivos móviles que utilizan un sistema operativo u otro en el último año:

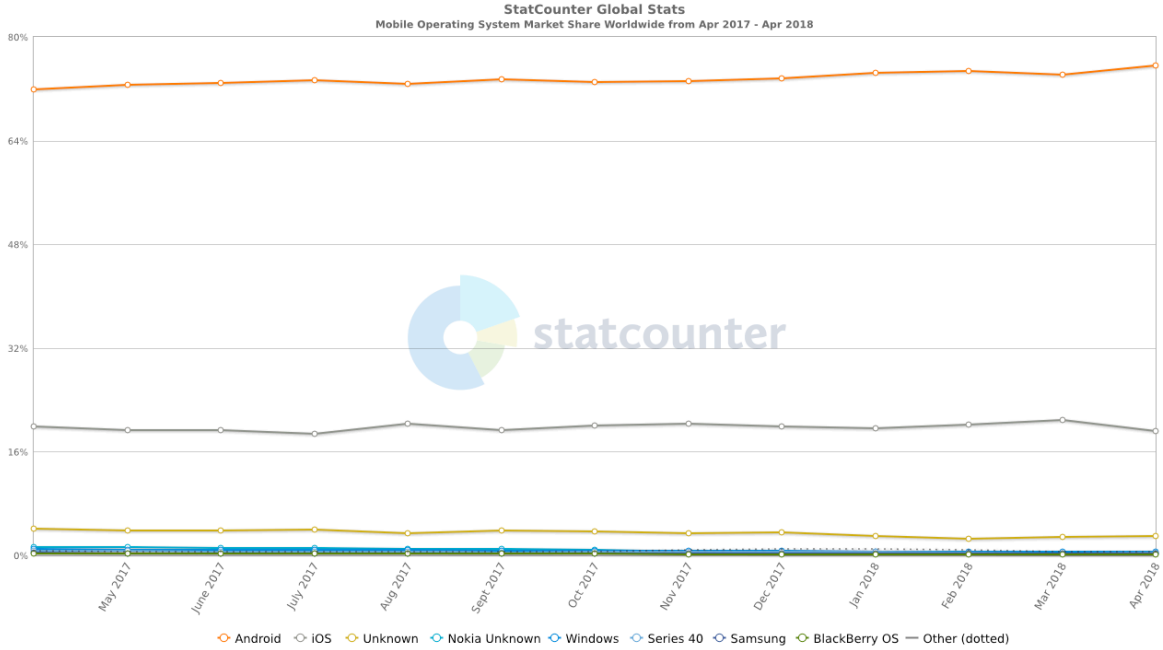

Figura 3-1: Gráfico comparativo entre SO en dispositivos móviles. (Fuente: Statcounter).[1]

Se observa una clara predominancia del SO Android frente al resto.

Un punto a tener en cuenta es que los sistemas operativos puedan determinar su elección, por ejemplo, si es un sistema operativo libre o no lo es. En el caso de Android se trata de un sistema operativo de código abierto completamente libre, cuyas herramientas de desarrollo son totalmente gratuitas, mientras que en el caso de iOS no es de código abierto y sus herramientas son de pago. Android además es multiplataforma, por lo que se podrá desarrollar en Windows, en Linux o en Mac OS, otro punto a favor, ya que en iOS no ocurre igual. Otro punto a tener en cuenta es el precio de subir la aplicación a la plataforma correspondiente una vez acabado su desarrollo, y es que la cuenta como desarrollador iOS que te permite subir aplicaciones a la "App Store" es de 99 dólares anuales frente a los 25 dólares que cuesta una licencia de desarrollador Android para subir aplicaciones a la "Google Play Store", que además es un pago único ya que esa licencia es vitalicia, no es necesario renovarla anualmente.

Debido a las facilidades que proporciona Android en cuando al desarrollo de aplicaciones, la facilidad de herramientas gratuitas y código abierto, el precio de mantenimiento de la licencia de desarrollador y la cuota de mercado que abarca, se decide elegir éste sistema operativo para el desarrollo de éste proyecto.

3.2.2 Versiones compatibles

Una vez escogido el sistema operativo hay que determinar en cuales de las diferentes versiones de ese SO será compatible la aplicación.

A continuación se muestran los datos sobre la cantidad de dispositivos que utilizan una versión determinada de la plataforma de Android.

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.3%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.4%
4.1.x	Jelly Bean	16	1.5%
4.2.x		17	2.2%
4.3		18	0.6%
4.4	KitKat	19	10.3%
5.0	Lollipop	21	4.8%
5.1		22	17.6%
6.0	Marshmallow	23	25.5%
7.0	Nougat	24	22.9%
7.1		25	8.2%
8.0	Oreo	26	4.9%
8.1		27	0.8%

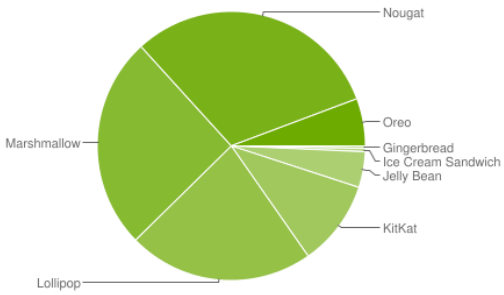

Figura 3-2: Distribución de dispositivos Android por versión del SO, enero 2018. (Fuente: Android Developers).[2]

El propósito de la creación de esta App es llegar a la mayor cantidad de usuarios posibles, por lo que se ha decidido que la aplicación se podrá instalar en dispositivos que cumplan con unos requisitos mínimos de versión del sistema operativo. Para ello, estará disponible en dispositivos con versiones entre la "Ice Cream Sandwitch" (API=15) y la más actual.

La elección de las versiones soportadas del SO se encuentra configurada en éste proyecto en el archivo "AndroidManifest.xml".

3.2.3 Pantallas compatibles

La presente aplicación sólo será compatible para tabletas y smartphones, ya que en otros dispositivos no sería tan útil. Se busca un fin docente y se ha considerado que ésta elección es la más acertada.

Android se puede ejecutar en dispositivos con diferentes tamaños y densidades de pantalla, siendo el mismo sistema operativo Android quien se encarga de ajustar las aplicaciones a la interfaz de la pantalla en la que se esté ejecutando.

Las principales características que tenemos que analizar para determinar en qué dispositivos será compatible la presente aplicación son:

- Tamaño de la pantalla → Es el tamaño físico real de la pantalla del dispositivo. Es la medida de la longitud de la diagonal de la pantalla.

En Android se han encontrado cuatro clasificaciones en las cuales se agrupan los dispositivos según sea la longitud de la diagonal de su pantalla tomando como unidad de medida las pulgadas:

  - Small (pequeño) miden como mínimo 426x320dp, diagonal de 2 a 3 pulgadas.

- Normal (normal) miden como mínimo 470x320dp, diagonal de 3 a 4.5 pulgadas.
- Large (grande) miden como mínimo 640x480dp, diagonal de 4.5 a 7 pulgadas.
- Xlarge (extragrande) miden como mínimo 960x720dp, diagonal de 7 a 10 pulgadas.

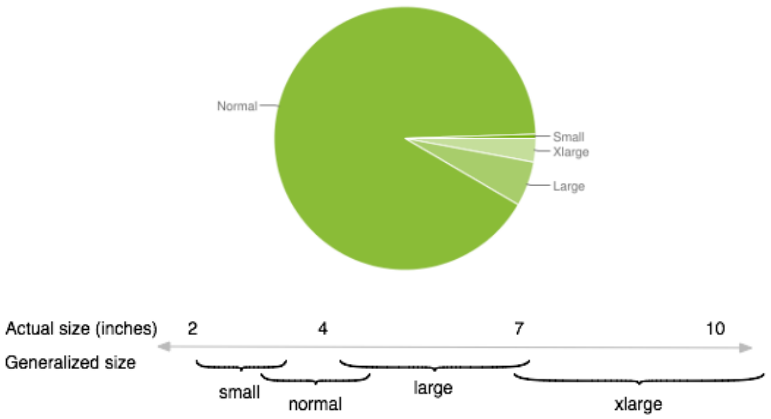

Figura 3-3: Distribución de dispositivos Android por tamaño de pantalla, enero 2018. (Fuente: Android Developers).[3]

- Densidad de pantalla → Es la cantidad de píxeles que hay dentro del área física de la pantalla medida en puntos por pulgada "dpi". En Android se agrupan en seis categorías generales:
  - ldpi (baja) ~120 dpi.
  - mdpi (media) ~160 dpi.
  - hdpi (alta) ~240 dpi.
  - xhdpi (extraalta) ~320 dpi.
  - xxhdpi (extra extraalta) ~480 dpi.
  - xxxhdpi (extra extra extraalta) ~640 dpi.

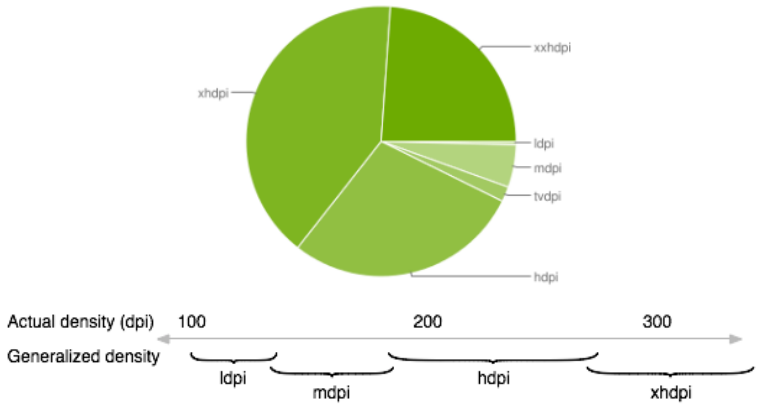

Figura 3-4: Distribución de dispositivos Android por densidad de pantalla, enero 2018. (Fuente: Android Developers).[3]

- Orientación de la pantalla → Es la orientación en la que se ejecutara la aplicación en la pantalla del dispositivo desde el punto de vista del usuario. Las posibilidades de la orientación de la pantalla son la horizontal y la vertical.

