UNIVERSIDAD MIGUEL HERNÁNDEZ DE ELCHE

ESCUELA POLITÉCNICA SUPERIOR DE ELCHE

GRADO EN INGENIERÍA MECÁNICA

"DESARROLLO DE UNA HERRAMIENTA MÓVIL PARA LA AUTOEVALUACIÓN DE ASIGNATURAS DE INGENIERÍA MECÁNICA"

TRABAJO FIN DE GRADO

Junio - 2018

AUTOR: Ángel Frutos Valero

DIRECTOR: César Fernández Peris

ÍNDICE

TABLA DE ILUSTRACIONES

1 INTRODUCCIÓN

En el presente documento *Desarrollo de una herramienta móvil para la autoevaluación de asignaturas de Ingeniería Mecánica* se describe el proyecto realizado por el alumno Ángel Frutos Valero perteneciente al Grado de Ingeniería Mecánica en la Universidad Miguel Hernández de Elche. En este proyecto participa César Fernández Peris, perteneciente al área de Ingeniería de Sistemas y Automática, como tutor del mismo.

Dicho proyecto consiste en el desarrollo de una aplicación Android en dos entornos de programación distintos, *Android Studio* y *MIT App Inventor 2*, en la cual se mostrará preguntas relacionadas con las asignaturas de Ingeniería Mecánica y se evaluará al alumno en función de sus respuestas.

1.1 MOTIVACIÓN Y OBJETIVOS DEL PROYECTO

La realización de este Trabajo de Fin de Grado viene motivada por mis ganas de adquirir y ampliar conocimientos dentro de mi carrera universitaria y poder aplicarlos en un futuro dentro de mi vida profesional.

Teniendo en cuenta que la programación no es un campo en el que se profundice mucho en Ingeniería Mecánica me pareció interesante desarrollar una aplicación para dispositivos *Android*, para mejorar mis capacidades dentro de la programación, después de la charla que realizó César sobre sus propuestas para Trabajos de Fin de Grado. Una vez que había contactado con él lleve a cabo un curso, de un mes de duración, dentro de la universidad sobre desarrollo de aplicaciones Android en el cuál César impartía la parte de programación en *Android Studio* y María Asunción Vicente Ripoll la parte de *MIT App Inventor 2*.

Además de mejorar mis habilidades de programación el objetivo que tengo con este trabajo es comparar la programación en dos entornos distintos para el desarrollo de una misma aplicación.

1.2 SISTEMA OPERATIVO ANDROID

1.2.1 HISTORIA

Android es un sistema operativo para móviles basado en *Linux* el cual fue fundado en 2003 en Palo Alto, California. En el año 2005 fue comprado por *Google* pero no fue hasta el año 2007 cuando se anunció la primera versión del sistema operativo: *Android 1.0 Apple Pie*. Justo antes ese mismo año se creó

la *Open Handset Alliance*, un conglomerado de fabricantes y desarrolladores de hardware, software y operadores de servicio. Además *Google* liberó la mayoría del código de Android bajo la licencia *Apache*, una licencia libre y de código abierto.

Aunque los inicios fueran un poco lentos, debido a que se lanzó antes el sistema operativo que el primer móvil, rápidamente se ha colocado como el sistema operativo de móviles más vendido del mundo, situación que se alcanzó en el último trimestre de 2010.

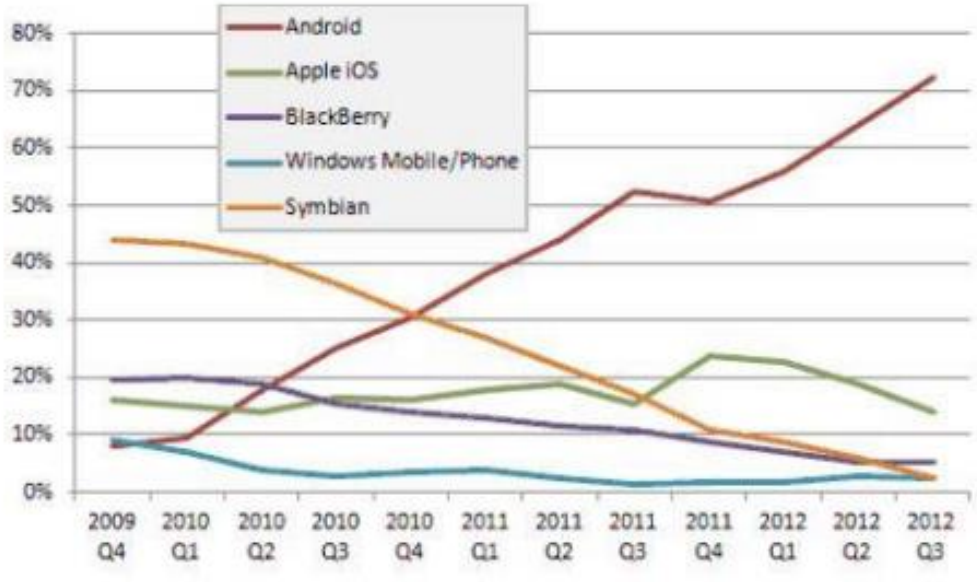


Ilustración 1. Crecimiento del sistema operativo Android.

.

En febrero de 2011 se anunció la versión 3.0 de Android optimizada para tabletas en lugar de teléfonos móviles. Por tanto, Android ha transcendido de los teléfonos móviles a dispositivos más grandes.

1.2.2 ACTUALIDAD

En la actualidad el sistema operativo *Android* sigue siendo el más popular y utilizado por los consumidores de smartphones en todo el mundo cómo podemos ver en la siguiente gráfica elaborada por *Kantar WorldPanel*:

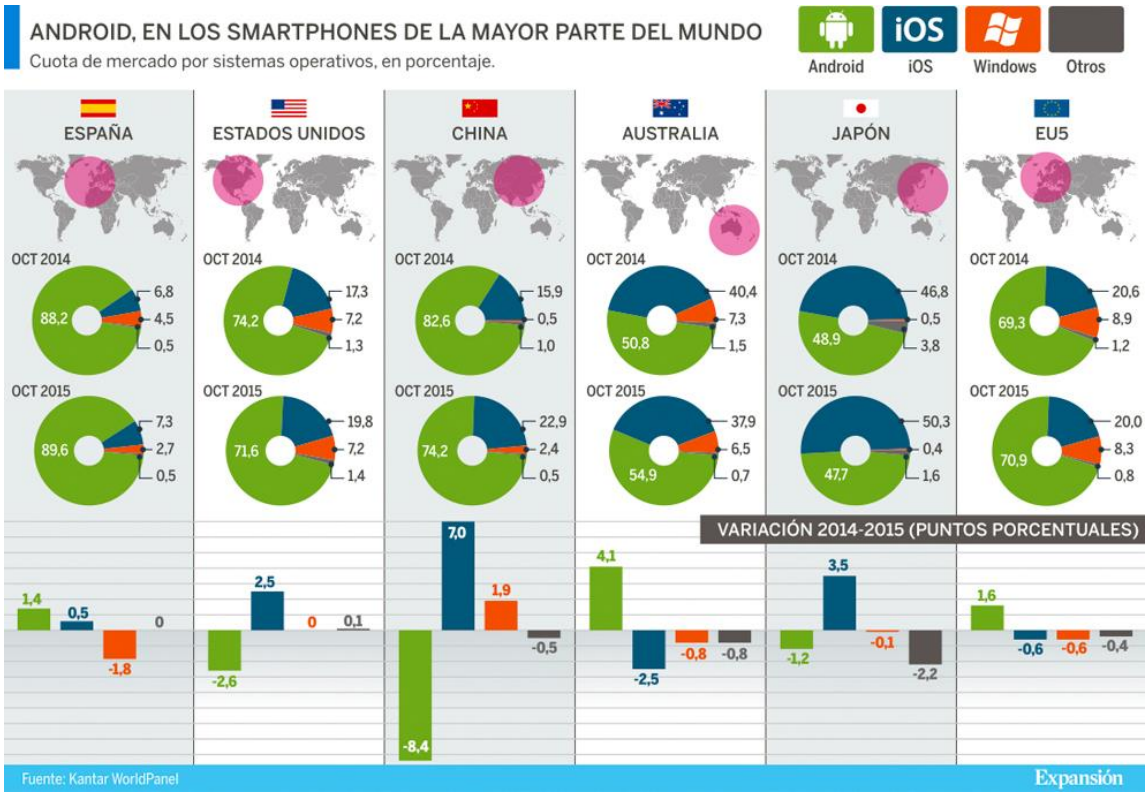


Ilustración 2. Cuota de mercado por sistemas operativos.

Además se prevé que siga siéndolo durante los próximos años. Esta popularidad se debe a su orientación multiplataforma ya que hace unos años un sistema operativo se asociaba a un único dispositivo, además es el más accesible y libre ya que no hay que pagar para programar ni para incluirlo en el terminal lo que lo hace muy popular entre fabricantes y desarrolladores.

Tan grande es el crecimiento de *Android* que incluso ya no compite solo con sus rivales directos dentro de los smartphones sino que está compitiendo con *Windows* como el sistema operativo más utilizado para navegar por internet como podemos ver en la siguiente gráfica de *statista*:

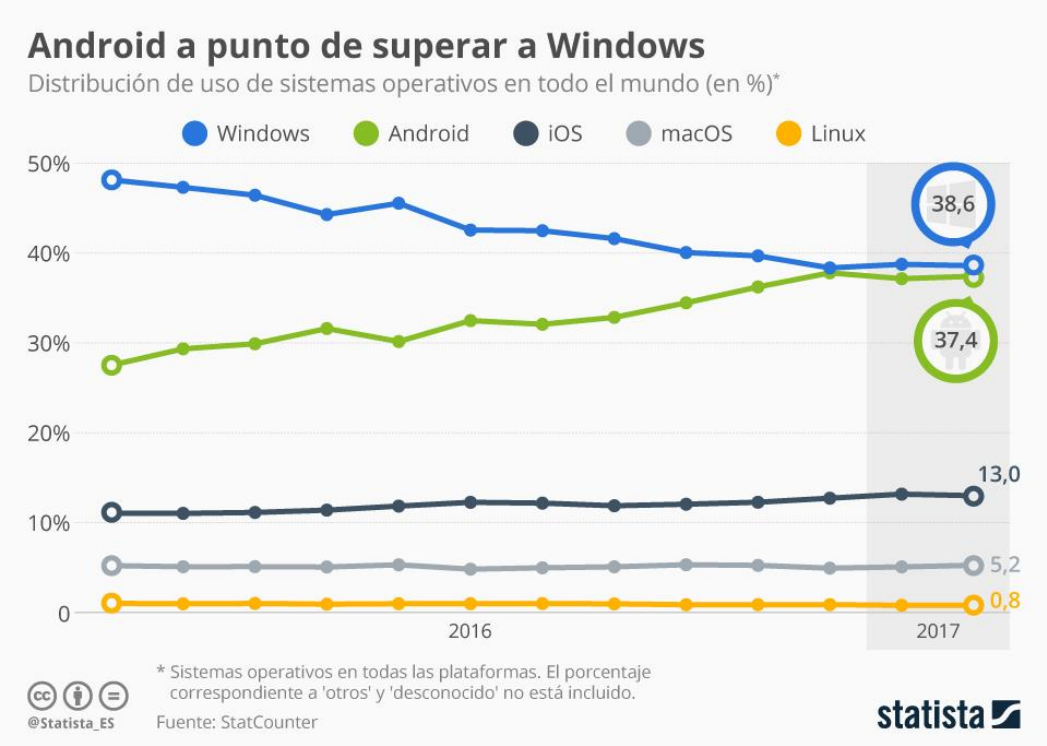


Ilustración 3. Sistemas operativos en todas las plataformas.

1.2.3 VERSIONES SISTEMA OPERATIVO

Android no ha parado de evolucionar desde el lanzamiento de su primera versión. En Febrero de 2009 *Google* lanza su primera actualización 1.1 que se diferenciaba de la primera en poder adjuntar archivos en los mensajes.

Si echáramos la vista atrás y lo comparásemos con una versión actual, comprenderíamos la inmensa evolución del sistema operativo. Todas las actualizaciones siguientes incorporan funcionalidades nuevas o mejoran las ya existentes.

Todas las versiones de Android reciben del inglés el nombre de diferentes postres, siguiendo, además, orden alfabético.

- A: **Apple Pie (v1.0)**: tarta de manzana.
- B: **Banana Bread (v1.1):** pan de plátano.
- C: **Cupcake (v1.5):** panqué.
- D: **Donut (v1.6):** rosquilla.
- E: **Éclair (v2.0/v2.1):** pastel francés.
- F: **Froyo (v2.2)** (abreviatura de «frozen yogurt»): yogur helado.
- G: **Gingerbread (v2.3):** pan de jengibre.
- H: **Honeycomb (v3.0/v3.1/v3.2):** panal de miel.

- I: **Ice Cream Sandwich (v4.0):** emparedado de helado.
- J: **Jelly Bean (v4.1/v4.2/v4.3):** gominola.
- K: **KitKat (v4.4):** tableta de chocolate con leche.
- L: **Lollipop (v5.0/v5.1.1):** piruleta.
- M: **Marshmallow (v6.0/v6.0.1):** golosina malvavisco.
- N: **Nougat (v7.0/v7.1/v7.1.1/v7.1.2):** dulce francés.
- O: **Oreo (v8.0/8.1):** galleta de chocolate con crema.

Actualmente conviven muchas de las versiones mencionadas en el mercado, debido a que muchos terminales no se han actualizado a nuevas versiones. En la siguiente imagen puede observarse el porcentaje de terminales que usan las diferentes versiones.

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.6%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.6%
4.1.x	Jelly Bean	16	2.3%
4.2.x		17	3.3%
4.3		18	1.0%
4.4	KitKat	19	14.5%
5.0	Lollipop	21	6.7%
5.1		22	21.0%
6.0	Marshmallow	23	32.0%
7.0	Nougat	24	15.8%
7.1		25	2.0%
8.0	Oreo	26	0.2%

Data collected during a 7-day period ending on October 2, 2017.
Any versions with less than 0.1% distribution are not shown.

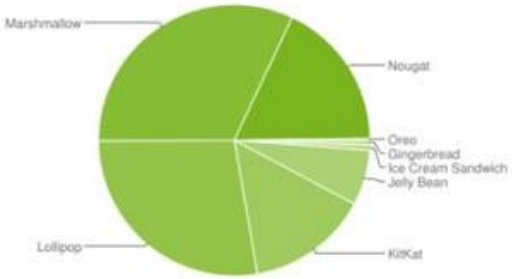


Ilustración 4. Versiones Android con mayor distribución en el mercado.

1.3 ¿POR QUÉ ANDROID Y NO IOS?

El sistema para desarrollar la app ha sido *Android* ya que es un sistema más abierto y libre para el desarrollador, permitiéndole crear apps sin tantas trabas, pudiendo tener en una app un control mucho más amplio del terminal que en *iOS*.

Otro punto a tener en cuenta es el coste de las herramientas para desarrollar. Mientras *iOS* requiere un *Mac* y pagar una licencia anual de 99$, en *Android* vale cualquier equipo actual y la licencia es de 25€ en un único pago para publicar en *Google Play* en el caso de que se publicara la aplicación.

Si hablamos de potencial de distribución de una app en cada uno de los sistemas, *Android* domina el mercado mundial. Es cierto que en USA el mercado es similar (53% *Android*, 44% *iOS*), pero en casos como España la diferencia se dispara hasta 10 veces más: *Android* 91,2%; *iOS* 8,3%.

Germany	3 m/e Feb '16	3 m/e Feb '17	% pt. Change
Android	76.6	76.4	-0.2
iOS	16.2	20.2	4.0
Windows	6.3	3.0	-3.3
Other	0.9	0.3	-0.6
GB	**3 m/e Feb '16**	**3 m/e Feb '17**	**% pt. Change**
Android	55.5	55.0	-0.5
iOS	37.8	42.3	4.5
Windows	6.2	2.1	-4.1
Other	0.4	0.6	0.2
France	**3 m/e Feb '16**	**3 m/e Feb '17**	**% pt. Change**
Android	71.8	73.4	1.6
iOS	19.9	24	4.1
Windows	7.4	2.4	-5.0
Other	0.9	0.2	-0.7
Italy	**3 m/e Feb '16**	**3 m/e Feb '17**	**% pt. Change**
Android	78.4	79.4	1.0
iOS	14.3	15.5	1.2
Windows	6.7	4.3	-2.4
Other	0.5	0.7	0.2
Spain	**3 m/e Feb '16**	**3 m/e Feb '17**	**% pt. Change**
Android	90	92.2	2.2
iOS	9.1	7.4	-1.7
Windows	0.9	0.4	-0.5
Other	0	0	0.0

USA	3 m/e Feb '16	3 m/e Feb '17	% pt. Change
Android	58.9	55.9	-3.0
iOS	38.3	42	3.7
Windows	2.6	1.7	-0.9
Other	0.2	0.4	0.2
China	**3 m/e Feb '16**	**3 m/e Feb '17**	**% pt. Change**
Android	77.1	86.4	9.3
iOS	22.1	13.2	-8.9
Windows	0.3	0.2	-0.1
Other	0.4	0.2	-0.2
Australia	**3 m/e Feb '16**	**3 m/e Feb '17**	**% pt. Change**
Android	55.1	59.5	4.4
iOS	38.2	38.9	0.7
Windows	5.8	0.7	-5.1
Other	0.9	0.9	0.0
Japan	**3 m/e Feb '16**	**3 m/e Feb '17**	**% pt. Change**
Android	48.2	53.9	5.7
iOS	50.2	44.8	-5.4
Windows	0.5	1.3	0.8
Other	1	0	-1.0
EU5	**3 m/e Feb '16**	**3 m/e Feb '17**	**% pt. Change**
Android	74.3	75.2	0.9
iOS	19.1	21.8	2.7
Windows	5.9	2.7	-3.3
Other	0.6	0.4	-0.3

Ilustración 5. Mercado de SO en varios países.

A nivel mundial, *Apple* tiene el 13% del mercado global, y *Android* en torno al 80%.

El punto más débil del desarrollo para *Android*, y el que más miedo suele dar a un desarrollador nuevo, es la gran cantidad de terminales diferentes que hay en el mercado.

A pesar de esa desventaja, entrar en el desarrollo *Android* es más fácil, más barato y encima se cubren a más usuarios potenciales.

1.4 HERRAMIENTAS PARA DESARROLLAR UNA APP

1.4.1 ANDROID

Dentro de la programación en *Android* podemos destacar dos entornos diferentes y en los cuáles se ha desarrollado este proyecto, *Android Studio* y *Mit App Inventor 2*.

1.4.1.1 ANDROID STUDIO

Es el entorno de desarrollo integrado oficial para la plataforma *Android.* Está basado en el software *IntelliJ IDEA* de *JetBrains* y ha sido publicado de forma gratuita a través de la *Licencia Apache 2.0*. Además está disponible para las plataformas *Microsoft Windows, macOS y GNU/Linux*.

Anteriormente, el desarrollo de aplicaciones para la plataforma *Android* se realizaba mediante *Eclipse*.

La última versión es la 3.0 la cual proporciona las siguientes características:

- Integración de ProGuard y funciones de firma de aplicaciones.
- Renderizado en tiempo real
- Consola de desarrollador: consejos de optimización, ayuda para la traducción, estadísticas de uso.
- Soporte para construcción basada en Gradle.
- Refactorización específica de Android y arreglos rápidos.
- Un editor de diseño enriquecido que permite a los usuarios arrastrar y soltar componentes de la interfaz de usuario.
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versiones y otros problemas.
- Plantillas para crear diseños comunes de Android y otros componentes.
- Soporte para programar aplicaciones para Android Wear.
- Soporte integrado para Google Cloud Platform, que permite la integración con Google Cloud Messaging y App Engine.
- Un dispositivo virtual de Android que se utiliza para ejecutar y probar aplicaciones.

La interfaz que podemos observar es la siguiente:

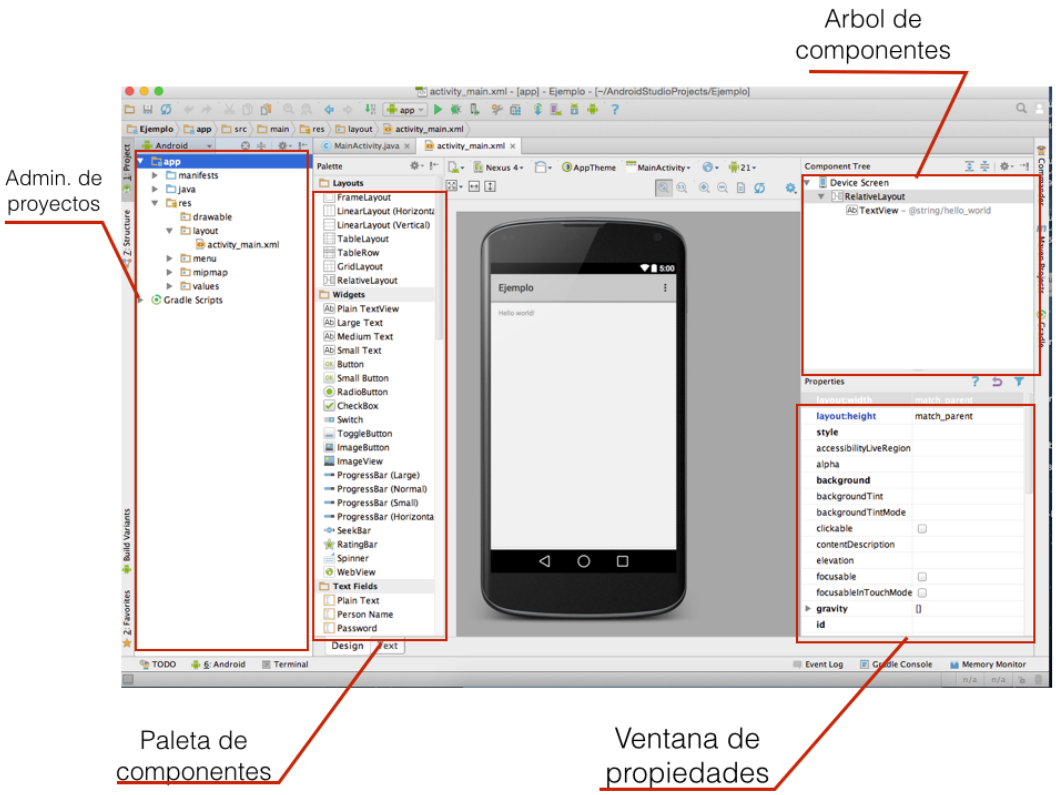


Ilustración 6. Interfaz Android Studio.

- **Administrador de proyectos:** En esta ventana, veremos los archivos y carpetas que le darán estructura a nuestra app.

- **Paleta de componentes:** Cuando trabajamos en modo diseño, esta ventana contiene todos los componentes que podemos colocar en la interfaz de nuestra app.

- **Árbol de componentes:** Es una lista de forma de árbol jerarquico en la que veremos los componentes que hemos colocado en la interfaz de nuestra app y como se agrupan.

- **Propiedades:** En esta ventana veremos la lista de propiedades que tiene cada uno de los componentes que hemos agregado a la interfaz.

1.4.1.2 MIT APP INVENTOR 2

App Inventor es un entorno de desarrollo de software creado por *Google Labs* para la elaboración de aplicaciones destinadas al sistema operativo *Android*. La plataforma se puso a disposición del público el 15 de diciembre de 2010 y está dirigida a personas que no están familiarizadas con la programación informática.

El usuario puede, de forma visual y a partir de un conjunto de herramientas básicas, ir enlazando una serie de bloques para crear la aplicación. El sistema es gratuito y se puede descargar fácilmente de la web.

Dentro de sus inconvenientes se encuentra que las aplicaciones creadas con App Inventor están limitadas por su simplicidad, aunque permiten cubrir un gran número de necesidades básicas en un dispositivo móvil.

Las características que tiene son:

- Basado en JavaScript para crear un lenguaje visual.
- Permite crear una aplicación en menos tiempo que otros entornos de programación.
- La interfaz gráfica permite al usuario crear aplicaciones con muchas funcionalidades.

La interfaz que podemos observar es la siguiente:

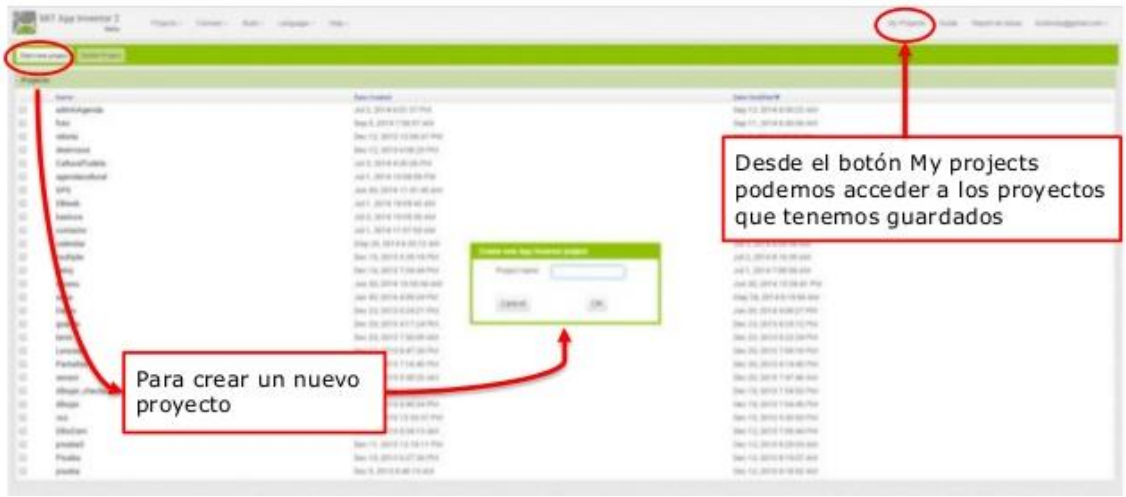


Ilustración 7. Mis proyectos.

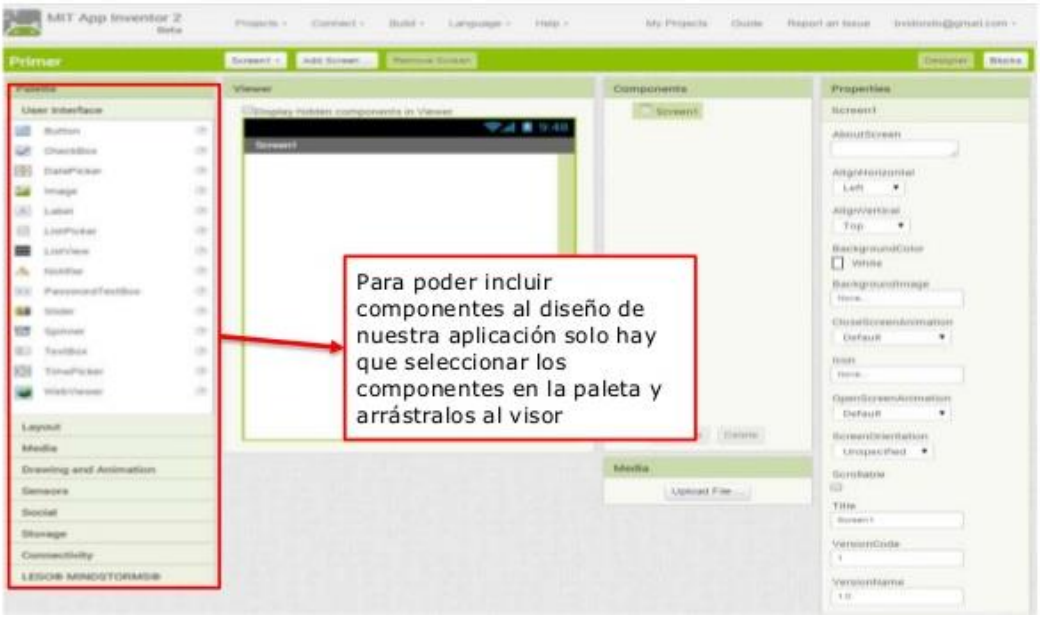


Ilustración 8. Diseñador.