Una vez analizadas las diferentes posibilidades que contempla Android en cuando a la compatibilidad de pantallas, se procede a decidir las especificaciones que tendrá la aplicación, las cuales determinarán en qué dispositivos se podrá instalar la misma.
Para ello, se ha relacionado la distribución de dispositivos Android según su tamaño y densidad de pantalla:

	ldpi	mdpi	tvdpi	hdpi	xhdpi	xxhdpi	Total
Small	0.4%					0.1%	0.5%
Normal		0.9%	0.3%	27.3%	39.3%	23.3%	91.1%
Large		2.4%	1.5%	0.4%	0.7%	0.5%	5.5%
Xlarge		1.8%		0.6%	0.5%		2.9%
Total	0.4%	5.1%	1.8%	28.3%	40.5%	23.9%	

Figura 3-5: Distribución de dispositivos Android por tamaño y densidad de pantalla, enero 2018. (Fuente: Android Developers).[1]

De esta distribución se puede determinar que los tamaños en los que se podrá ejecutar la presente aplicación para llegar al mayor número de dispositivos posibles serán en tamaño "Normal" y en tamaño "Large". La gran mayoría de smartphones y tabletas Android del mercado se mueven en ese tamaño, coincidiendo ese grupo de dispositivos con el abordado en este proyecto.

Además en cuanto a la densidad de pantalla, nuestra aplicación podrá ejecutarse para cualquier dispositivo, ya que Android dispone de la posibilidad de redimensionar las imágenes que incorpore nuestra aplicación de forma automática, pero aun así cada una de las imágenes que se utilice en nuestra App estará disponible en las tres densidades de pantalla más comunes que son la "hdpi", "xhdpi" y "xxhdpi".

Respecto al icono de la aplicación que veremos en la pantalla principal de un dispositivo una vez descarguemos la App, se ha tenido en cuenta también que sea compatible con todas las densidades de pantalla, y además está disponible en formato cuadrado y en formato circular. Según tenga configurado el cliente el diseño de los iconos de las aplicaciones de su dispositivo, se mostrará el diseño en una densidad de pantalla determinada y con una forma tanto cuadrada como redonda.

Por último, se ha diseñado la aplicación para que se pueda ejecutar únicamente en orientación vertical, ya que la dinámica de la App es un examen tipo test y se ha decidido que la interfaz visual queda más estética en esta orientación.

Para configurar tanto los tamaños de pantalla compatibles, las diferentes densidades, la orientación de la App o la posibilidad de redimensionar las imágenes de forma automática, se tendrá que configurar todo ello en el archivo "AndroidManifest.xml" del proyecto.

3.2.4 Idioma de la aplicación

El último requisito de diseño que se ha tenido en cuenta antes de comenzar con el desarrollo del proyecto, es el idioma en el que estaría disponible. Para ello, se ha realizado un estudio sobre los idiomas más utilizados en internet. En cuanto a la comunicación de las personas, el idioma más utilizado en el mundo es el chino, estando éste por encima del idioma inglés. A pesar de ello, en Internet, el idioma más utilizado es el inglés, siendo éste el motivo por el cual se ha decidido el desarrollo de la presente aplicación completamente en inglés.

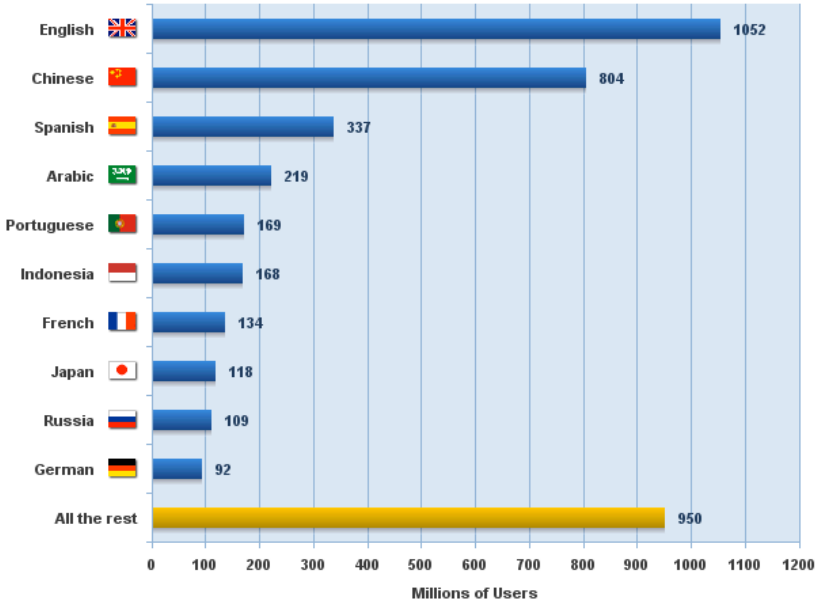

Figura 3-6: Top 10 de idiomas utilizados en Internet, diciembre 2017. (Fuente: Internet World Stats).[4]

3.2.5 Resumen de los requisitos de diseño

Este apartado recoge los principales requisitos que se han elegido para el diseño de la aplicación por los motivos anteriores descritos:

- Sistema operativo: Android
- Versiones compatibles: desde la versión "Ice Cream Sandwitch" (API=15) hasta las más actual.
- Pantallas compatibles: las de tamaño "Normal" o "Large", en todas las densidades de pantalla "ldpi", "mdpi", "hdpi", "xhdpi", "xxhdpi" y "xxxhdpi". Además se ejecutará en orientación vertical.
- Idioma: Inglés.

3.3 Diagrama de Bloques

Como se menciona anteriormente, este proyecto tiene como fin el desarrollo de una aplicación Android para la autoevaluación de conocimientos en electrónica digital. Su finalidad es que la persona que se descargue la aplicación en su terminal pueda evaluar sus conocimientos y pueda a su vez adquirir otros nuevos.

Para la realización del proyecto se ha intentado hacer que la interfaz sea intuitiva para el usuario. Por ello, se ha seguido una estructura en forma de árbol, mediante la cual la navegación sea lo más elemental posible. A ella, se puede acceder de forma sencilla, desde el menú principal hasta el desarrollo del test, así como también a las etapas posteriores al test.

El siguiente diagrama representa la estructura de la aplicación:

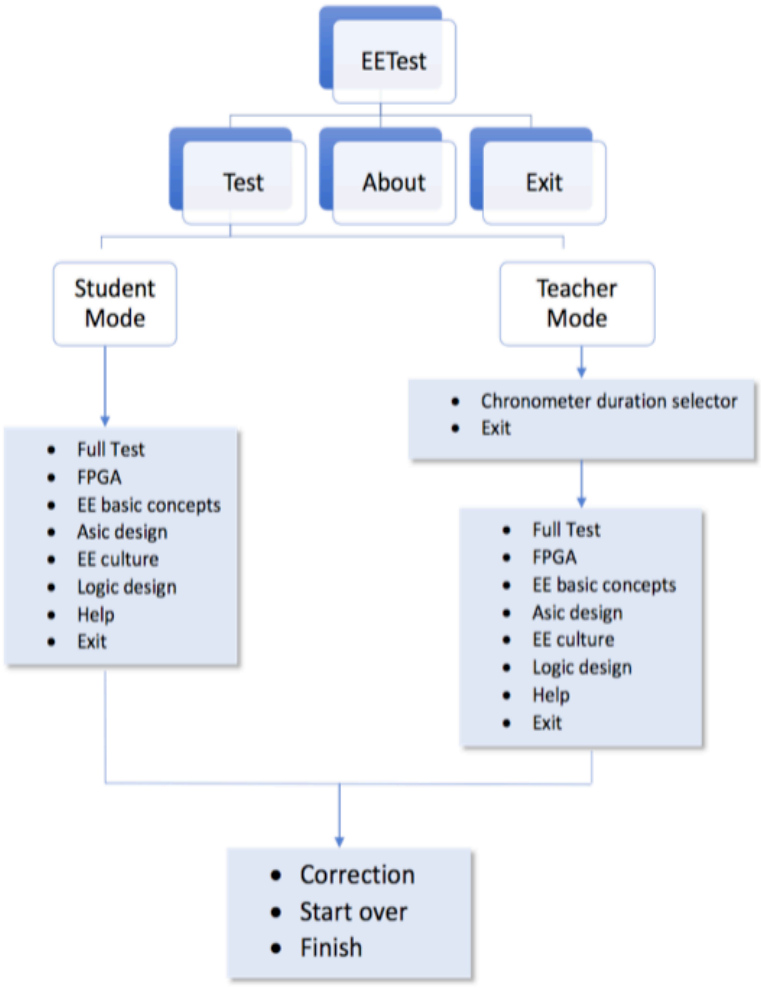

Figura 3-7: Diagrama representante de la estructura de la aplicación. (Fuente: elaboración propia).

3.4 Estructura de la aplicación

En esta sección se expondrán de manera resumida las diferentes partes de las que consta la aplicación, viendo las diferentes interfaces y menús por los que se puede navegar.

3.4.1 Menú principal

El menú principal es la primera pantalla que se mostrará en la aplicación, desde la cual se puede acceder a diferentes opciones:

- Test.
- About (acerca de).
- Exit (salir).

3.4.2 Test

Desde esta opción se accede a un segundo menú desde el cuál se pueden seleccionar los diferentes modos de realización del examen. A través de esta pantalla se posibilita el acceso a las siguientes opciones:

- Student Test Mode (modo estudiante).
- Teacher Test Mode (modo profesor).
- Help (Ayuda).
- Exit (salir).

3.4.3 About

Esta elección es simplemente informativa. En ella se encuentra el contenido de información del desarrollador, información del tutor del proyecto y del laboratorio en el cual se ha desarrollado el mismo.

3.4.4 Exit

La opción “Exit” (salir), está habilitada para cerrar la actividad en cualquier pantalla en la que encontremos la opción de salir, si pulsamos en ella hará que se cierre la actividad o menú en el que nos encontramos llevándonos al paso anterior. Esto ocurre así en toda la aplicación excepto en el menú principal en el cual si elegimos ésta opción estaremos saliendo de la aplicación.

3.4.5 Student Test Mode

Ésta opción permite realizar el examen en modo estudiante. La característica que presenta este modo, es que no tiene un límite de tiempo para realizar el test. Se tendrá que elegir la materia deseada para examinar.

Además incluye una opción de ayuda y otra para salir de este modo, pudiendo encontrar las siguientes opciones:

- Full Test.
- FPGA.
- EE Basic Concepts.
- Asic Design.
- EE Culture.
- Logic Design.
- Help.
- Exit.

3.4.6 Teacher Test Mode

Ésta opción permite hacer el examen en modo profesor. Éste modo tiene la característica de contar con un cronómetro que indicará la duración del examen. Al pulsar en éste modo se accederá a una pantalla de selección de horas y minutos para poder programar el tiempo de duración del examen. Una vez se selecciona el tiempo de duración, la aplicación conducirá a un menú de selección de la materia igual que en el modo estudiante con las siguientes secciones:

- Full Test.
- FPGA.
- EE Basic Concepts.
- Asic Design.
- EE Culture.
- Logic Design.
- Help.
- Exit.