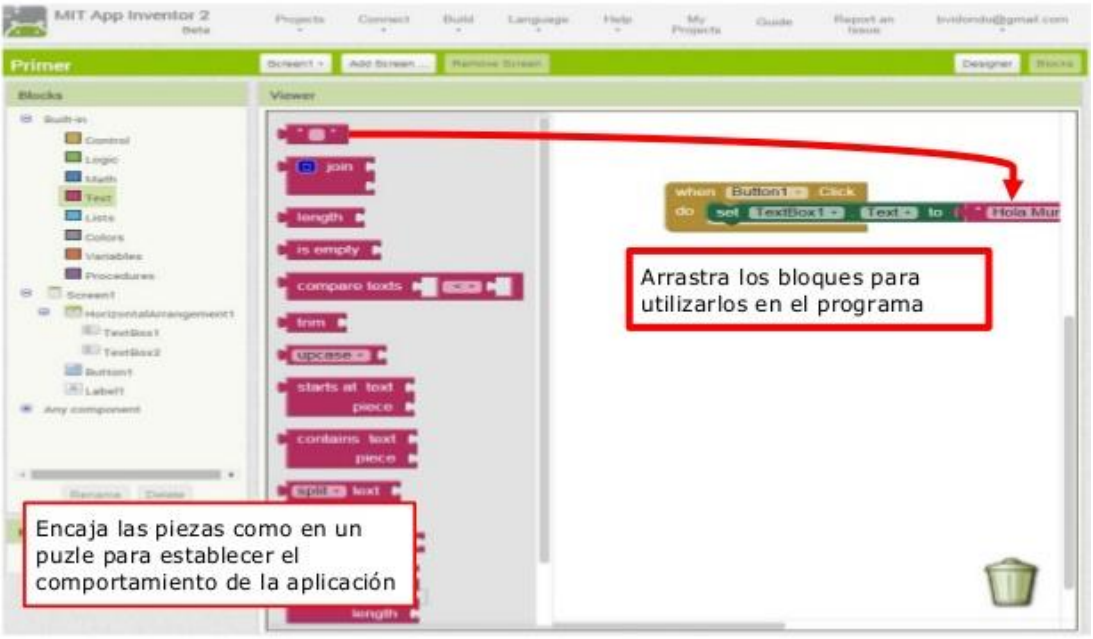


Ilustración 9. Editor de bloques.

1.4.2 IOS

Xcode es un entorno de desarrollo integrado para macOS que contiene un conjunto de herramientas creadas por Apple destinadas al desarrollo de software para *macOS, iOS, watchOS y tvOS*. Su primera versión tiene origen en el año 2003 y actualmente su versión número 9 se encuentra disponible de manera gratuita en el *Mac App Store* o mediante descarga directa desde la página para desarrolladores de *Apple*.

Entre las características más apreciadas de *Xcode* está la tecnología para distribuir el proceso de construcción a partir de código fuente entre varios ordenadores, utilizando *Bonjour.*

Xcode incluye la colección de compiladores del proyecto *GNU(GCC)*, y puede compilar código *C, C++, Swift, Objective-C, Objective-C++, Java y AppleScript* mediante una amplia gama de modelos de programación.

1.4.3 MULTIPLATAFORMA

Dentro de los frameworks multiplataforma *Apache Cordova*, *Ionic o Flutter* entre otras. Estos permiten el desarrollo de aplicaciones híbridas.

1.4.3.1 APACHE CORDOVA

Apache Cordova (versión de código abierto de PhoneGap) es un popular entorno de desarrollo de aplicaciones móviles.

Permite a los programadores de software, construir aplicaciones para dispositivos móviles utilizando *CSS3, HTML5, y JavaScript* en vez de utilizar APIs específicas de cada plataforma como *Android, iOS, o Windows Phone*.4 Permite encapsular *CSS, HTML*, y código de *Javascript* dependiendo de la plataforma del dispositivo.

Las aplicaciones resultantes son híbridas, lo que significa que no son ni una aplicación móvil nativa (porque toda la representación gráfica se realiza vía vistas de Web en vez del framework nativo) ni puramente basadas en web (porque no son solo aplicaciones web, sino que están empaquetadas como aplicaciones para su distribución y tienen acceso a las APIs nativas del dispositivo).

1.4.3.2 IONIC

Ionic nació en 2013. Framework basado en *AngularJS*, utiliza tecnologías web para desarrollar y renderizar las aplicaciones y se apoya en *PhoneGap* y *Cordova* para acceder a funciones nativas de los dispositivos. Esto significa que el mismo código se podrá ejecutar en todas las plataformas que se quiera, *Ionic* simplemente adaptará sus elementos a cada una de las plataformas a la hora de pintar elementos en la pantalla para dar una impresión "nativa".

Al igual que *Apache Cordova* son esencialmente páginas web incrustadas en una aplicación móvil a través de un *WebView.*

1.4.3.3 FLUTTER

Flutter es un framework de Google, presentado en 2018, de código abierto (basado en Dart) para crear aplicaciones nativas tanto para Android como para iOS*. Flutter* se integra en los editores más populares como *Android Studio, VSCode, Xcode, IntelliJ* y compañía, instalándose como un plugin, de modo que se puede seguir usando las mismas herramientas de siempre.

Además no hay archivos XML para las plantillas de las aplicaciones separadas del código, sino que la propia interfaz se genera desde el código.

2 DESCRIPCIÓN DE LA APLICACIÓN

La aplicación llamada *Engineering Quiz* sirve para evaluar a los alumnos de Ingeniería mediante una serie de preguntas y respuestas en las cuáles se presentan diferentes materias del Grado y dichos alumnos van respondiendo con el fin de ver que nota obtienen.

Engineering Quiz consta de una serie de actividades comenzando por la actividad inicial la cuál le da al usuario la opción de jugar y continuar a la siguiente actividad o la de cerrar la aplicación.

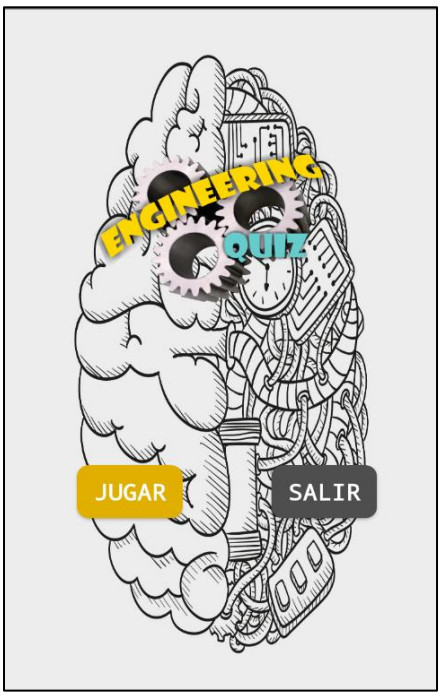


Ilustración 10. Pantalla principal.

Una vez el usuario haya seleccionado la opción de jugar se abrirá la segunda actividad en la cual podrá seleccionar en que materia quiere ser evaluado y una vez elegida dicha asignatura, pasará a la siguiente pantalla o actividad. Cabe destacar que cada materia abre una actividad principal distinta en las que aparecerán las cuestiones correspondientes a la asignatura seleccionada.

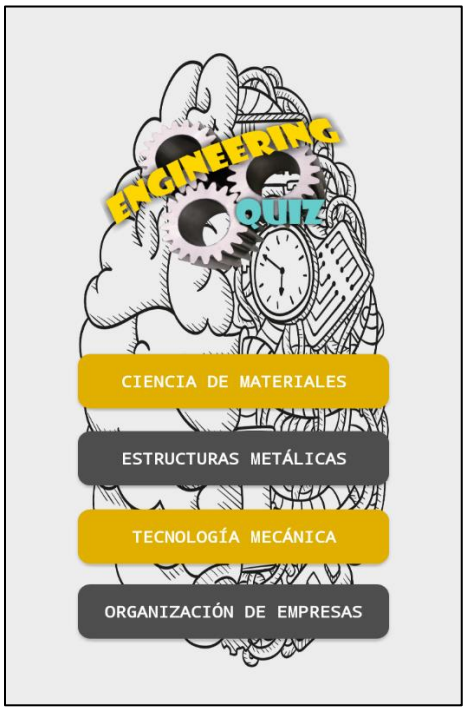


Ilustración 11. Pantalla de selección de asignaturas.

Como he comentado anteriormente, hay una actividad diferente para cada asignatura pero todas tienen el mismo funcionamiento. Dentro de esta pantalla el usuario va respondiendo a las cuestiones que le aparecen.

Cabe destacar que cada cuestión consta de una imagen, la cuestión correspondiente a esa imagen y tres respuestas de las cuales solo una es la correcta. Además para llevar a cabo dicha evaluación dentro de estas actividades aparece una puntuación que va aumentando de uno en uno conforme se vaya acertando la pregunta, esta puntuación se puede observar en la imagen siguiente en la esquina superior derecha:

Ilustración 12. Cuestionario.

Por cierto, para este Trabajo se han incluido tres preguntas por asignaturas y un total de cuatro asignaturas, aunque se podría aumentar dicho número tanto de cuestiones como materias sin ningún problema.

A continuación se verá una serie de ilustraciones con las preguntas y respuestas correspondientes a cada asignatura.

Para la asignatura de Ciencia de Materiales, las cuestiones que le aparecerán al usuario son las siguientes:

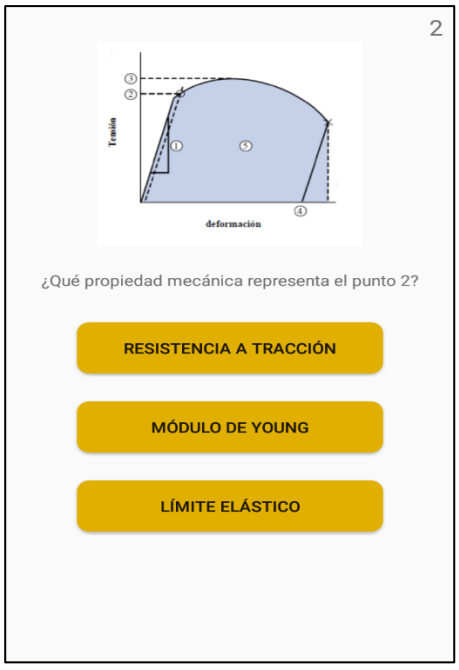
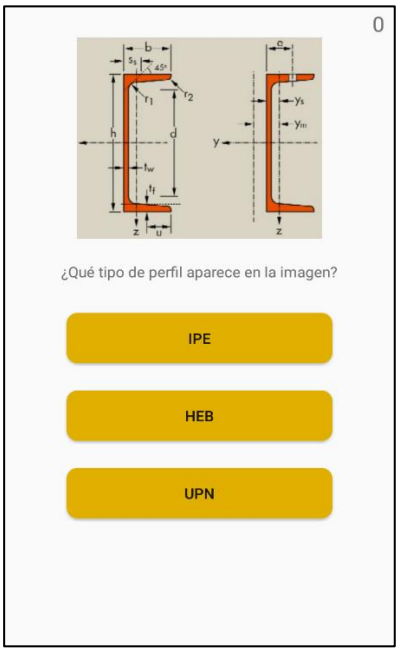
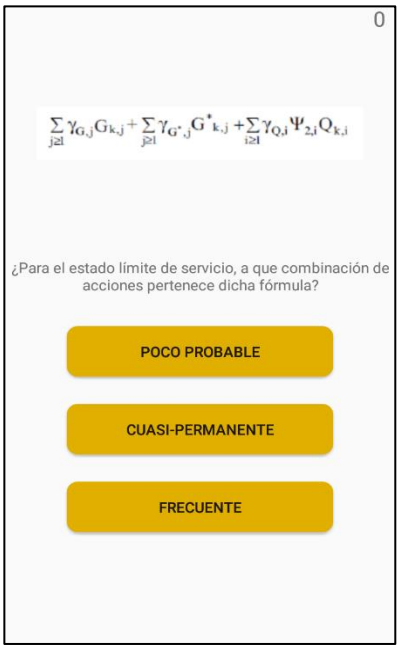


Ilustración 13. Cuestiones Ciencia de Materiales

Dentro de la asignatura Estructuras Metálicas tendremos que responder las cuestiones que aparecen a continuación:





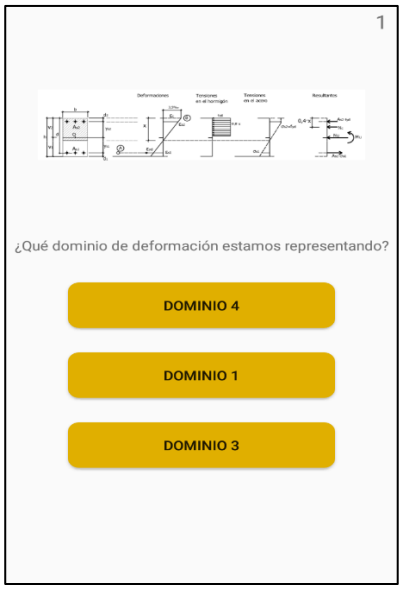


Ilustración 14. Cuestiones Estructuras Metálicas.

Las cuestiones que aparecen en la categoría de Tecnología Mecánica son las siguientes:

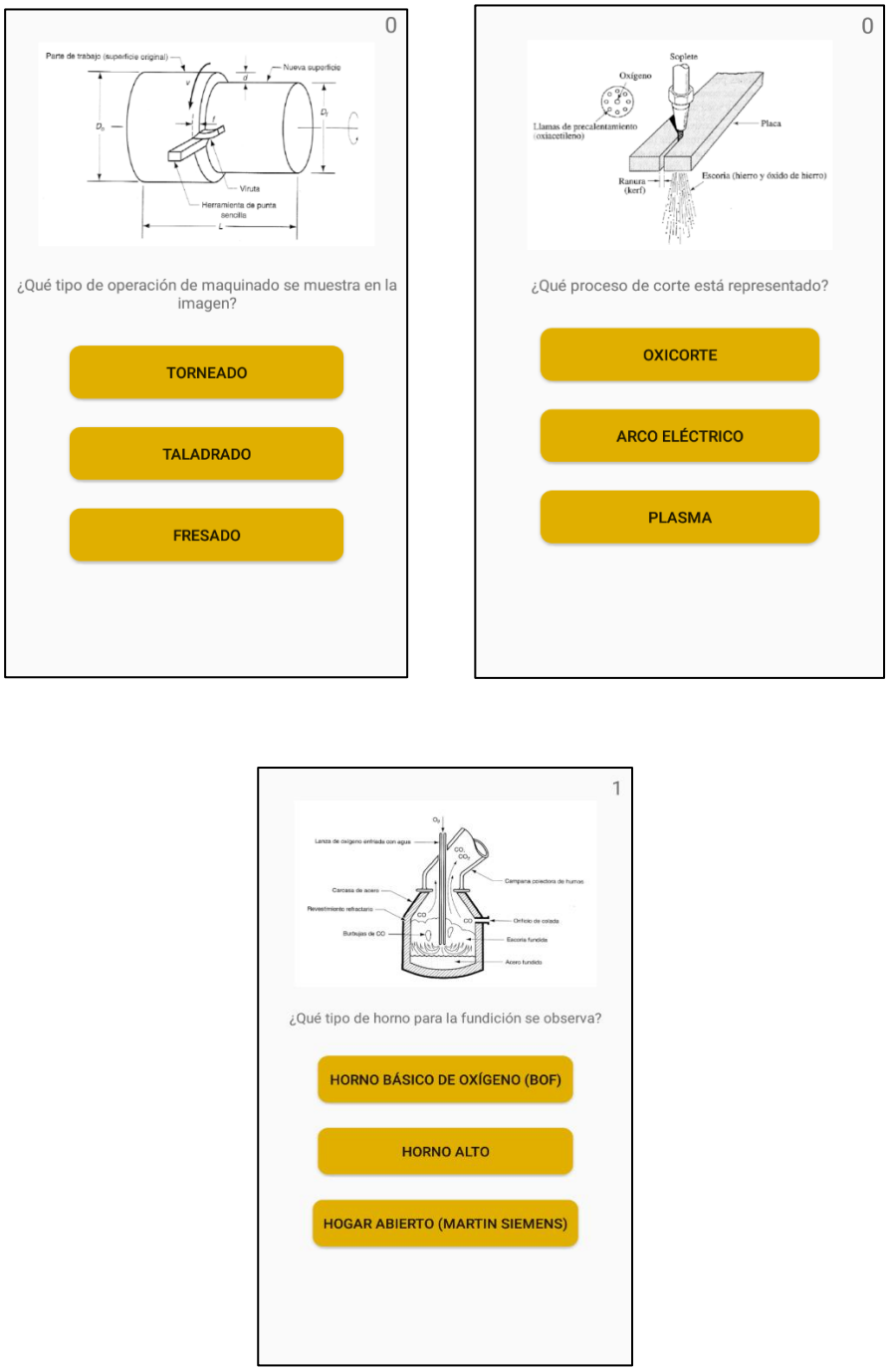


Ilustración 15. Cuestiones Tecnología Mecánica.

Por último se muestran las correspondientes a la asignatura de Organización de Empresas:

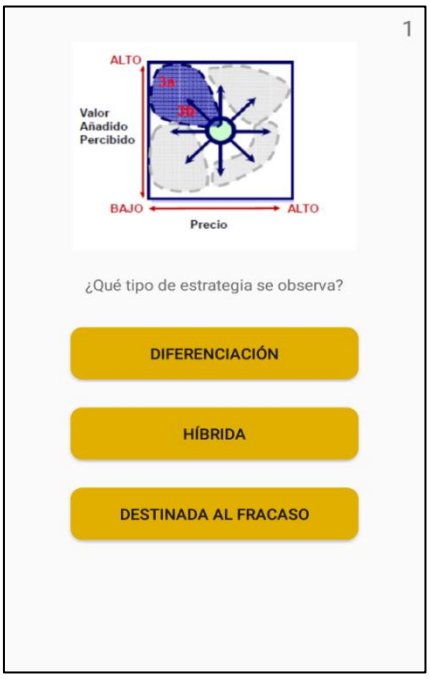


Ilustración 16. Cuestiones Organización de Empresas.

Conforme se vayan respondiendo las cuestiones, además de aumentar la puntuación si se acierta, aparecerá una notificación que nos diga si la respuesta es correcta o incorrecta como podemos ver en las dos imágenes siguientes:

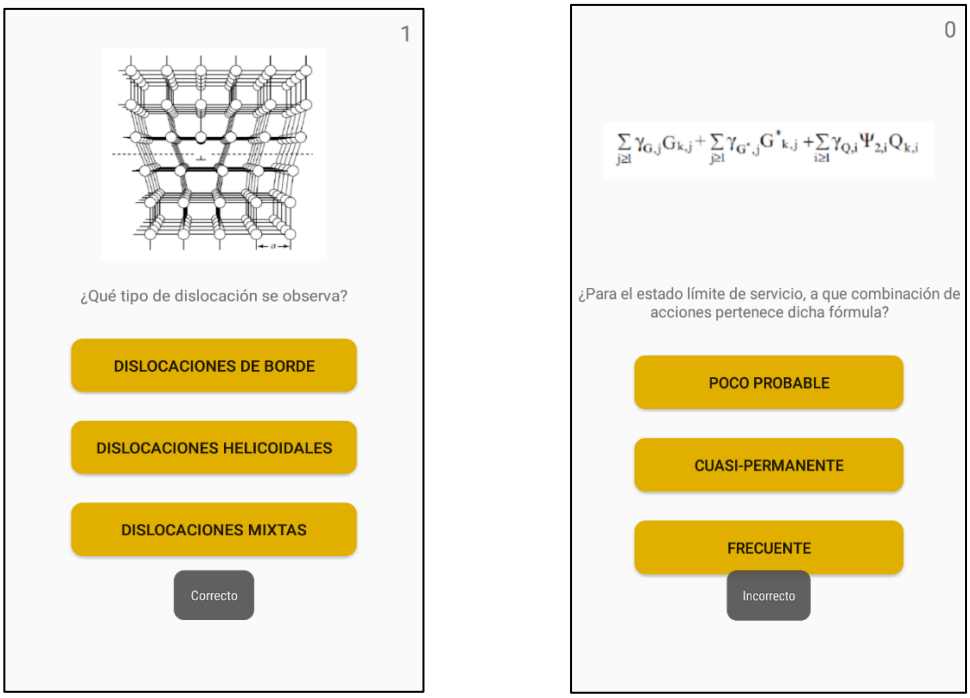


Ilustración 17. Notificaciones.

Por último una vez respondidas todas las preguntas se nos abre la última actividad en la cual nos aparece la puntuación que hemos obtenido respecto al total de preguntas. Además hay dos botones, uno para compartir, con el que el usuario puede enviar la puntuación que ha obtenido al profesor de dicha asignatura, y otro botón para volver a la actividad inicial:

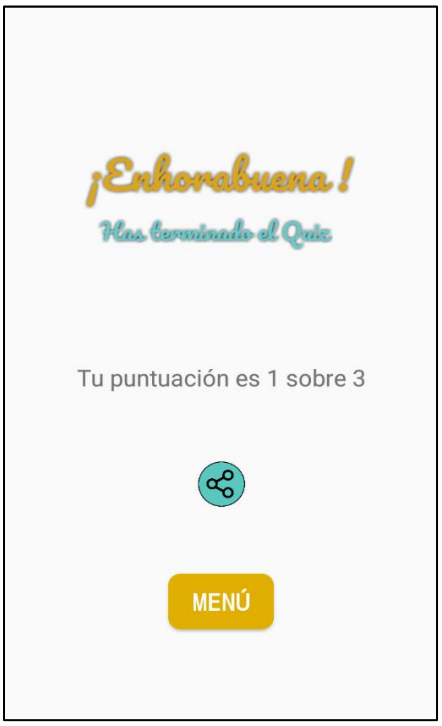


Ilustración 18. Pantalla de puntuación.

Pulsando el botón de compartir se nos abrirá una pestaña en la cual nos aparecerá una serie de aplicaciones con las cuales podemos compartir dicha puntuación:

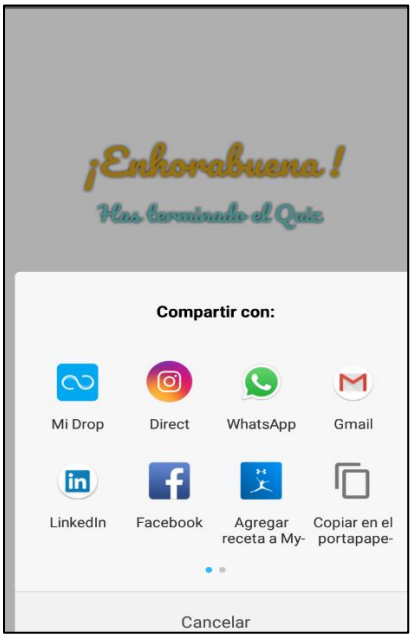


Ilustración 19. Pestaña de compartir,

El objetivo es enviar la puntuación obtenida al tutor de la asignatura de la cual hemos sido evaluados por lo que pinchando en Gmail se nos abre el correo y automáticamente aparece escrita la puntuación obtenida

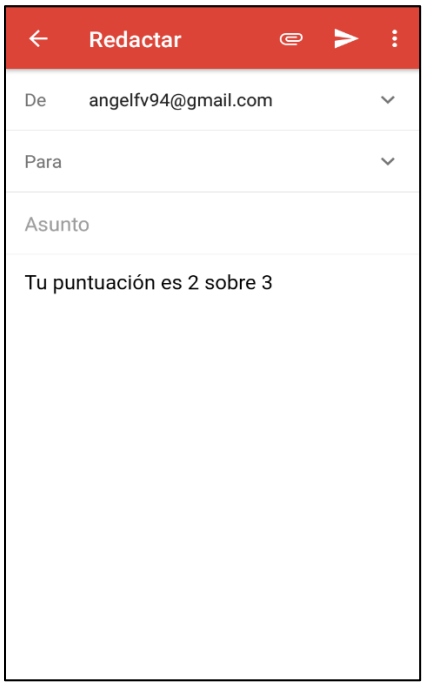


Ilustración 20. Compartir con Gmail.

3 MATERIALES Y MÉTODOS

3.1 ANDROID STUDIO

En este apartado se comentará el desarrollo de la aplicación dentro del entorno de programación Android Studio.

3.1.1 DISEÑO Y PROGRAMACIÓN

Los pasos a seguir antes de comenzar a programar nuestra aplicación en Android Studio serán:

- Configurar nuestro proyecto, es decir, darle un nombre que en nuestro caso se llamará EngineeringQuiz.

- Seleccionar donde queremos que funcione nuestra app, yo he realizado una aplicación para smartphones y tablets.

- Añadir una actividad y darle un nombre. El nombre que le hemos dado a nuestra actividad principal es MainActivity que es el que viene por defecto.

Una vez realizado todo este proceso comenzamos a desarrollar la aplicación utilizando archivos *.xml* y *.java* los cuales se explican en los siguientes apartados.

3.1.1.1 ARCHIVOS XML

Dentro de este tipo de archivos podemos destacar el *Android Manifest* que contiene nodos descriptivos sobre las características de la aplicación.

Estas pueden ser los building blocks existentes, la versión SDK usada o los permisos necesarios para ejecutar algunos servicios entre otras. A continuación podemos ver el manifest de esta app en la cual aparece parte de la información así como permisos para acceder a internet:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.engineeringquiz">

    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/iconapp"
        android:label="@string/app_name"
        android:roundIcon="@drawable/iconapp"
        android:supportsRtl="true"
        android:theme="@style/Theme.AppCompat.Light.NoActionBar">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".QuizActivity"
            android:screenOrientation="portrait"/>
        <activity android:name=".ResultsActivity"
            android:screenOrientation="portrait"/>
        <activity android:name=".MenuActivity"
            android:screenOrientation="portrait"/>
        <activity android:name=".Quiz2Activity"
            android:screenOrientation="portrait"/>
        <activity android:name=".Quiz3Activity"
            android:screenOrientation="portrait"/>
        <activity android:name=".Quiz4Activity"
            android:screenOrientation="portrait"></activity>
    </application>