3.4.7 Help

En esta sección se muestra una ayuda al cliente para indicar información útil sobre los diferentes menús que tiene la aplicación. Dependiendo de la pantalla de menú en la que encontremos esta opción de ayuda, podrá conducir a una ayuda u otra diferente en la cual se explican aspectos importantes de ese apartado mediante contenido de información y apoyo para el usuario.

3.4.8 Cuadro de diálogo de fin de Test

Alcanzado el final de la realización del examen, aparecerá un cuadro de diálogo con información relevante del test recientemente concluido. En éste cuadro de diálogo se cuentan con varias opciones que podemos seleccionar:

- Correction.
- Start Over.
- Finish.

3.4.9 Cuadro de diálogo para abandonar un examen

Durante la realización de cualquiera de los exámenes se podrá salir y poner fin al mismo desde un botón disponible en todo momento durante el transcurso del test. Antes de salir de forma definitiva, aparecerá un cuadro de diálogo que avisará del abandono del test, preguntando si realmente lo quiere abandonar o prefiere seguir haciendo el examen. Es un cuadro de diálogo que verifica si el usuario quiere salir de la actividad y dar por concluido el examen.

3.4.10 Correction

Esta opción permitirá hacer una corrección del examen que se acaba de hacer. La aplicación tiene las soluciones de cada pregunta del test, y con las soluciones y el examen realizado es capaz de realizar su corrección.

3.4.11 Start Over

Si se opta por ésta opción, tenemos la posibilidad de comenzar de nuevo el mismo examen que se acaba de realizar y empezar desde el principio.

3.4.12 Finish

Ésta es la opción que permite concluir la realización del examen. Con esta elección se concluye el test de manera definitiva.

3.4.13 Preguntas de la aplicación

La aplicación está dimensionada para albergar 255 preguntas tipo test. Cada pregunta constará de cuatro respuestas de las cuales solo será una la correcta. Podrá incluir opcionalmente una fotografía si así lo requiere la pregunta y además estas imágenes podrán incorporar su pie de foto. La duración del examen general de 255 preguntas tendrá una duración aproximada de 3 horas.

4 Desarrollo

En este apartado se especifica en profundidad cada parte del desarrollo del proyecto. Se cuenta con las siguientes secciones:

- Pasos previos a su desarrollo.
- Herramientas necesarias para desarrollar el proyecto.
- Elementos de una aplicación.
- Desarrollo funcional de la aplicación.

4.1 Pasos previos

Los primeros pasos que se han dado antes de comenzar a realizar el proyecto han sido los siguientes:

- Introducción a la programación en Java. Realización de ejercicios simples para asimilar los conceptos básicos.
- Introducción a la programación en Android. Se realizan aplicaciones sencillas para familiarizarse con el entorno de desarrollo.
- Estudio de libros de Java y Android para reforzar la base adquirida en los puntos anteriores.

4.2 Herramientas

Las siguientes herramientas son las que se han utilizado para el desarrollo de la aplicación:

- Ordenador portátil personal.
- Smartphone con sistema operativo Android.
- Entorno de desarrollo Eclipse para programar en Java.
- Entorno de desarrollo Android Studio para realizar el proyecto en Android.
- Componente Android SDK Tools con el conjunto de herramientas de desarrollo y depuración del entorno.
- Emulador de Android AVD Manager.
- Editor gráfico para el tratamiento de imágenes.

4.3 Elementos de una aplicación

A continuación se explicarán las partes principales que tiene una aplicación Android y posteriormente veremos en detalle el desarrollo de cada sección de nuestro proyecto.

4.3.1 Actividad

Se conoce como actividad a cada una de las pantallas o interfaces que se muestran en una aplicación Android. Cada actividad está formada por dos partes, la parte lógica alojada en archivos ".java" donde se implementan las funcionalidades de la aplicación y la parte

gráfica en archivos ".xml" donde se definen todas las componentes gráficas que conformarán la interfaz.

Toda actividad se inicializa con el método "OnCreate()" en el cual se encuentra la llamada a "setContentView" mediante la cual se enlaza la parte lógica y la gráfica de la actividad. Si se necesita utilizar en la parte lógica elementos de la parte gráfica se hará recuperando su identidad con "findViewById()".

La manera en la que se administran las actividades en una aplicación es mediante una pila. Cuando una actividad se inicializa se coloca en la primera posición de la pila siendo la actividad que se está ejecutando en ese momento, dejando a la actividad previa en la posición anterior de la pila. Cuando la actividad en ejecución se destruye es cuando vuelve a aparecer activa la actividad que se encontraba en una posición inferior de la pila.

Cada actividad tiene cuatro estados fundamentales:

- **Activa**: una actividad estará activa o en ejecución si aparece en primer plano de la interfaz, se encontrará en la primera posición de la pila.
- **Pausada**: una actividad se encontrará pausada cuando sea visible pero no se encuentre en primer plano. Esta actividad está completamente viva, pero no se encuentra en la primera posición de la pila, mantiene toda la información de estado para en cualquier momento pasar al estado de activa. La actividad pausada puede ser eliminada en cualquier momento por el sistema por situaciones de poca memoria.
- **Detenida**: una actividad se encuentra detenida cuando hay otra que la oculta por completo. La actividad conserva toda la información de estado y miembros, pero no es visible en el dispositivo, la interfaz está oculta y el sistema eliminará esta actividad con frecuencia en situaciones de escasez de memoria.
- **Destruida**: una actividad se encontrará destruida cuando pasa de estar en el estado de pausa o detenida y es eliminada por el sistema. Si el usuario solicita que se vuelva a mostrar, el sistema tiene que reiniciar esa actividad completamente y restaurarse a su estado previo.

El siguiente diagrama muestra el ciclo de vida de una actividad en Android:

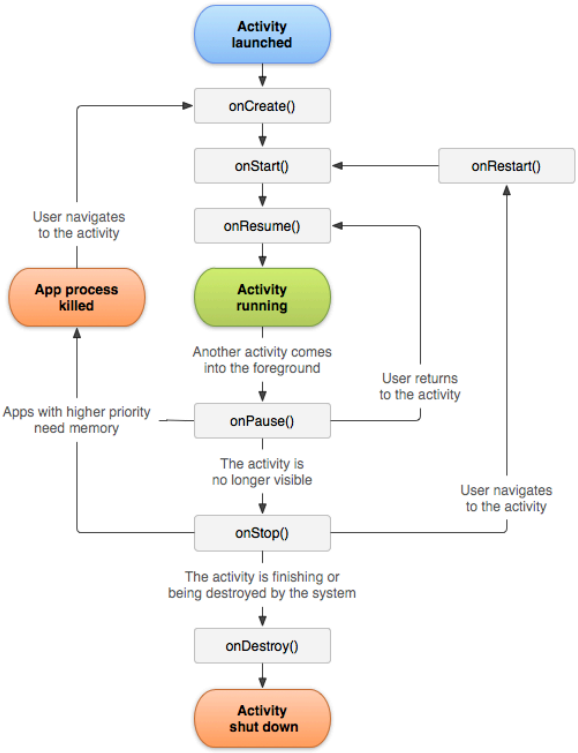

Figura 4-1: Ciclo de vida de una actividad en Android. (Fuente: Android Developers).[5]

Los cambios de estado en el ciclo de vida de una actividad se realizan con una serie de llamadas a métodos:

- **onCreate():** creación de la actividad donde se configuran las vistas, se vinculan datos, etc. Si la actividad ya existía contiene el paquete que contiene el estado anterior.
- **onStart():** se llama a este método para que la actividad se vuelva visible por el usuario.
- **onResume():** en este punto la actividad se encuentra en la parte más alta de la pila y se produce la interacción con el usuario.
- **onPause():** se llama a este método cuando el sistema quiere reanudar una actividad previa, por ejemplo para detener animaciones, para confirmación de cambios no guardados de datos persistentes, etc.
- **onStop():** cuando la actividad se vuelve invisible para el usuario, ya que otra actividad se reanuda y pasa al primer plano.
- **onDestroy():** es la llamada final que destruye la actividad ya sea porque queremos que esa actividad termine o porque el sistema necesita ahorrar espacio en memoria y la destruye.

4.3.2 Archivo manifiesto

Toda aplicación Android cuenta con un archivo en el directorio raíz que aporta toda la información esencial de la aplicación para que el sistema la pueda ejecutar correctamente. Este archivo se conoce como "AndroidManifest.xml". Algunas de las funciones que tiene el archivo manifiesto en la aplicación son las siguientes:

- Contiene el nombre del paquete de Java de la aplicación, utilizado como identificador.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.ivan.multiquiz">
```

- Describe algunos de los componentes de la aplicación como pueden ser las diferentes actividades:

```
<activity android:name=".MenuQuizClass"
    android:label="EETest: TEST GROUPS"
    android:theme="@style/EETestTheme"
    android:screenOrientation="portrait"></activity>
```

  En este caso se observa como en la declaración del manifiesto de una actividad se puede dar información de la clase Java que la ejecuta, el nombre que llevará esa interfaz en su barra superior, el estilo del tema que se utilizará en esa actividad y la orientación en la que se podrá ejecutar la misma, que en este caso será en orientación vertical (portrait).

- Define los procesos que albergan los componentes de la aplicación y enumera bibliotecas con las que se vincula la misma.

- En él se configura la versión mínima del sistema operativo que requiere el dispositivo para poder instalar la aplicación y con qué versión se compila :