</manifest>
```

Dentro de la carpeta layout encontramos los archivos con los que se crea la interfaz de usuario que aparecerá en el Smartphone y con la cual se podrá interactuar. En ellos se establecen los widgets que agregan una actividad.

Los archivos layout que he desarrollado son los siguientes:

- **activity_main.xml** (Pantalla de inicio)
- **activity_menu.xml** (Pantalla con asignaturas)
- **activity_quiz.xml** (Pantalla con cuestiones de asignatura)
- **activity_quiz2.xml** (Pantalla con cuestiones de asignatura)
- **activity_quiz3.xml** (Pantalla con cuestiones de asignatura)
- **activity_quiz4.xml** (Pantalla con cuestiones de asignatura)
- **activity_results.xml** (Pantalla final con resultados)

Para la realización de un layout podemos diferenciar entre la parte de *Design* y la de *Text*. En *Design* arrastramos los elementos de la paleta de componentes sobre la pantalla y los ubicamos a nuestro gusto.

Mientras que en la parte de *Text* es donde podemos editar los componentes que hemos añadido para cambiarles la forma, el color, tamaño entre otra gran variedad de opciones. En las imágenes siguientes podemos ver un ejemplo para la pantalla de las preguntas:

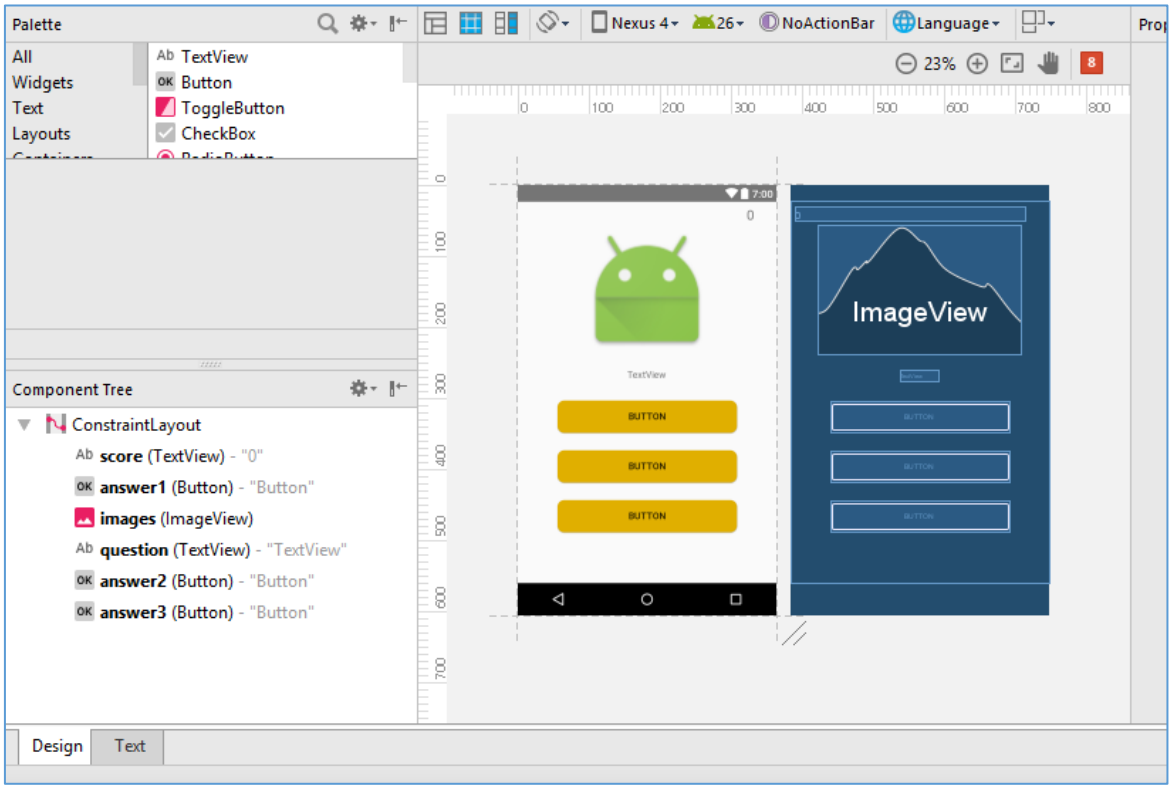


Ilustración 21. Archivos layout (Design).

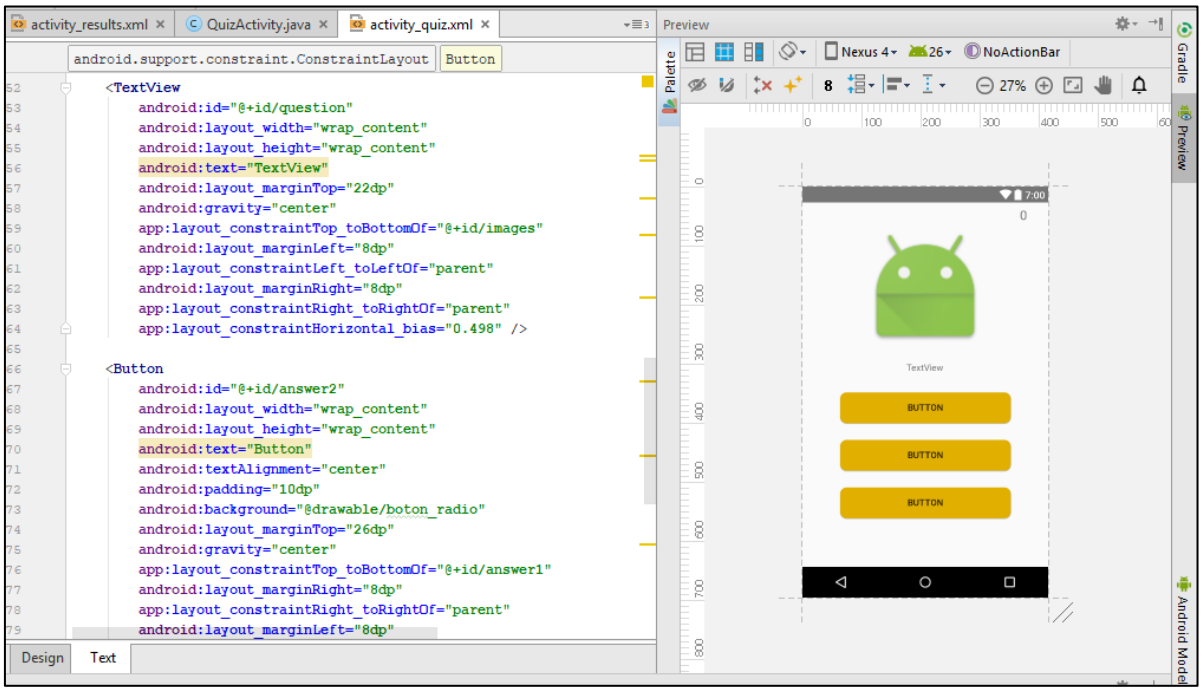


Ilustración 22. Archivos layout (Text).

En la carpeta drawable tenemos almacenadas las imágenes que hemos utilizado para la aplicación así como el diseño de los botones con los cuáles se interactúa. Todo esto se puede ver a continuación:

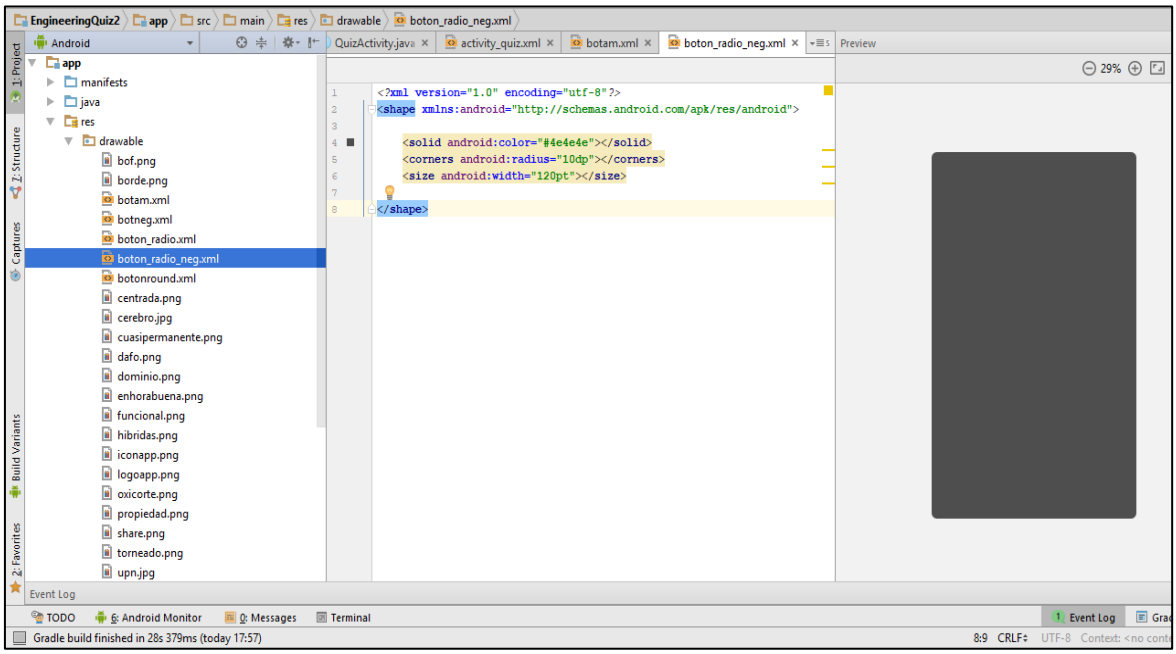


Ilustración 23. Botones creados.

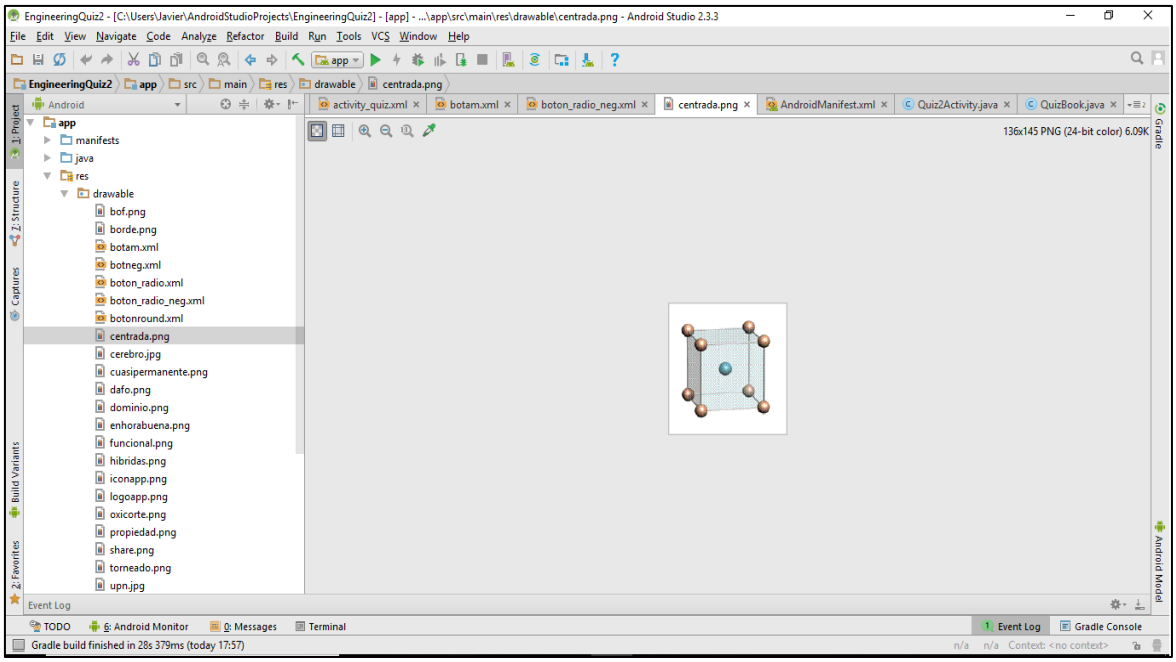


Ilustración 24. Imágenes para aplicación.

Los archivos que se han utilizado para el diseño de los iconos de los botones son los siguientes:

- **botam.xml**
- **botneg.xml**
- **boton_radio.xml**
- **boton_radio_neg.xml**
- **botonround.xml**

Por último otros archivos *.xml* utilizados en la aplicación son los siguientes:

Diseño de los estilos:

- **styles.xml**

Cadenas de texto:

- **strings.xml**

3.1.1.2 ARCHIVOS JAVA

Aquí se alojarán todos los archivos relacionados con nuestras actividades y otros archivos fuente auxiliares. Al abrir uno de estos archivos veremos toda la lógica necesaria para que la actividad interactúe de manera correcta con el usuario, es decir aquí se programa el funcionamiento de lo que se ha realizado en los archivos *.xml*.

Dentro de los archivos relacionados con las actividades tenemos:

- **MainActivity** (Contiene el funcionamiento de la página inicial)
- **MenuActivity** (Contiene el funcionamiento de la página de elección de asignaturas)
- **QuizActivity** (Contiene el funcionamiento de un cuestionario)
- **Quiz2Activity** (Contiene el funcionamiento de un cuestionario)
- **Quiz3Activity** (Contiene el funcionamiento de un cuestionario)
- **Quiz4Activity** (Contiene el funcionamiento de un cuestionario)
- **ResultsActivity** (Contiene el funcionamiento de la página de resultados)

Además también podemos diferenciar en archivos auxiliares en los cuáles actúan como base de datos y se almacenan tanto las asignaturas de las cuales

se examina en la aplicación como las preguntas, respuestas e imágenes para cada asignatura:

- **Asignaturas** (Archivo que contiene la lista con asignaturas)
- **QuizBook** (Contiene la lista con preguntas, respuestas e imágenes de una asignatura)
- **QuizBook2** (Contiene la lista con preguntas, respuestas e imágenes de una asignatura)
- **QuizBook3** (Contiene la lista con preguntas, respuestas e imágenes de una asignatura)
- **QuizBook4** (Contiene la lista con preguntas, respuestas e imágenes de una asignatura)

A continuación se explica la programación de estos archivos para el buen funcionamiento de la aplicación.

MainActivity

Para el botón de *JUGAR* la programación es la siguiente:

```
mStarButton = (Button) findViewById(R.id.empezar);
mStarButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(MainActivity.this,
MenuActivity.class));
    }
});
```

Por el contrario para el de *SALIR* realizamos lo siguiente:

```
mExitButton = (Button) findViewById(R.id.salir);
mExitButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        finish();
        Intent intent = new Intent(Intent.ACTION_MAIN);
        intent.addCategory(Intent.CATEGORY_HOME);
        intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
        startActivity(intent);
```

```
    }
});
```

MenuActivity

Al iniciarse la pantalla se descargan las asignaturas para que aparezcan en los botones:

```
updatematerias();


private void updatematerias() {
    mbutton.setText(Asignaturas.materia[0]);
    mbutton2.setText(Asignaturas.materia[1]);
    mbutton3.setText(Asignaturas.materia[2]);
    mbutton4.setText(Asignaturas.materia[3]);
}
```

Además el funcionamiento de los botones es el mismo para todos:

```
mbutton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(MenuActivity.this,
QuizActivity.class));
        MenuActivity.this.finish();
    }
});
```

QuizActivity

Todas las pantallas de QuizActivity tienen el mismo funcionamiento por lo que aquí se muestra la programación de una de ellas. Este archivo es el que más código tiene ya que el funcionamiento principal de la aplicación se encuentra dentro de estos archivos.

Al iniciarse la pantalla correspondiente a QuizActivity la primera acción que realiza es descargar las cuestiones:

```
updateQuestion();
private void updateQuestion(){

    mImageView.setImageResource(QuizBook.images[mQuestionNumber]);
    mQuestion.setText(QuizBook.questions[mQuestionNumber]);
    mAnswer1.setText(QuizBook.choice1[mQuestionNumber]);
    mAnswer2.setText(QuizBook.choice2[mQuestionNumber]);
    mAnswer3.setText(QuizBook.choice3[mQuestionNumber]);
    mAnswer = QuizBook.answers[mQuestionNumber];;
    mQuestionNumber++;

}
```

A continuación vemos el funcionamiento para cada respuesta. Dentro de esta está programado el almacenamiento de la puntuación así como el cambio de preguntas y la notificación de correcto e incorrecto dependiendo de nuestra elección.

Este código lo que hace es al responder nos aparece la notificación si hemos acertado o no, nos suma la puntuación en el caso de que la respuesta sea correcta y por último descarga la siguiente cuestión:

```
//Lógica de los botones

mAnswer1.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        if (mAnswer1.getText() == mAnswer){
            mScore++; //This updates the score int variable
            updateScore(mScore); // This converts the int variable to
a String
            Toast.makeText(QuizActivity.this, "Correcto",
Toast.LENGTH_SHORT).show();

            //Perform this check before you update the question
            if (mQuestionNumber == QuizBook.questions.length){
                Intent i = new Intent(QuizActivity.this,
ResultsActivity.class);
                Bundle bundle = new Bundle();
                bundle.putInt("finalScore", mScore);
                i.putExtras(bundle);
                QuizActivity.this.finish();
                startActivity(i);
            }else {
                updateQuestion();
            }
        }

        // Si la respuesta es incorrecta
        else{
            if (mQuestionNumber == QuizBook.questions.length){
                Intent i = new Intent(QuizActivity.this,
ResultsActivity.class);
                Bundle bundle = new Bundle();
                bundle.putInt("finalScore", mScore);
                i.putExtras(bundle);
                QuizActivity.this.finish();
                startActivity(i);
                Toast.makeText(QuizActivity.this, "Incorrecto",
Toast.LENGTH_SHORT).show();
            }else {
                updateQuestion();
                Toast.makeText(QuizActivity.this, "Incorrecto",
Toast.LENGTH_SHORT).show();
            }
        }
    }
});
```

En el caso de que se hayan respondido todas las preguntas, nos enviará al layout de ResultsActivity.

ResultsActivity

Aquí nos aparecerá la puntuación final obtenida en función del número total de cuestiones que hay almacenadas además de poder compartir dicha puntuación. El código empleado es el siguiente:

```
mFinalScore.setText("Tu puntuación es " +score+ " sobre " +
QuizBook.questions.length);

mShare.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent intent = new Intent(Intent.ACTION_SEND);
        intent.setType("text/plain");
        intent.putExtra(Intent.EXTRA_TEXT, "Tu puntuación es "
+score+ " sobre " + QuizBook.questions.length);
        startActivity(Intent.createChooser(intent, "Compartir
con:"));
    }
});
```

Además en esta pantalla tenemos el botón de volver al menú inicial:

```
mRetryButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(ResultsActivity.this,
MainActivity.class));
        ResultsActivity.this.finish();

    }
});
```

Por último se muestra el archivo .java donde se almacenan las cuestiones junto a las respuestas y las imágenes. Para todas las asignaturas funciona igual.

QuizBook

```
public static String[] questions = new String[]{

            "¿Qué tipo de estructura se observa en la imagen?",
            "¿Qué tipo de dislocación se observa?",
            "¿Qué propiedad mecánica representa el punto 2?"
    };

    public static int[] images = new int[]{

            R.drawable.centrada,
            R.drawable.borde,
            R.drawable.propiedad
    };

    public static String [] choice1 = new String[]{
            "Cúbica centrada en las caras (FCC)", "Dislocaciones de
borde", "Resistencia a tracción"
    };

    public static String [] choice2 = new String[]{
            "Cúbica centrada en el cuerpo (BCC)", "Dislocaciones
helicoidales", "Módulo de Young"
    };

    public static String [] choice3 = new String[]{
            "Hexagonal compacta (HCP)", "Dislocaciones mixtas",
"Límite elástico"
    };