```
<uses-sdk
    android:minSdkVersion="15"
    android:targetSdkVersion="27">
```

  Si no se declara la versión del "targetSdkVersion" de compilación el sistema utilizará para ello la versión indicada en el "minSdkVersion". Debido a que Android evoluciona sacando versiones nuevas continuamente, aunque el nivel de la versión API de la plataforma sea superior a la indicada en el "targetSdkVersion", el sistema adaptará su comportamiento para mantener la compatibilidad entre cualquier dispositivo y la aplicación, haciendo que todo funcione de la forma esperada.

4.3.3 Vistas

La clase "View" de Android se utiliza para representar en la interfaz del dispositivo todas sus componentes y para poder manejar los eventos. Es la clase donde se alojan todos los "widgets" como pueden ser los botones, los textos, los recursos gráficos, etc. Los atributos de las vistas se establecen en los archivos de diseño ".xml" del proyecto, aunque esos atributos se pueden ver modificados en los códigos ".java".

Para organizar los elementos de la interfaz dentro de las vistas se utilizan los llamados contenedores, que sirven para facilitar el trabajo a la hora de conseguir un diseño más

óptimo. En este proyecto se han utilizado algunos contenedores como son el "RelativeLayout" o los "ConstraintLayout". Además se ha incorporado el contenedor "ScrollView" que ayuda al desplazamiento hacia arriba o hacia abajo en caso de necesitar interactuar con la pantalla del dispositivo para visualizar contenido que queda fuera de la pantalla.

4.3.4 Carpeta Java.

Es la carpeta donde se alojan todos los archivos ".java" que contienen el código de la lógica de la aplicación

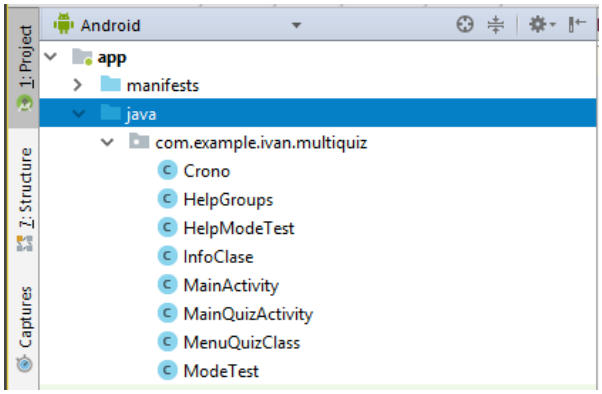

Figura 4-2: Carpeta Java con las clases del proyecto. (Fuente: elaboración propia).

4.3.5 Carpeta Recursos

Es la carpeta donde se alojan todos recursos (resources) y datos externos que se utilizan en la aplicación, como pueden ser estilos, imágenes, iconos, strings, colores, etc. Cada recurso utilizado se debe alojar en su carpeta correspondiente. Algunos de los recursos utilizados en este proyecto son:

- **Drawable:** carpeta donde se alojan todos los recursos gráficos de la aplicación, en este caso se han utilizado imágenes en formato ".png" y además donde se adaptan las imágenes para las diferentes densidades de pantalla.
- **Layout:** carpeta donde se alojan los archivos de diseño ".xml" de la aplicación, en versión para pantallas normales y para pantallas grandes.
- **Mipmap:** aquí se alojan los iconos de la aplicación en las distintas densidades de pantalla y en formato tanto circular como cuadrado para hacer posible la adaptación a la configuración de cualquier dispositivo.
- **Raw:** en esta carpeta se alojan los audios que se utilizan en la aplicación en formato ".mp3".
- **Values:** aquí se encuentran diferentes archivos de configuración como son el "colors.xml" donde se definen colores utilizados en la App en hexadecimal, el archivo "strings.xml" para cadenas de caracteres y el "styles.xml" donde se definen los estilos de los temas de las diferentes pantallas.

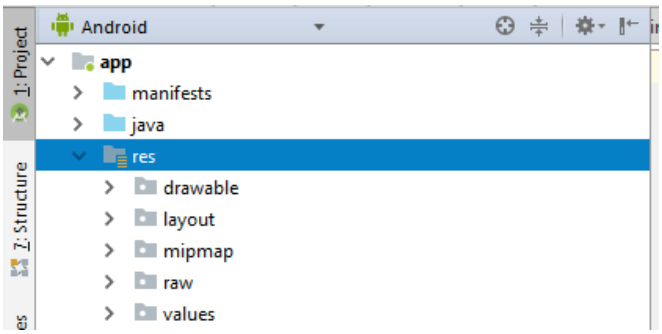

Figura 4-3: Carpeta "resources" con los recursos del proyecto. (Fuente: elaboración propia).

4.3.6 Carpeta Gradle Scripts

En esta carpeta, se aloja el sistema de compilación del proyecto, el cual coge automáticamente los archivos fuente ".java" y ".xml", los combina en un solo archivo comprimido el APK, el cual contiene la aplicación Android.

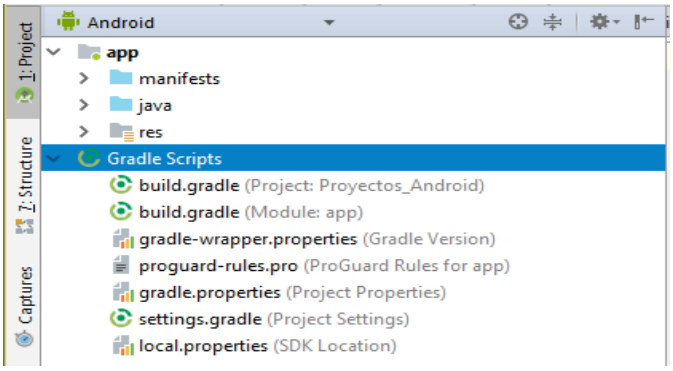

Figura 4-4: Carpeta "Gradle Scripts" con el sistema de compilación de proyecto. (Fuente: elaboración propia).

4.4 Desarrollo funcional de la aplicación

En este apartado se profundiza en detalle sobre las funcionalidades de cada parte de la aplicación de una forma más técnica.

4.4.1 Menús de la aplicación

Esta aplicación tiene tres menús principales, los cuales disponen de diferentes botones, de manera que su diseño proporciona una experiencia de usuario sencilla e intuitiva, y se puede interaccionar e ir accionando las diferentes funcionalidades que proporciona cada botón.

Los menús que encontramos son los siguientes:

- **Menú principal:** es el primer menú que aparece al abrir la aplicación por primera vez. En él encontraremos tres botones; el primero es el botón "Test" el cual llevará a un segundo menú, otro botón "About" con información del programador y el botón "Exit" el cual cerrará la aplicación y se accederá a la pantalla del escritorio de nuestro dispositivo.

Figura 4-5: Menú principal de la aplicación. (Fuente: elaboración propia).

- **Menú de selección del modo test:** en este menú encontramos 4 botones; el primero de todos "Student Test Mode" permitirá realizar el test en modo estudiante, el segundo "Teacher Test Mode" en modo profesor, el tercero es el botón "Help" de ayuda y por último encontramos el botón "Exit" el cual hace que la actividad de esa interfaz acabe y por tanto conducirá a la pantalla anterior.

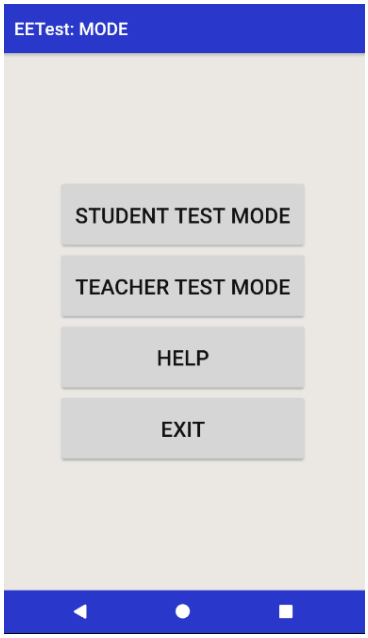

Figura 4-6: Menú de selección de modo de la aplicación. (Fuente: elaboración propia).

- **Menú de selección del grupo de preguntas del test**: en este menú encontramos 8 botones; el primero de todos es el "Full Test" que permitirá realizar el test completo con toda la colección de preguntas, a continuación, encontramos 5 botones que llevarán a realizar un examen tipo test de las diferentes materias. Estos botones son: "FPGA", "EE Basic Concepts", "Asic Design", "EE Culture" y "Logic Design". Por último encontramos el botón "Help" de ayuda y el botón "Exit" que conducirá a la pantalla anterior.

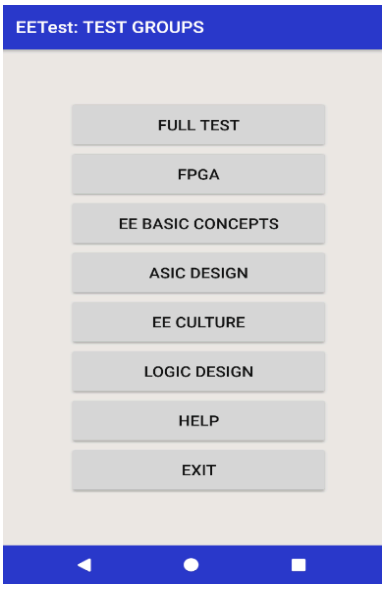

Figura 4-7: Menú de selección del grupo de preguntas de la aplicación. (Fuente: elaboración propia).

4.4.2 About

El botón acerca de (about) que se encuentra en el menú principal conducirá a una actividad que dispone de información de la aplicación, como es el lugar donde se ha realizado el proyecto, el laboratorio donde se ha efectuado el mismo, el programador que ha llevado a cabo su desarrollo o los email de contacto tanto del desarrollador como del tutor del proyecto. Esta interfaz sigue el formato establecido por el Digital System Laboratory (DSLab) para todas sus aplicaciones Android.

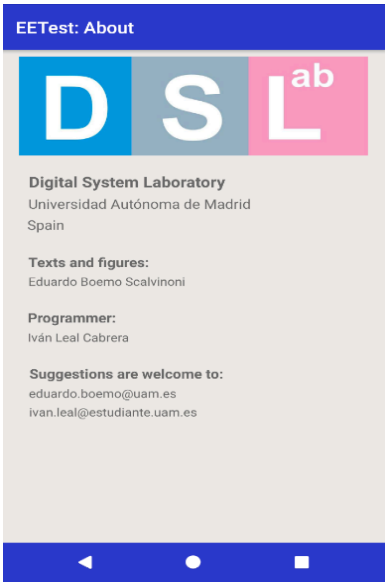

Figura 4-8: Interfaz "About" de la aplicación. (Fuente: elaboración propia).

4.4.3 Modos de realización del test

En la realización de los exámenes tipo test de la aplicación se pueden seguir dos modalidades:

- **Modo estudiante:** en esta modalidad el usuario que realice el examen dispondrá de todo el tiempo que necesite para resolver todas y cada una de las preguntas de las que conste el test en particular de la materia que haya escogido. En este caso, el cliente puede leer con tranquilidad cada una de las preguntas y respuestas y puede dedicar todo el tiempo que sea necesario.

  A esta variante se accede desde el menú de selección de modo del test si se pulsa en el botón "Student Test Mode", pasando directamente a elegir la materia de la se quiera examinar. A continuación irán apareciendo las preguntas sin la presencia de un contador de tiempo.

- **Modo profesor:** en este caso el usuario que elija realizar el examen en éste modo, se enfrentará al test pero dispondrá de un tiempo limitado para contestar a las preguntas. Tendrá que gestionar correctamente el tiempo para intentar obtener la mayor calificación posible.

  Igual que ocurría en el caso anterior, a esta modalidad se accede desde el menú de selección de modo del test pero pulsando en el botón "Teacher Test Mode". Al hacer esto, se mostrará una pantalla nueva de selección de la duración del test, y posteriormente se podrá elegir la materia de la cual se quiere realizar el examen, como también ocurría en el modo estudiante.

  En este caso se mostrará durante el tiempo que se esté realizando el examen, un contador de cuenta atrás en la parte superior de la pantalla.

4.4.4 Interfaz de ayuda

Para conseguir que la navegación por las interfaces resulte lo más clara posible, se han desarrollado dos pantallas de ayuda para el usuario:

- **Ayuda del menú de modos:** se accederá a ésta ayuda si se pulsa en el botón "Help" en el menú de selección de modo. Llevará a una pantalla que explicará en qué consiste el modo estudiante y el modo profesor.
- **Ayuda del menú de selección de grupos:** se accederá a ésta ayuda si se pulsa en el botón "Help" en el menú de selección de grupos donde se escoge la materia de la que se quiere realizar el examen. De esta manera, se accede a una interfaz que explica las diferentes opciones.

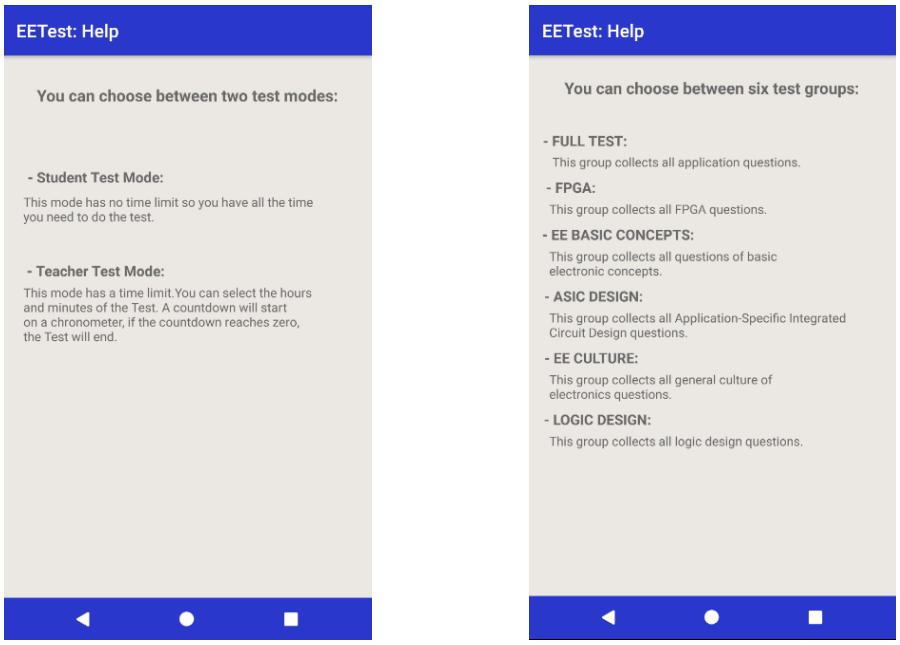

Figura 4-9: Menús de ayuda de la aplicación. A la izquierda ayuda de selección de modos y a la derecha la ayuda de selección de grupos. (Fuente: elaboración propia).

4.4.5 Cronómetro

Antes de comenzar a hacer el test si se ha escogido la modalidad de profesor, la aplicación llevará a una nueva interfaz la cual permitirá elegir cuánto tiempo queremos que dure el examen, posibilita introducir tanto el número de horas como el de minutos.

De este modo se limita la realización del examen, lo cual es muy útil en el ámbito docente, de tal modo que se puede hacer un mismo examen a diferentes alumnos y éstos tendrán el mismo tiempo para contestar a todas las preguntas, lo cual requerirá que hagan una buena administración de tiempos.

La interfaz del cronómetro contará con dos casillas, una primera para introducir las horas en la que se pueden introducir desde 0 a 10 horas y una segunda casilla, la de los minutos, en la que se pueden introducir entre 0 y 59 minutos. Se han escogido estos límites debido a que la aplicación está dimensionada para un número de preguntas específico y no se necesitará en ningún caso realizar un examen de más de diez horas.

Si en este cronómetro se introducen 0 horas y 0 minutos será el equivalente a realizar el examen en modo estudiante, y por tanto durante la realización del mismo se dispone de todo el tiempo necesario para finalizarlo.

En caso de que el usuario intente introducir un número que exceda los límites o si por algún motivo intenta introducir un número negativo, la aplicación automáticamente mostrará un mensaje de error recordando el rango que puede seleccionar y no dejará fijar el tiempo hasta que no introduzca valores correctos.

A continuación se muestra la interfaz del cronómetro:

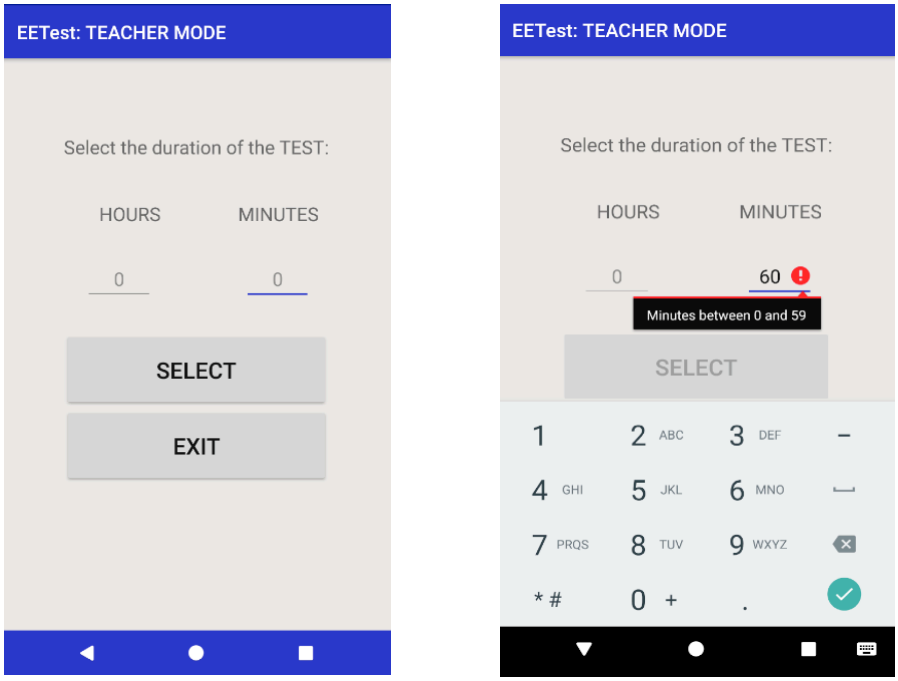

Figura 4-10: Interfaz del cronómetro de la aplicación. (Fuente: elaboración propia).

El cronómetro se ha diseñado para que haga una cuenta atrás desde la hora que se haya introducido en la interfaz de selección de tiempos hasta llegar a cero. En el momento que la cuenta llegue a cero automáticamente el examen habrá finalizado y se mostrarán los resultados obtenidos del mismo.

Para el desarrollo del cronómetro de cuenta atrás se ha utilizado la clase proporcionada por Android llamada "CountDownTimer" como observamos a continuación:

```
timer =new CountDownTimer(tiempoCronometro, 1000){}
```

A esta clase se le pasa el tiempo total que tendrá el test en milisegundos en la variable "tiempoCronometro" y el intervalo de tiempo que utilizará el cronómetro para ir haciendo llamadas a su método principal para ir actualizando el texto mostrado en la pantalla descontando el tiempo. En este caso, el intervalo de tiempo es de 1000 milisegundos lo que equivale a un segundo, lo cual quiere decir que nuestro contador irá descontando cada vez 1 segundo.

Se ha establecido en el diseño de la interfaz que el cronómetro se encontrará en la esquina superior derecha y el formato del cronómetro que se verá en la pantalla tendrá el aspecto siguiente: "Time remaining: horas : minutos : segundos".

Para iniciar la cuenta atrás hay que activar el cronómetro mediante la implementación en el código la instrucción "timer.start()" y para detenerlo se utiliza "timer.cancel()".

4.4.6 Diseño y desarrollo de los test

En este punto se define la manera en la que se realizan los exámenes así como las diferentes partes que conforman la interfaz.

Una vez que el usuario ha escogido la materia de la que quiere examinarse le aparecerá la primera pregunta. Desde esa pregunta podrá ir avanzando a las siguientes con el botón "Next" e ir contestando una a una. En caso de que quiera volver hacia atrás a revisar preguntas anteriores o contestar alguna pregunta que ha dejado sin contestar, el usuario podrá retroceder con el botón "Previous". En la primera pregunta del test el botón "Previous" no aparece ya que no hay preguntas anteriores, y en la última pregunta el botón "Next" se sustituye por otro botón llamado "Finish" que será utilizado para finalizar el test.

La interfaz de la pantalla se divide en diferentes partes. En la esquina superior izquierda se pueden ver una cuenta de las preguntas totales que tiene el test que se está realizando así como el número de la pregunta que se está contestando, es decir, si por ejemplo el test tiene 255 preguntas en total y se está contestando a la pregunta 7 en esa esquina se mostrará "Question: 7 of 255". La esquina superior derecha está reservada para mostrar el cronómetro en el caso de que el usuario haya elegido realizar el test en modo profesor. El formato del cronómetro como se ha contado en la sección anterior será "Time remaining: horas : minutos : segundos".

A continuación se mostrará la pregunta del test. Primero aparece el enunciado de la pregunta, y después podrán verse las cuatro posibles respuestas con las que cuenta esa pregunta. Estas cuatro respuestas se han implementado en el proyecto Android mediante un "RadioGroup" que no es más que un grupo de botones que sirven para conocer que botón ha pulsado el usuario y poder así almacenar las respuestas. Cada una de las respuestas dentro de ese grupo de botones es un botón independiente definido en el código Android como "RadioButton" un tipo de botón que le permite al usuario elegir una de las opciones entre las cuatro del conjunto completo. Se ha decidido utilizar este tipo de botón debido a que las preguntas del test solamente tendrán una respuesta verdadera y esta manera de implementar los botones hace que los botones sean mutuamente excluyentes, lo que significa que si se pulsa sobre una opción, el resto quedarán desmarcadas.

Después, en la pantalla aparecerá de forma opcional la posibilidad de incluir una fotografía que acompañe a la pregunta. Algunas de las preguntas llevarán imagen pero otras solamente serán preguntas de texto que no necesitan una fotografía para completar la pregunta. Técnicamente se ha utilizado para implementar esta funcionalidad el uso de "ImageView" de Android para mostrar los recursos de las imágenes, para cada pregunta se muestra un recurso diferente.

Seguidamente la interfaz dispone de un texto que indicará la fuente de la imagen de la pregunta. Esta fuente podrá ser propia, desconocida o si se conoce la fuente donde se ha recopilado, se indicará en la parte inferior de la imagen.

Por último en la zona inferior de la interfaz se pueden encontrar tres botones; el de pregunta siguiente y pregunta anterior ya comentados, y el botón "End" que podrá utilizar el usuario en cualquier momento para abandonar el test con una pequeña verificación para confirmar que realmente quiere salir del test y no ha pulsado ese botón por error.

A continuación se muestra el diseño de la interfaz de preguntas:

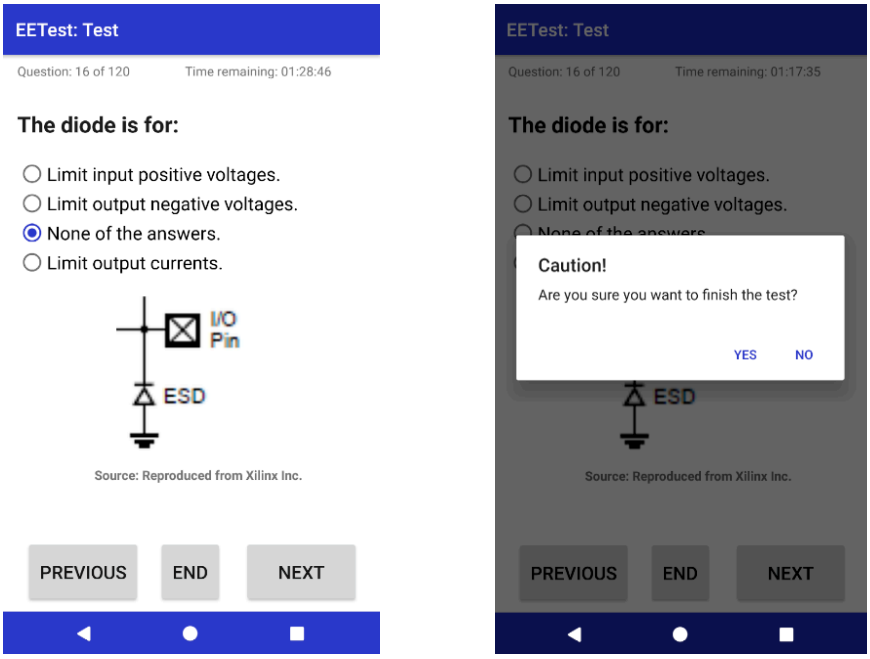

Figura 4-11: A la izquierda interfaz de una pregunta cualquiera, y a la derecha resultado de pulsar botón END. (Fuente: elaboración propia).

4.4.7 Formato preguntas

En cuanto al almacenamiento y manejo de las preguntas de la aplicación se ha establecido que la manera más sencilla, ágil y eficiente de tratar las preguntas es declarando todas las preguntas en arrays de strings de texto plano alojándolas en el archivo "strings.xml" de la aplicación.

Los arrays de preguntas se estructuran de la siguiente manera:

```
<array name="ArrayPreguntas">

    <item>Enunciado pregunta 1;respuesta 1;respuesta 2;*respuesta
3;respuesta 4;Nombre imagen;pie de foto1</item>
    <item>Enunciado pregunta 2;respuesta 1;*respuesta 2;respuesta
3;respuesta 4;Nombre imagen2;pie de foto2</item>
    <item>Enunciado pregunta 3;*respuesta 1;respuesta 2;*respuesta
3;respuesta 4;Nombre imagen3;pie de foto3</item>

</array>
```

En este ejemplo existe un array que en este caso se llama ArrayPreguntas. Este array tiene tres posiciones; en la primera posición del array en la posición 'ArrayPreguntas[0]' tendremos la primera pregunta del test completo que está definida entre las etiquetas "ítem".

Cada pregunta del array consta de diferentes partes, las cuales están separadas por el signo ';' el cual servirá de gran ayuda ya que este símbolo permite ir leyendo las diferentes partes que contiene la pregunta. Lo primero que aparece en la primera pregunta es el texto que tendrá el enunciado de la pregunta, que en este caso sería "Enunciado pregunta 1". A continuación separado todo por el símbolo ';' tenemos las 4 posibles respuestas que incluirá esta pregunta, marcando la respuesta verdadera con el símbolo asterisco '*'.

Después se incluirá a la pregunta el nombre exacto de la fotografía que llevará ésta pregunta, nombre que tendrá que ser el mismo que tenga la imagen que se guarde en la carpeta Drawable de la aplicación, donde se encontrarán todas las imágenes. Por último se incluirá el texto del pie de foto que llevarán las imágenes de cada pregunta que será en el que aparezca la fuente de donde se ha obtenido esa imagen.

De este modo lo que se consigue es que en un simple array de texto se obtenga toda la información necesaria para ir mostrando todas las preguntas de un test completo. Así, solo se necesita tener un archivo de diseño ".xml" con el diseño de la estructura de las preguntas que se mostrarán en la interfaz y después en la implementación de las clases ".java" donde simplemente se irán actualizando los diferentes textos e imágenes de la interfaz.

Cada una de las materias en las que el usuario puede examinarse cuenta con diferentes arrays de preguntas las cuales contienen las mismas preguntas pero ordenadas de manera diferente y con las respuestas también en ordenes distintos, de este modo se dispone de modelos diferentes del mismo examen y hace que la aplicación de manera aleatoria escoja el modelo de examen que va a realizar el usuario. De este modo se consigue que un examen de una determinada materia sea diferente cada vez que el usuario vaya a hacerlo para dificultar las copias o para impedir el aprendizaje del orden de las preguntas y que se obtengan mejores resultados.

Para añadir, eliminar o corregir preguntas. Sirve con modificar el archivo 'strings.xml' con los cambios nuevos y la aplicación quedará actualizada con las nuevas preguntas.

4.4.8 Interfaz de resultados

Una vez finaliza el test ya sea porque el usuario ha acabado de contestar a las preguntas y ha decidido que el examen concluya o debido a que el tiempo se le ha acabado y automáticamente se haya puesto fin a su examen, la interfaz mostrará los resultados obtenidos a través de un cuadro de diálogo situado en primer plano.

La primera información que se mostrará será el número de respuestas acertadas, el número de respuestas incorrectas y las preguntas no contestadas. Con esta información se calcula la nota final del test teniendo en cuenta las respuestas acertadas y las incorrectas, el número de posibles respuestas que tiene cada pregunta que en este caso son 4, y se muestra una nota normalizada entre 0 y 10 y se calculará con una precisión de dos decimales.

La fórmula que se ha utilizado para calcular la nota final es la siguiente: [6]

$$Nota = \left(\text{correctas} - \frac{\text{incorrectas}}{número\ posibles\ respuestas - 1}\right) x \left(\frac{10}{número\ preguntas\ totales}\right)$$

Por último esta interfaz contará con tres botones que permitirán acceder a diferentes funcionalidades. Uno de ellos, el botón de "correction" llevará a la corrección del examen, pantalla donde se mostrarán las preguntas acertadas y las falladas. Después, se puede acceder al botón "start over" el cual hará comenzar el mismo examen que se acaba de realizar para poder repetirlo nuevamente. Por último, encontramos el botón "finish" quien pondrá fin al test y se saldrá de la actividad del examen.

A continuación se muestra un ejemplo de prueba de uno de los test realizados y los resultados obtenidos:

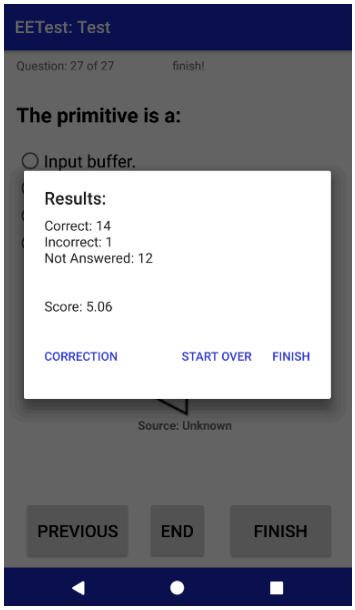

Figura 4-12: Interfaz de resultados de la aplicación. (Fuente: elaboración propia).

Además de todo ello se han introducido sonidos en la aplicación y según la calificación que el usuario obtenga se reproducirá un sonido u otro. En caso de que la puntuación final sea menor a un 5 en el momento de mostrar los resultados obtenidos, la aplicación reproducirá un sonido en el dispositivo de un "Gong". En caso de obtener una calificación igual o superior a 5 la aplicación reproducirá un aplauso ya que este usuario habrá obtenido un aprobado en el test.

Para introducir sonidos en la aplicación se han metido los archivos de los audios en formato ".mp3" en una nueva carpeta "raw" dentro de los recursos "res" de la aplicación. Para reproducir un sonido en Android se ha utilizado la funcionalidad MediaPlayer creando un sonido con el audio de la carpeta "raw" y después reproduciéndolo activando ese sonido. Se ha implementado de la siguiente manera:

```
MediaPlayer sonidoFinAprobado;
sonidoFinAprobado=MediaPlayer.create(this,R.raw.aplauso);
sonidoFinAprobado.start();
```

4.4.9 Autocorrección

Una vez que el usuario ha terminado de contestar a todas las preguntas que ha decidido responder, la pantalla de resultados cuenta con la opción de corregir el examen, lo cual permitirá saber qué preguntas se han acertado, qué preguntas se han fallado y en caso de dejar sin contestar alguna también se muestra la respuesta correcta para ofrecer al usuario así una corrección completa y útil para su aprendizaje.

Durante la realización del test, las respuestas del usuario se van guardando en un vector, se van almacenando los botones en los que se ha hecho el "check" para una vez acabe el examen se pueda comparar la respuesta correcta con el botón pulsado por el usuario.

En caso de que el usuario acierte una pregunta, se mostrará en la pantalla el botón de la respuesta correcta presionado, el cual había sido pulsado por el usuario y aparecerá esa

respuesta con fondo verde. En caso de fallar la pregunta, se mostrará el botón elegido por el usuario pulsado con el fondo rojo y la respuesta correcta en verde. Por último en las preguntas no contestadas simplemente aparecerá el grupo de botones con todos los botones deshabilitados pero en la respuesta correcta se mostrará el fondo de color verde.

La manera en la que se ha implementado esta funcionalidad ha sido mediante el desarrollo de un algoritmo que va mostrando cada pregunta una a una, y como previamente se han almacenado los valores escogidos por el usuario, lo que se hace en ese algoritmo es ir comparando la opción del usuario y la respuesta correcta y se van modificando los colores de fondo de las respuestas para que el usuario vea claramente cuales ha acertado y cuales ha fallado o cuales no ha contestado.

A continuación podemos ver la corrección de un examen una vez hemos concluido su realización y hemos accedido a éste modo:

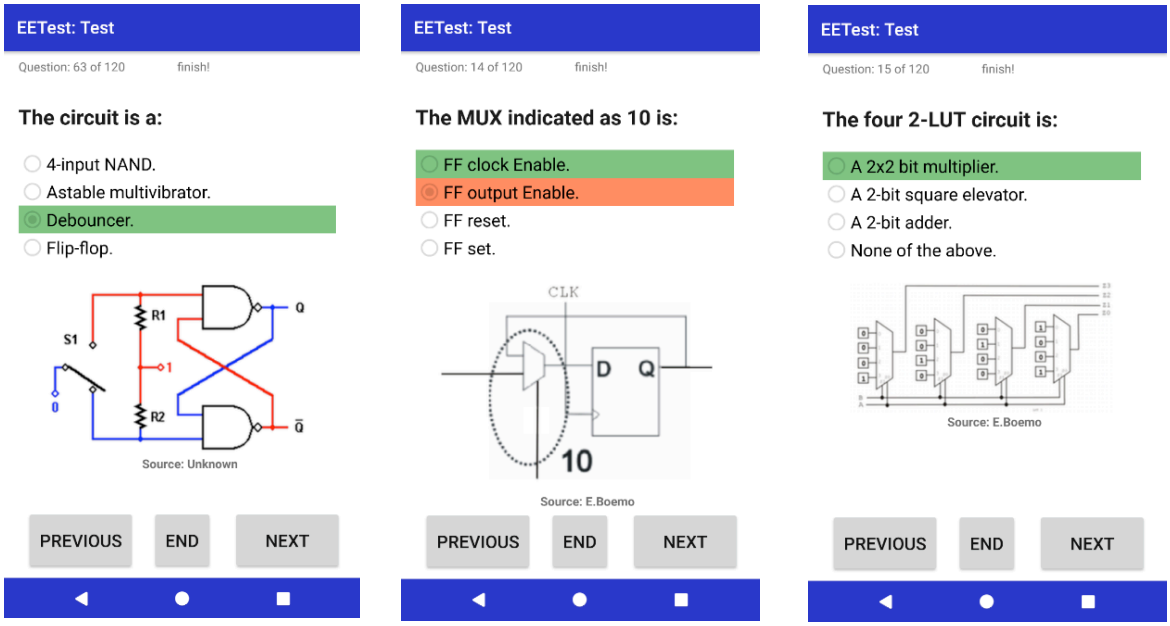

Figura 4-13: Interfaz de corrección: a la izquierda respuesta correcta, en el centro respuesta incorrecta y a la derecha respuesta sin contestar. (Fuente: elaboración propia).

4.4.10 Diagrama y descripción de las clases Java

Diagrama de las clases utilizadas:

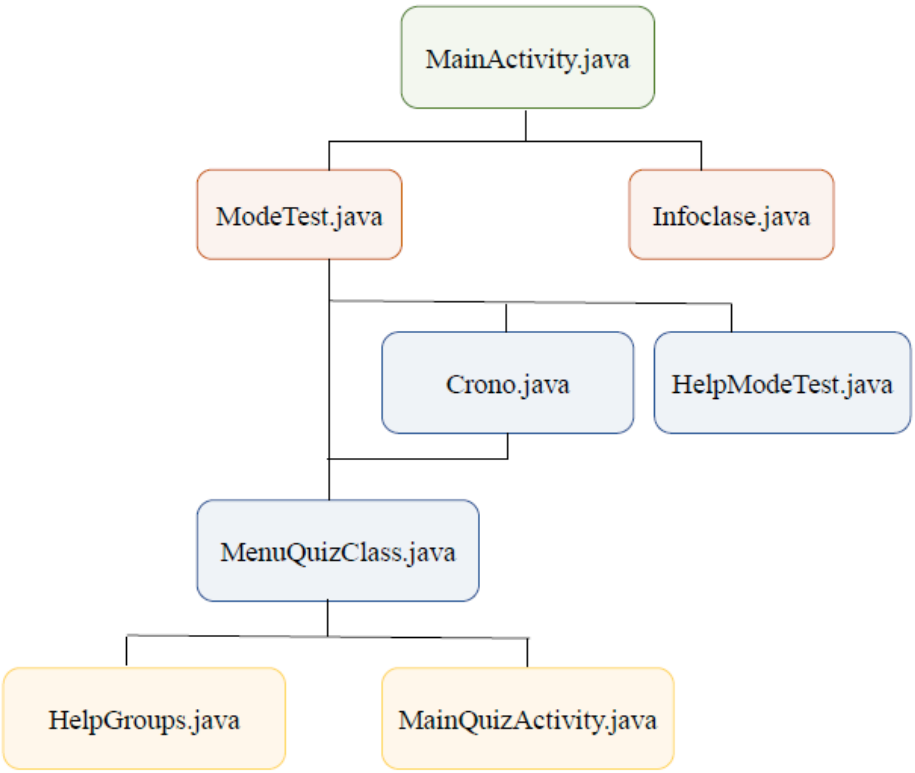

Figura 4-14: Diagrama de clases de la aplicación. (Fuente: elaboración propia).

Descripción de las clases Java utilizadas:

Clase	Descripción
MainActivity	Esta clase es la que muestra el menú inicial de la aplicación.
InfoClase	Es la clase que muestra la información del desarrollador y de la aplicación.
ModeTest	Sirve para elegir el modo de realización del test, ya sea el modo estudiante o el modo profesor.
HelpModeTest	Esta clase informa al usuario de los diferentes modos de realizar el examen.
Crono	Se utiliza para la elección del tiempo de duración del examen si elegimos el modo profesor.
MenuQuizClass	Clase que sirve para la selección de la materia en la que el usuario se va a examinar.
HelpGroups	Clase de ayuda que informa al usuario de los diferentes grupos de preguntas en los que se puede examinar.
MainQuizActivity	Esta es la clase más importante de la aplicación en la que se desarrolla tanto la realización del examen tipo test como su corrección.

Tabla 4-1: clases Java. (Tabla de elaboración propia).

4.4.11 Métodos principales

Descripción de los métodos principales utilizados:

Método	Descripción	Actividades
onCreate()	Método que se llama cuando se inicia una actividad para su creación.	Todas
ejecutar_info()	Se utiliza para mostrar la información del desarrollador.	MainActivity.java
ejecutar_quiz()	Sirve para pasar al menú de selección del modo de realización del test.	MainActivity.java
salirApp()	Se utiliza para salir de la actividad en la que nos encontramos poniéndola fin.	Todas
studentQuiz()	Método utilizado para ejecutar el modo estudiante y pasar al menú de selección de materia de examen.	ModeTest.java
teacherQuiz()	Este método ejecuta el modo profesor y pasa al selector de duración del examen.	ModeTest.java
helpMode()	Se utiliza para mostrar la ayuda con la explicación de los diferentes modos de realizar el test.	ModeTest.java
select()	Método que ejecuta la selección del tiempo del cronómetro de la aplicación.	Crono.java
completeQuiz(), fpgaQuiz(), basicConcepsQuiz(), asicDesignQuiz(), cultureQuiz(), logicDesignQuiz()	Estos métodos se utilizan para ejecutar el examen tipo test de la aplicación en una determinada materia en específico. Hay 6 materias diferentes de preguntas por lo que tenemos 6 métodos diferentes que ejecutan esos exámenes.	MenuQuizClass.java
helpQuiz()	Se utiliza para mostrar la ayuda de las diferentes materias en las que el usuario se puede examinar.	MenuQuizClass.java
onSaveInstanceState()	Método que guarda la información de las variables para recuperarlas posteriormente.	MainQuizActivity.java
startOver()	Sirve para comenzar de nuevo un test una vez lo hemos finalizado, para poder repetirlo.	MainQuizActivity.java
contador()	Método que crea el cronómetro de cuenta atrás de la aplicación.	MainQuizActivity.java
checkResults()	Sirve para calcular el resultado del test, la nota obtenida y los aciertos y fallos que el usuario ha cometido.	MainQuizActivity.java
checkOut()	Método que se utiliza para mostrar al usuario un cuadro de diálogo para advertir de que está finalizando el test.	MainQuizActivity.java
checkAnswer()	Se utiliza para saber en qué botón habíamos pulsado en la realización del test.	MainQuizActivity.java
showQuestion()	Método que va mostrando al usuario cada pregunta del test y le permite ir seleccionando sus respuestas.	MainQuizActivity.java
showCorrection()	Éste es el método de corrección del test.	MainQuizActivity.java

Tabla 4-2: Métodos. (Tabla de elaboración propia).

4.4.12 Detalles del proyecto

Para la realización de este proyecto se comenzó con el estudio de Android y Java desde Julio de 2017, mediante la realización de tutoriales online y estudio de documentación. De media se han dedicado desde entonces unas 15 horas semanales, lo que hacen un total de 660 horas de trabajo.

Se han implementado para el desarrollo del mismo en torno a las 7000 líneas de código.

La aplicación está preparada para albergar 255 preguntas tipo test con una duración aproximada de 3 horas.

4.4.13 Dificultades y soluciones

Durante el proceso de aprendizaje y desarrollo de la aplicación se han encontrado diferentes problemáticas que finalmente se han conseguido solucionar.

En un primer momento, los conocimientos adquiridos en programación Android así como en programación Java, eran bastante escasos, por lo que tuve que aprender mediante la realización de diferentes tutoriales y a través del estudio de documentación de ambos lenguajes como método de preparación previo antes de comenzar a desarrollar el proyecto.

Otras dificultades encontradas, han sido las continuas actualizaciones de Android Studio prácticamente cada mes. Debido a ello, al descargar ciertas actualizaciones, no ha sido posible el funcionamiento de algunas partes de la aplicación, por lo que se tuvo que proceder a la restauración de los fallos obtenidos tras la actualización. Android Studio va evolucionando continuamente surgiendo de esta manera versiones actuales, que en este caso, han provocado las dificultades mencionadas anteriormente, donde procedimientos que funcionaban con una versión, tras la actualización, eran motivo de fallo o problema. Todo ello se ha estado solucionando buscando diferentes maneras de implementarlo y cambiarlo en el proyecto para su correcto funcionamiento.

El interrogante de cómo almacenar una colección de aproximadamente 255 preguntas mediante una base de datos en un servidor para su uso en la aplicación, también ha sido un motivo de limitación en este proyecto, debido a que si el dispositivo que ejecutaba la aplicación perdía la conexión se interrumpiría el desarrollo del test. Por tanto, esa opción fue descartada. Posteriormente se pensó en la opción de crear una base de datos albergada en la misma aplicación Android, pero debido al gran número de preguntas y a la dificultad para manejar tal cantidad de las mismas durante el desarrollo de la aplicación, se consideró como otra limitación para este proyecto, decidiendo como solución a todo ello, la introducción de las preguntas como texto plano en un array en el archivo “strings.xml”, desarrollando, de esta manera, la lógica de aplicación tratando ese archivo de texto de una manera más sencilla. De este modo, en todo momento se mantienen todas las preguntas disponibles en un array y se identifican fácilmente qué respuestas son verdaderas o falsas o qué foto corresponde a cada pregunta, entre otros.

Otra dificultad encontrada durante el desarrollo de la aplicación fue la transformación del tiempo en el cronómetro, ya que se utilizaba una función de Android que transformaba las horas y minutos a milisegundos y cuando se realizaban las diferentes pruebas en el emulador de Android, todo funcionaba de forma correcta, hasta que las pruebas se realizaron en un dispositivo real donde se vio que los tiempos en el dispositivo no se ajustaban correctamente. La solución que se encontró ante esta limitación fue calcular de

manera manual las operaciones para transformar las horas y minutos que introduce el usuario a milisegundos con las fórmulas correspondientes.