    public static String[] answers = new String[]{

            "Cúbica centrada en el cuerpo (BCC)",
            "Dislocaciones de borde",
            "Límite elástico"
    };
}
```

3.2 MIT APP INVENTOR 2

3.2.1 DISEÑO Y PROGRAMACIÓN

Como hemos comentado anteriormente *Mit App Inventor 2* utiliza una programación visual mediante bloques para generar el código. En este apartado se va explicar cómo se ha desarrollado la app dentro de este entorno de este entorno de programación.

En primer lugar se genera un nuevo proyecto en la pantalla principal de *Mit App Inventor 2* en la cual se almacenan todos nuestros proyectos

desarrollados. Este proyecto recibe el nombre de *EngineegingQuizMit* para diferenciarlo del realizado en *Android Studio*.

Una vez creado se nos abre el entorno de programación en el que podemos diferenciar la parte de *Diseñador* y la de *Bloques*. En los siguientes apartados pasaremos a explicar como ha evolucionado el diseño y la programación de la app.

3.2.1.1 DISEÑADOR

Esta es la parte en la que se desarrolla la interfaz gráfica que verá el usuario mientras utilice la app. La aplicación, al ser igual que la desarrollada en Android Studio, tiene el mismo diseño con unas pequeñas diferencias que se verán más adelante.

Una vez se inicia la parte de diseñador nos aparece la primera Screen que se abrirá al iniciar la aplicación en un dispositivo. En nuestro caso la Screen1 es la pantalla principal que aparecen los botones de *JUGAR* Y *SALIR*. Esta Screen tiene la siguiente visualización dentro de Mit App Inventor 2:

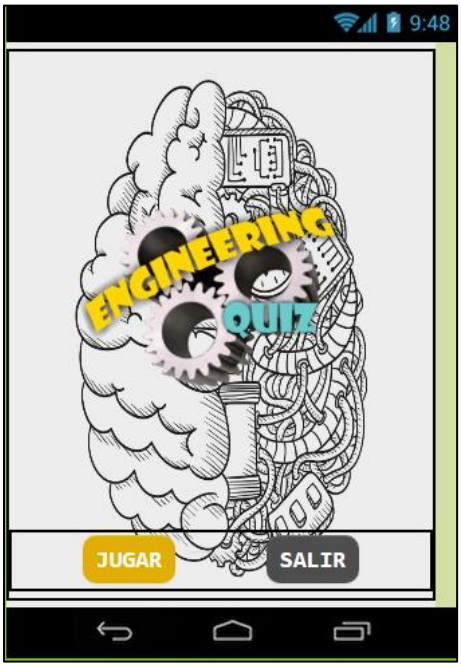


Ilustración 25. Interfaz Screen1.

Podemos observar que dentro de la parte Diseño hemos colocado el fondo de pantalla, el logotipo de la app y los botones con los nombres de

JUGAR y SALIR ya que estos no variarán. Veremos que esto no se realiza igual para las siguientes Screens.

La siguiente pantalla creada es Screen0 (el hecho de la elección de los nombres se explica al final del apartado) la cual muestra las asignaturas que se pueden elegir para ser evaluados.

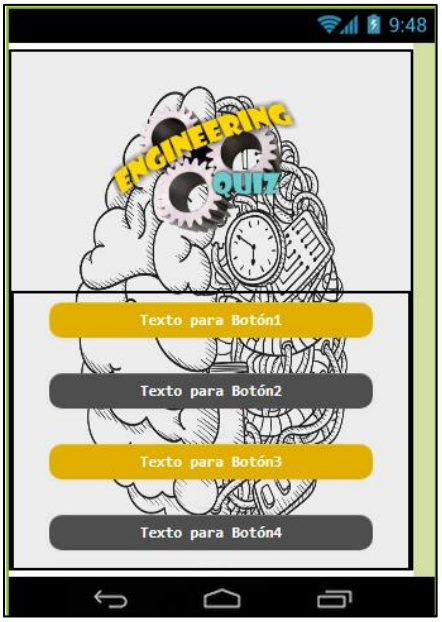


Ilustración 26. Interfaz Screen0.

Se observa como se mantiene el logotipo y el fondo pero a diferencia de la anterior pantalla no aparecen los nombres de las asignaturas, esto se debe a que tanto el número como el tipo de asignaturas pueden variar con el desarrollo de la app.

Las siguientes pantallas creadas son las del cuestionario. Aquí tenemos la Screen2, Screen3, Screen4 y Screen6. En estas ventanas nos aparecerán las preguntas y respuestas junto a la foto correspondiente y la puntuación obtenida.

La interfaz de usuario es la misma en todas las pantallas la diferencia es que cada Screen pertenece a una asignatura diferente:

- **Screen2:** Estructuras Metálicas
- **Screen3:** Tecnología de Materiales
- **Screen4:** Organización de Empresas
- **Screen6:** Ciencias de Materiales

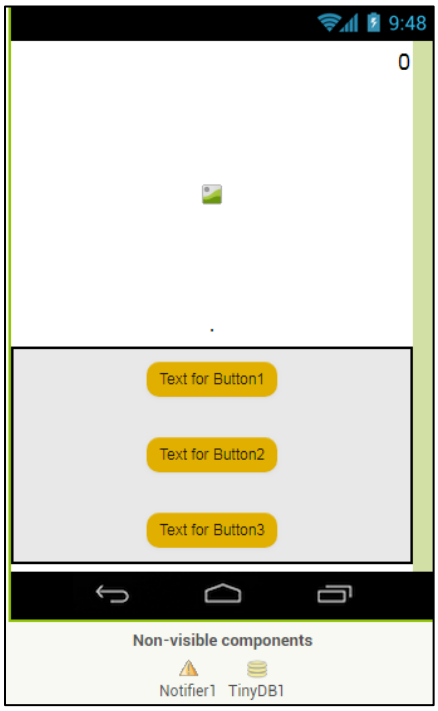


Ilustración 27. Interfaz Screen2, Screen3, Screen4 y Screen6.

Como vemos en la imagen no aparecen las preguntas ni las respuestas establecidas ya que estas aparecerán aleatoriamente. Además tenemos dos elementos no visibles, uno será la notificación de Correcto o Incorrecto y el otro es la base de datos en la que almacenamos la puntuación que vamos obteniendo.

Por último tenemos la Screen5 en la que nos aparecerá la puntuación obtenida y podremos compartirla con las redes sociales o por Gmail para enviarla al profesor de la asignatura mediante el botón que aparece de compartir. Además tenemos un botón para volver al menú principal así como la base de datos en elementos no visibles para poder mostrar la puntuación que se ha ido almacenando en las pantallas anteriores.

Ilustración 28. Interfaz Screen5.

Observamos también que tanto el nombre del botón MENÚ como el logo están colocados en la interfaz porque se mantienen estables y no cambiarán.

Estos nombres y el hecho de que la Screen5 sea la de resultados y no la Screen6 se debe a la dificultad de manejar los archivos de las Screens dentro de *Mit App Inventor 2* ya que para cambiar los nombres o la pantalla de inicio es necesario descargar el archivo *.aia* y descargarlo en el ordenador para manipularlo mediante un editor de texto *Notepad++* el cual tuve que instalar para cambiar la pantalla inicial ya que comencé la aplicación desarrollando primero la Screen de preguntas y respuestas.

3.2.1.2 BLOQUES

Dentro de las Screens anteriores tenemos también la parte de bloques en la cual se realiza la programación que le dará el funcionamiento a la parte de interfaz de usuario.

En la Screen1 se le da el funcionamiento de pasar a la siguiente pantalla o salir de la aplicación:

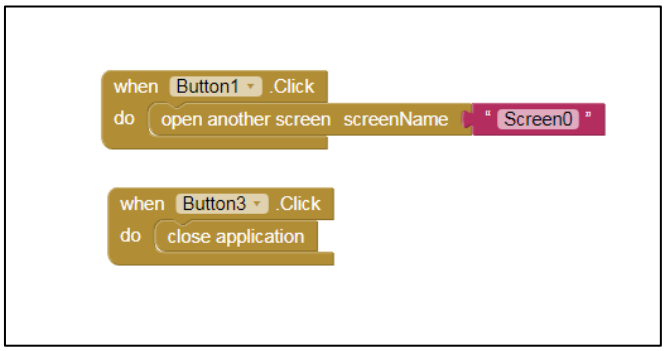


Ilustración 29. Funcionamiento Screen1.

La Screen0 nos enviará a la Screen correspondiente a la asignatura que hayamos pulsado, para ello se crea una lista para almacenar las asignaturas que aparezcan en la pantalla.

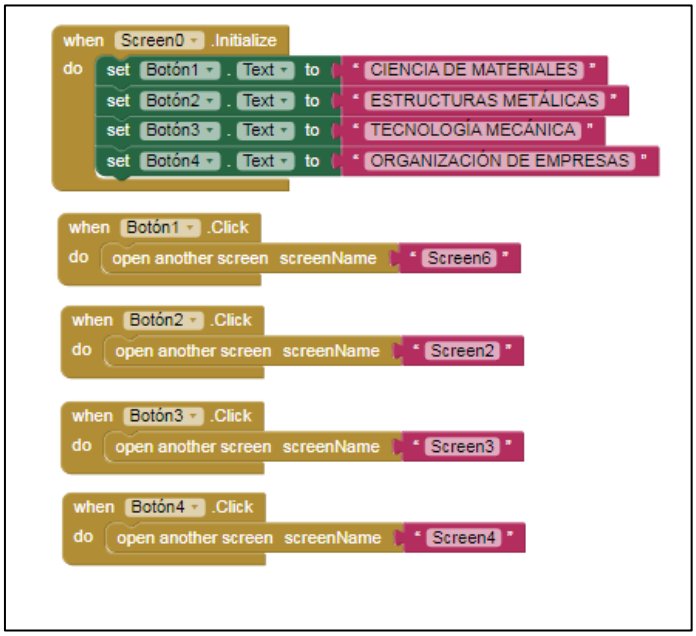


Ilustración 30. Funcionamiento Screen0.

Las Screens 2, 3, 4 y 6 tienen el mismo funcionamiento y en ellas se crea una lista con las preguntas, respuestas y fotos correspondientes. Estas se programan para que vayan apareciendo aleatoriamente en la pantalla y no se repita el orden.

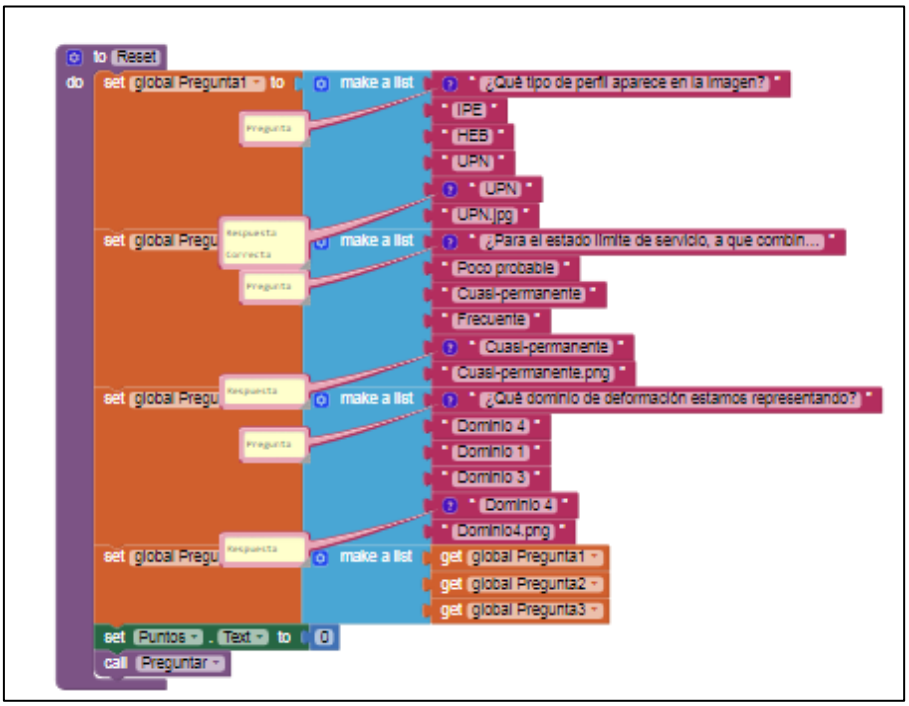


Ilustración 31. Lista con preguntas, respuestas e imágenes.

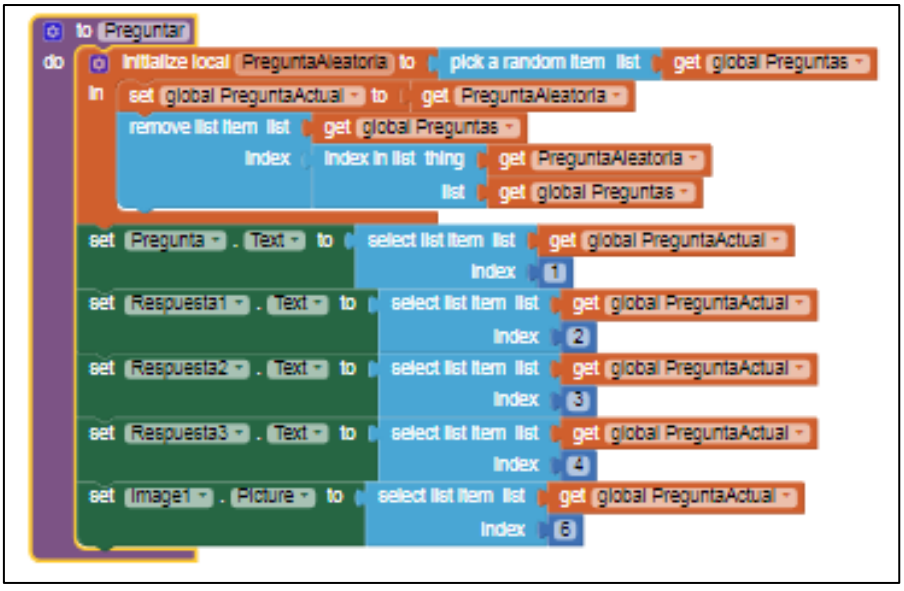


Ilustración 32. Funcionamiento preguntas aleatorias.

Además programamos el funcionamiento de la puntuación así como un notificador que nos avisa si la respuesta es correcta o incorrecta.

Ilustración 33. Funcionamiento de las cuestiones y puntuación.

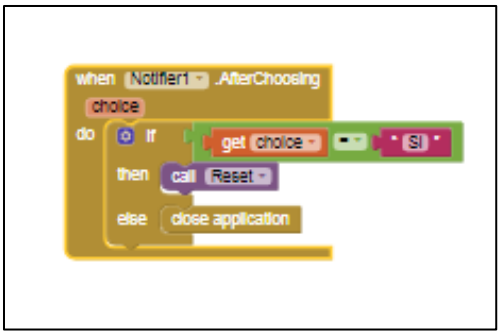


Ilustración 34. Funcionamiento del notificador.

Por último se crea una base de datos con TinyDB para almacenar la puntuación y que aparezca en la pantalla de resultados.

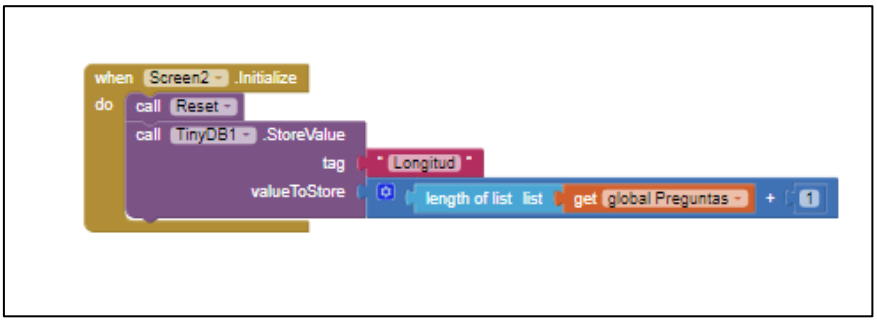


Ilustración 35. Funcionamiento base de datos TinyDB.

Por último en la Screen5 se aplican los bloques para que aparezca la puntuación obtenida cogiéndola de la base de datos así como el botón de compartir y que se comparta esa puntuación. Además con el botón menú hacemos que al pulsarlo vuelva a la Screen1.

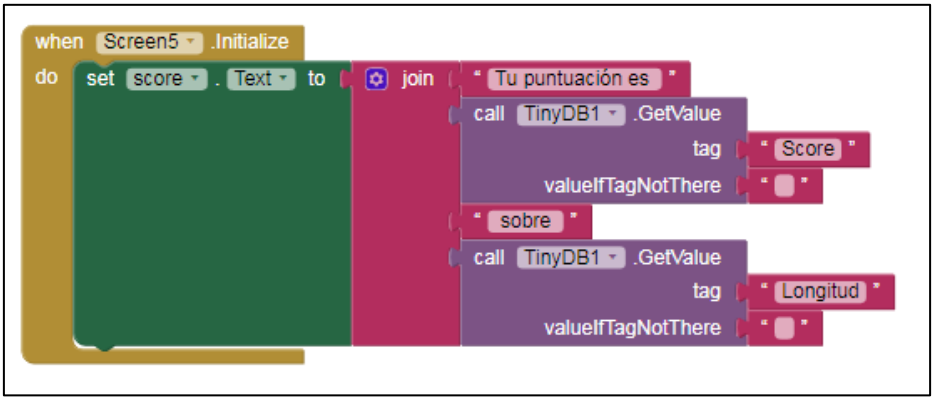


Ilustración 36. Funcionamiento para la obtención de la puntuación final.

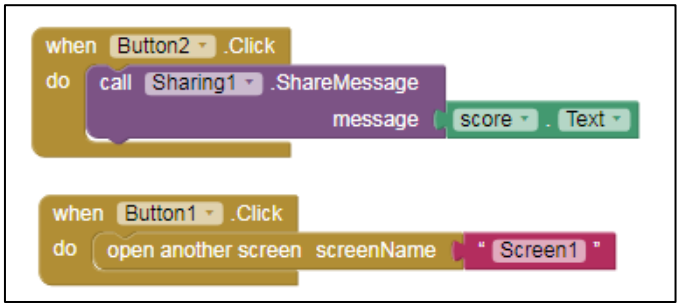


Ilustración 37. Funcionamiento botón compartir y botón MENÚ.

3.3 DISEÑO GRÁFICO

Para el desarrollo de esta aplicación se ha llevado a cabo el diseño del Logo, el título de la pantalla de resultados así como el para botón compartir. El programa utilizado ha sido la app Logo Maker la cual te permite seleccionar un tipo de letra, colocar una imagen y editarlo a tu gusto. Estos tres diseños son los siguientes:

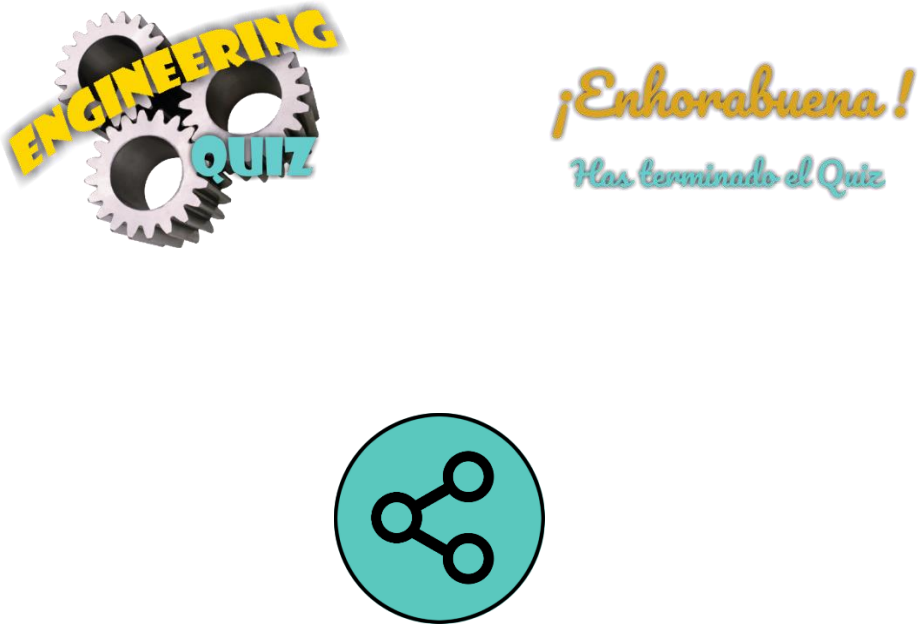


Ilustración 38. Diseños con Logo Maker.

4 RESULTADOS Y DISCUSIÓN

Tras haber realizado la misma aplicación en dos entornos de programación distintos aunque ambas para el sistema operativo *Android* puedo destacar una serie de diferencias que se me han presentado para cada entorno durante el transcurso del desarrollo de la app.

Los puntos fuertes que se pueden destacar de *Mit App Inventor 2* es la facilidad de desarrollo ya que la programación mediante bloques es muy visual y se hace muy intuitiva. Además tiene la opción de variar el idioma lo que te da la facilidad para entenderlo todo ya que puedes poner tu idioma nativo y cambia el idioma tanto de la parte de diseño como la parte de bloques. Otro punto a favor es que para probar la app solo hace falta conectarte a la misma red WIFI que el ordenador con el que trabajes y escanear un código QR y tienes la app en tu móvil y a tiempo real. Todo esto para una persona que comienza a programar le es muy útil para aprender.

El punto débil que le he visto es quizás las limitaciones que tiene a la hora de programar funcionalidades esto reduce la calidad de las aplicaciones que se pueden crear. Crear una aplicación sencilla no será un problema pero a medida que quieres añadir más funcionalidades a la app se hará más costoso o no se podrá realizar. Otro gran problema que le he visto es que no se puede

cambiar la pantalla de inicio fácilmente, es decir la pantalla con la que empiezas a programar es la que aparecerá y si quieres cambiar esta pantalla inicial o el nombre de cualquiera deberás utilizar un editor de texto y descargarte el archivo *.aia*.

Para *Android Studio* el punto fuerte quizás sea que permite desarrollar aplicaciones más elaboradas y puedes retocar muchas más partes del código, es decir permite una mayor explotación y manipulación de tu app, además es posible desarrollar la aplicación para distintos tamaños de pantallas fácilmente.

Como punto débil destacaría que no es tan intuitivo ya que necesitas tener conocimiento de programación y sobre todo en *java* lo que lo hace más complicado para alguien que se esté iniciando en el mundo de la programación y esto te llevará a pasar horas viendo tutoriales y buscando como programar alguna funcionalidad.

5 CONCLUSIONES

Este proyecto surgió como la posibilidad de mejorar mis habilidades y capacidades en un ámbito que no se tiene muy en cuenta en el Grado de Ingeniería Mecánica ya que solo la asignatura de Fundamentos de Informática proporciona una pequeña introducción al mundo de la programación sin llegar a profundizar mucho en ella y sin adquirir muchos conocimientos.

Por eso al realizar el desarrollo de la aplicación con la ayuda de César he alcanzado el objetivo principal por el cuál decidí seleccionar este proyecto y he obtenido amplios conocimientos de programación y he aprendido a desenvolverme en el desarrollo de apps.

Este proceso de mejora de capacidades ha provocado que con el paso del tiempo haya evolucionado el funcionamiento de la aplicación el cual al principio lo concebí como un simple cuestionario de preguntas y respuestas hasta terminar con la evaluación del alumno y poder compartir dicha información con el profesor de la asignatura. Aun así *Engineering Quiz* tiene mucho margen de mejorar y se le pueden añadir más funcionalidades que se comentan en el apartado siguiente.

5.1 MEJORAS

Archivo en Google Drive

Una mejora bastante importante sería tener un archivo en *Google Drive* u otra plataforma que los profesores de las asignaturas podrían manipular y añadir las preguntas y respuestas que ellos considerasen oportunas y así no tener que entrar en el código de la aplicación. Este archivo se descargaría al iniciar la aplicación.

Idiomas

Otro aspecto importante sería el de poder variar el idioma ya que actualmente solo está en español. Esto daría la posibilidad de llegar a un mayor número de usuarios.

Aumento cantidad de asignaturas

Como es un proyecto universitario y el objetivo principal era aprender a programar, solo he incluido cuatro asignaturas de la carrera. Lo idóneo sería que estuvieran todas las asignaturas que se cursan en el Grado.

Añadir otros Grados Universitarios

El funcionamiento actual de *Engineering Quiz* se podría aplicar también a otros Grados de Ingeniería así poder evaluar a una mayor cantidad de alumnos en diferentes ámbitos.

Registro del alumno

Quizás un fallo de la aplicación es que el usuario evaluado es anónimo y solo al enviar el resultado al profesor se podría saber quién es el alumno y aun así es posible que el alumno usase un correo con otro nombre. Por eso una buena solución sería hacer que el alumno se registrase dentro de la aplicación y así su nota se almacenase una vez haya terminado el cuestionario.

6 BIBLIOGRAFÍA

Las fuentes de información utilizadas para este proyecto se describen a continuación:

[1] Statista. Portal de estadísticas.

URL: https://es.statista.com/grafico/8473/android-se-acerca-al-liderazgo-de-los-sistemas-operativos/

[2] Android Developers. Web oficial de Android para desarrolladores.

URL: https://developer.android.com/

[3] Stack Overflow. Comunidad de desarrolladores de diversas plataformas y sistemas.

URL: https://es.stackoverflow.com/

[4] TuAppInventorAndroid. Web sobre Mit App Inventor.

URL: https://www.tuappinvetorandroid.com/2014/06/02/aplicaci%C3%B3n-de-preguntas-tipo-test-con-app-inventor-2/

[5] Youtube. Portal de vídeos.

URL: https://www.youtube.com/watch?v=70qBGgTOnn8

[6] Youtube. Portal de vídeos.

URL: https://www.youtube.com/watch?v=4g1_UH_6VQc

[7] Expansión. Periódico de información.

URL: http://www.expansion.com/economia-digital/companias/2015/12/09/56684be1ca474151018b4590.html

[8] Movilzona. Web de información sobre móviles.

URL: https://www.movilzona.es/2017/04/19/la-guerra-android-vs-ios-en-lo-que-llevamos-de-ano-tiene-un-claro-vencedor/

[9] Pura Vida Apps. Blog de tutoriales.

URL: https://puravidaapps.com/quiz.php

[10] Hermosa Programación. Blog sobre programación.

URL: http://www.hermosaprogramacion.com/2014/08/android-studio-proyecto-en/

[11] Wikipedia. La enciclopedia libre.

URL: https://es.wikipedia.org/wiki/Android_Studio

[12] Wikipedia. La enciclopedia libre.

URL: https://es.wikipedia.org/wiki/App_Inventor

7 ANEXOS

7.1 COMPARATIVA

En este apartado veremos una serie de ilustraciones en las cuales se comparan las correspondientes pantallas de la aplicación realizadas en Android Studio y en Mit App Inventor 2 para ver que a pesar de programar en dos entornos distintos se ha conseguido hacer muy similar la aplicación con unas mínimas diferencias:

Ilustración 39. Comparación pantalla principal en Android Studio (izquierda) y Mit App Inventor 2 (derecha).

Ilustración 40. Comparación pantalla asignaturas en Android Studio (izquierda) y Mit App Inventor 2 (derecha).

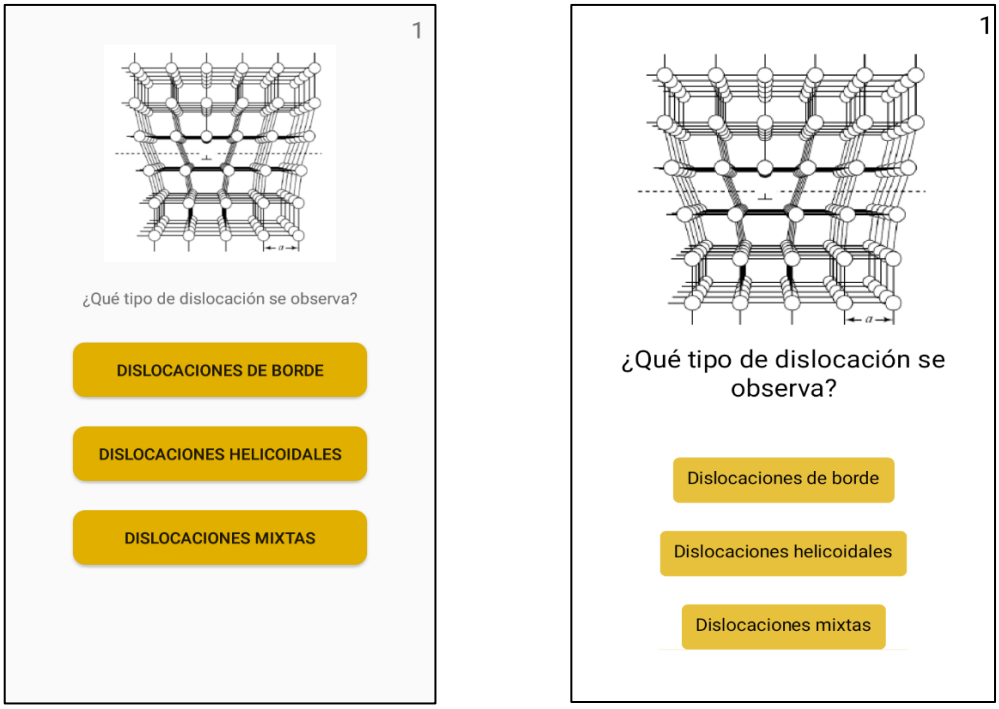


Ilustración 41. Comparación cuestionario en Android Studio (izquierda) y Mit App Inventor 2 (derecha).

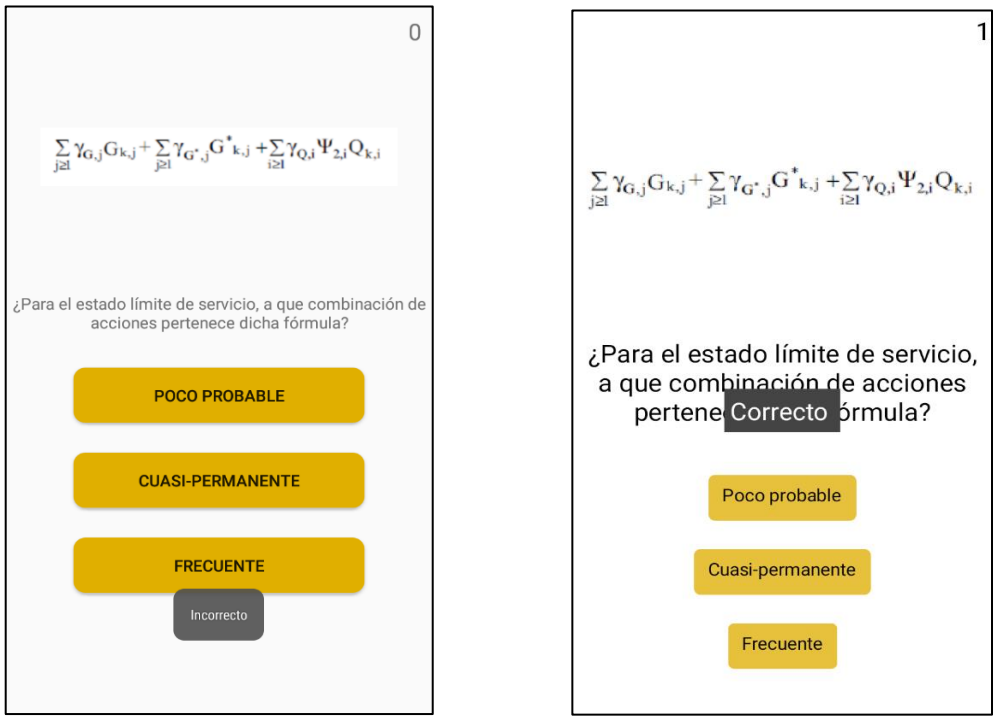


Ilustración 42. Comparación de notificaciones en Android Studio (izquierda) y Mit App Inventor 2 (derecha).

Ilustración 43. Comparación de pantalla de resultados en Android Studio (izquierda) y Mit App Inventor 2 (derecha).

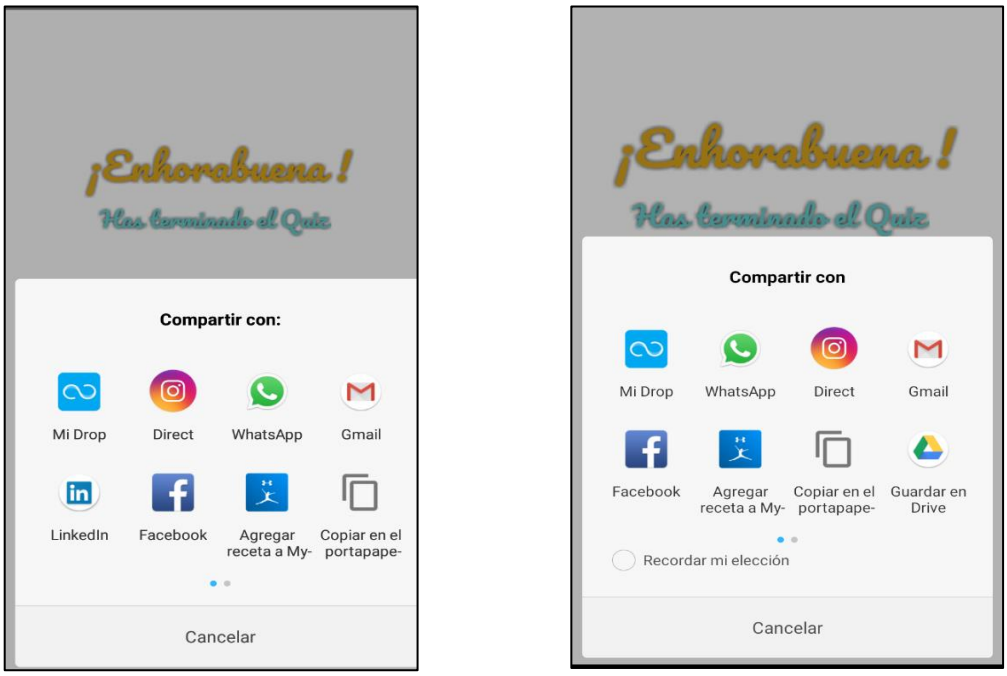


Ilustración 44. Comparación pestaña de compartir en Android Studio (izquierda) y Mit App Inventor 2 (derecha).

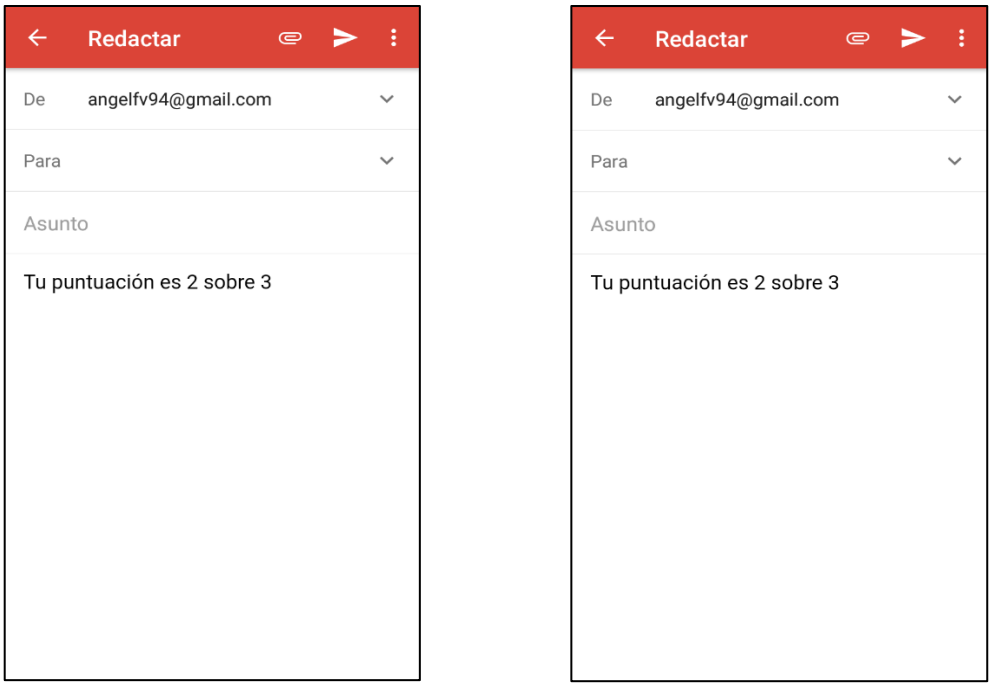


Ilustración 45. Comparación puntuación en Gmail en Android Studio (izquierda) y Mit App Inventor 2 (derecha).

7.2 GLOSARIO

SMARTPHONE: Anglicismo. Teléfono móvil construido sobre una plataforma informática móvil, con capacidad para almacenar datos y realizar actividades asemejándose a una minicomputadora.

ANDROID: Sistema operativo basado en linux para dispositivos móviles.

LINUX: Sistema operativo libre tipo Unix, desarrollado bajo los términos de la Licencia Pública General de GNU.

GNU: Proyecto colaborativo de software libre.

ACTIVIDAD: Cada una de las pantallas de la aplicación móvil.

API: Siglas de Interfaz de Programación de Aplicaciones. Es un conjunto de funciones, métodos o procedimientos que ofrece una biblioteca para ser utilizado por otro software como una capa de abstracción.

SCRIPT: Archivo de órdenes, archivo de procesamiento por lotes o guión es un programa usualmente simple, que por lo regular se almacena en archivos de texto plano.

MULTIPLATAFORMA: Válido para varios sistemas.

ENTORNO DE DESARROLLO: Es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software.

SISTEMA OPERATIVO: Conjunto de órdenes y programas que controlan los procesos básicos de una computadora y permiten el funcionamiento de otros programas

APK: Es la extensión de las aplicaciones Android, variante del Java. Así pues, un .apk es cualquier aplicación que te puedes instalar en tu móvil.

ANDROID MANIFEST: Es el archivo en el que se definen los rasgos principales de una aplicación: nombre, componentes o permisos entre otros.

SDK: Un kit de desarrollo de software o SDK (siglas en inglés de software development kit) es generalmente un conjunto de herramientas de desarrollo que le permite a un programador crear aplicaciones para un sistema concreto

WIDGET: Elemento que puede añadirse al escritorio de nuestro dispositivo Android, ofreciéndonos información y determinadas funcionalidades (según lo que el desarrollador lleve a cabo).