La dificultad más costosa, sin duda, ha sido el desarrollo del algoritmo completo que se encarga de la lectura pregunta por pregunta, a la vez que la va mostrando en la interfaz y permite al usuario introducir la opción elegida, almacenando las respuestas que va marcando para poder volver atrás y modificarlas si quisiera. Este algoritmo, además, almacena las respuestas en un vector y al final de la realización del test tiene la posibilidad de corregir el examen. Este otro algoritmo de corrección también es uno de los que han supuesto más trabajo a la hora de desarrollarlo, ya que se pretendía que la interfaz fuera lo más atractiva posible para el usuario. Finalmente se consiguió poco a poco ir desarrollando cada una de las partes del algoritmo hasta que al finalmente todo funcionaba según lo previsto.

El cronómetro de cuenta atrás también ha tenido su complicación ya que aparte de transformar las horas y minutos que el usuario introduce en una de las pantallas, se ha tenido que poner un rango determinado para que el usuario no pueda meter un tiempo de duración del examen “sin sentido”, como por ejemplo el introducir un valor negativo o el meter en la casilla de los minutos más de 59 minutos. Se ha conseguido solucionar esta dificultad mediante métodos que están a la escucha leyendo continuamente los valores que se introducen en las casillas del contador. En caso de que el valor introducido sea erróneo en la interfaz aparece un mensaje de error, mostrando el rango de tiempo permitido y además se deshabilitan los botones de selección para que el usuario no pueda continuar si no modifica estos valores.

5 Integración, pruebas y resultados

5.1 Subida a Google Play Store

Una vez se ha concluido el desarrollo de la aplicación y se ha verificado que todo funciona correctamente se procederá a subir la aplicación generando su fichero APK y llevando a cabo su publicación en la plataforma Google Play Store.

5.2 Generación de APK

Un fichero APK (Application Package File) de Android es el que empaqueta la aplicación en un archivo utilizado por el sistema operativo de Android para la distribución e instalación de esa misma aplicación. Al igual que ocurre con los sistemas operativos Windows que utilizan un archivo ".exe" para instalar su software, en Android ocurre de la misma manera pero estos archivos de ejecución son los ".apk".

Para generar el APK de esta aplicación se abrirá el proyecto con el programa Android Studio y una vez ahí se accederá a la opción "Build" y después se seleccionará la opción "Generate Signed APK" generando así el APK firmado.

La plataforma de Google Play Store no dejará publicar ninguna aplicación que no haya sido firmada con su certificado digital, garantizando de esta manera la seguridad de esta aplicación y gracias al conocimiento propio de la firma de la aplicación, serán los autores los únicos autorizados a modificar esa aplicación en el futuro. Para firmar la aplicación se creará un Key Store que servirá para futuras modificaciones y subidas de nuevas versiones de la aplicación. Para ello se solicitarán una clave y una serie de datos personales.

Una vez realizado el seguimiento de estos pasos, se dispondrá de un archivo APK firmado que ya se puede descargar en el equipo personal y subirlo a la plataforma de Google Play Store.

5.3 Cuenta de desarrollador

Para subir la aplicación a la plataforma hay que contar con una cuenta de desarrollador de Android la cual tiene un precio de 25$. En este caso se cuenta con una cuenta de desarrollador del DSLab de la universidad donde se ha desarrollado este proyecto, por lo que no habrá que darse de alta en la plataforma y se podrá subir la presente aplicación desde esa cuenta.

5.4 Subida a Google Play Store

Lo primero que se debe hacer es iniciar sesión en la cuenta de desarrollador y acceder a la consola para seguir los pasos de publicación de una nueva aplicación.

- Se hará click sobre "Add New Application" y se meterán los datos de la aplicación que piden; el nombre de la App, el idioma y además se adjuntará el archivo APK con la aplicación empaquetada.
- Se especificará la versión de la aplicación que se va a publicar ya sea alfa, beta o en modo producción.

- Después desde la pestaña de “Store Listening” se introducirá una serie de datos clave de la aplicación como es una descripción completa de la aplicación, un texto promocional, el icono, algunos pantallazos de la misma, la privacidad, la categoría o algunas palabras clave que utilizarán los usuarios Android para la búsqueda de la aplicación desarrollada en este proyecto.
- Por último se accederá a la pestaña “Pricing & Distribution” y ahí se indicará el precio de esta aplicación que será gratuita y el número de países en el que se distribuirá la misma, donde en este caso se escogerá el máximo posible.

Por último si se desactiva la opción de borrador la aplicación quedará publicada y los usuarios comenzarán a descargarla desde Google Play Store.

5.5 Pruebas y resultados

Para verificar que la aplicación funciona correctamente además de comprobar que desde el emulador de Android Studio todo se ejecuta sin errores, se ha instalado el APK con la aplicación en diferentes dispositivos, tanto smartphone como tabletas, y se ha comprobado que todos los botones y funcionalidades de las diferentes actividades se ejecutan sin ningún problema.

Además se comprueba que la aplicación tiene un diseño adecuado tanto para pantallas pequeñas como para pantallas grandes de tabletas.

6 Conclusiones y trabajo futuro

6.1 Conclusiones

El objetivo principal de este proyecto era el desarrollo de una aplicación Android para la autoevaluación de conocimientos en electrónica digital, en la cual se pudieran ingresar una lista de preguntas y respuestas en formato estándar y generar un examen tipo test de respuesta múltiple con capacidad de autocorrección.

Se puede decir que se han cumplido todos los objetivos satisfactoriamente, pues el resultado del proyecto incluye todas estas funcionalidades cumpliendo todos los requisitos de diseño.

Se ha desarrollado la aplicación para que se pueda utilizar en dispositivos con tamaños de pantalla diferentes como pueden ser smartphones y tabletas.

La aplicación cuenta con las siguientes funcionalidades:

- Diferentes modos de realización de exámenes.
- Posibilidad de incorporar cronómetro para determinar la duración de la prueba.
- Seis grupos de preguntas de diferentes materias de electrónica digital.
- Capacidad de autocorrección.
- Posibilidad de incorporar imágenes.
- Interfaces de ayuda para el usuario.

Desde el punto de vista educativo se han adquirido conocimientos en las siguientes áreas:

- Programación Java.
- Programación XML.
- Programación de aplicaciones Android.
- Manejo del entorno de desarrollo Android Studio.
- Manejo del emulador de Android AVD Manager.
- Manejo de Herramientas de diseño gráfico.

6.2 Trabajo futuro

Debido a la gran cantidad de funcionalidades que ofrece la plataforma Android en el desarrollo de aplicaciones, se han pensado una serie de avances que en un futuro pueden mejorar la aplicación desarrollada en éste proyecto, para que así se pueda llegar a un mayor número de usuarios.

Algunas propuestas de mejoras futuras son las siguientes:

- Idioma: resulta interesante ampliar el número de idiomas en los que la aplicación se puede ejecutar aumentando así el número de descargas y el número de usuarios que alcanzaría la aplicación.

- Ampliación del número de preguntas: sería interesante en el futuro ampliar las cantidades de preguntas de la aplicación así como aumentar los grupos de preguntas de electrónica en los que el usuario puede examinarse con el fin de hacer la aplicación aún más completa.
- Modificar la forma de almacenamiento de las preguntas incluyendo una base de datos con carga automatizada de preguntas.
- Adaptación de la aplicación a otros sistemas operativos como iOS.
- Mejoras de optimización de la batería.
- Mejorar el diseño de la interfaz.
- Desarrollar una aplicación con las soluciones detalladas a los ejercicios de esta primera aplicación, esta vez sacándola al mercado en versión de pago.

Referencias

[1] http://gs.statcounter.com/os-market-share/mobile/worldwide

[2] https://developer.android.com/about/dashboards/?hl=es

[3] https://developer.android.com/guide/practices/screens_support?hl=es#DeclaringTabletLayouts

[4] https://www.internetworldstats.com/stats7.htm

[5] https://developer.android.com/guide/components/activities?hl=es-419

[6] https://elpais.com/elpais/2017/02/22/el_aleph/1487760836_225306.html

[7] https://developer.android.com/

[8] http://stackoverflow.com/

[9] https://play.google.com/store

[10] B. Phillips, C. Stewart, B. Hardy, K. Mariscano. "Programación con Android", Edición 2016.

[11] J.R. Lequerica. "Desarrollo de aplicaciones para Android", Edición 2017.

[12] R.M. Miguel, "Java 9 (Guías Prácticas)", Edición 2017.

[13] H. Schildt, "Java: A Beginner's Guide", 6th Edition.

[14] J. Horton, "Android Programming for Beginners", 2015 Edition.

[15] https://freesound.org/

Glosario

API	Application Programming Interface
APP	Application
SO	Sistema Operativo
APK	Android Application Package
DSLab	Digital System Laboratory
SDK	Software Development Kit
FPGA	Field-Programmable Gate Array
ASIC	Application-Specific Integrated Circuit
AVD	Android Virtual Device
XML	Extensible Markup Language
Dpi	Dots Per Inch
Ldpi	Low Resolution Image
Mdpi	Medium Resolution Image
Hdpi	High Resolution Image
Xhdpi	Extra High Resolution Image
Xxhdpi	Extra Extra High Resolution Image
UAM	Universidad Autónoma de Madrid