

Desarrollo completo de una aplicación interactiva para dispositivos móviles

Comienzo de la práctica	13-12-2016
Entrega de la práctica	17-01-2017

1 Introducción

En este documento se explica cómo desarrollar completamente un proyecto con Android Studio ©, que incluye las librerías adecuadas para un componente software de teléfonos móviles que ejecuten Android (versión posterior a la 4.4) como sistema operativo, de acuerdo con el esquema general que muestra la siguiente figura:

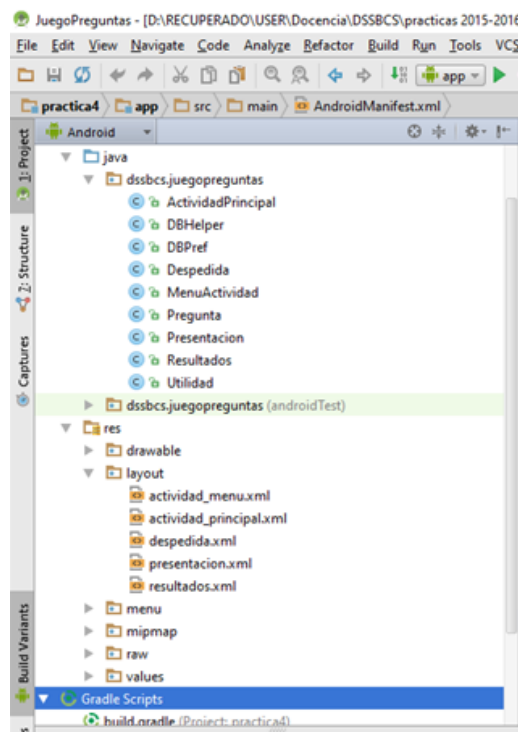


Figura 1: Estructura de carpetas de un proyecto de Android Studio

La estructura de un *app* para poder ser ejecutado en una plataforma Android ha de poseer la siguiente estructura de paquetes (o '*folders*' mostrados en la figura 1):

- *manifestos*: `AndroidManifest.xml` (ver contenido del archivo en la tabla 1)
- *java*: nombre de paquetes y clases–java, que componen la parte controladora de nuestro *app*
- Un prototipo de clase para incluir las pruebas unitarias de validación para nuestra aplicación
- *res*: un paquete que contiene los recursos, expresados como archivos `.xml`, que se utilizarán en el proceso de elaboración del *desplegable* que finalmente constituirá nuestro *app*.

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     package="dssbcs.juegopreguntas" >
4     <application
5         android:allowBackup="true"
6         android:icon="@mipmap/ic_launcher"
7         android:label="@string/app_name"
8         android:name="dssbcs.juegopreguntas.Utilidad">
9         <activity
10             android:name="dssbcs.juegopreguntas.MenuActividad"
11             android:label="@string/app_name" >
12             <intent-filter>
13                 <action android:name="android.intent.action.MAIN" />
14                 <category android:name="android.intent.category.LAUNCHER" />
15             </intent-filter>
16         </activity>
17         <activity
18             android:name="dssbcs.juegopreguntas.ActividadPrincipal"
19             android:label="@string/app_name"
20             android:parentActivityName="dssbcs.juegopreguntas.MenuActividad">
21             <meta-data
22                 android:name="android.support.PARENT_ACTIVITY"
23                 android:value="dssbcs.juegopreguntas.MenuActividad" />
24             </activity>
25         <activity
26             android:name="dssbcs.juegopreguntas.Presentacion"
27             android:label="@string/app_name"
28             android:parentActivityName="dssbcs.juegopreguntas.MenuActividad">
29             <meta-data
30                 android:name="android.support.PARENT_ACTIVITY"
31                 android:value="dssbcs.juegopreguntas.MenuActividad" />
32             </activity>
33         <activity
34             android:name="dssbcs.juegopreguntas.Resultados"
35             android:label="@string/app_name"
36             android:parentActivityName="dssbcs.juegopreguntas.MenuActividad" >
37         </activity>
38         <activity
39             android:name="dssbcs.juegopreguntas.Despedida"
40             android:label="@string/app_name"
41             android:parentActivityName="dssbcs.juegopreguntas.ActividadPrincipal" >
42         </activity>
43     </application>
44 </manifest>

```

Tabla 1: Archivo AndroidManifest.xml

El paquete de recursos más importante se denomina *layout*, que contiene las *vistas* o pantallas a las que dará lugar la ejecución de nuestra aplicación:

- actividad_menu.xml
- actividad_principal.xml
- despedida.xml
- presentacion.xml
- resultados.xml

La configuración completa de nuestro proyecto utilizando *Android Studio* viene descrita en un archivo xml denominado `AndroidManifest.xml`, que se puede ver en la tabla 1.

La carpeta Java de la figura 1 contendrá las clases que constituyen la aplicación del juego que vamos a desarrollar en esta práctica,

- ActividadPrincipal
- Despedida
- MenuActividad
- Pregunta
- Presentacion
- Resultados
- Utilidad

Así como también incluirá las clases “*helper*”, por ejemplo: DBHelper y DBPref, para poder acceder a la base de datos (`database.sqlite`) de preguntas cuyo contenido se guardará como un recurso más en la carpeta: `res/raw/`.

La mencionada carpeta `res` contendrá los diferentes recursos: pantallas, base de datos, imágenes, iconos, ... a los que acceden las clases Java cuando necesitan obtener alguna información desde su entorno o bien modificar el estado de la vista o de la propia aplicación.



Figura 2: Pantalla correspondiente la vista `actividad_menu` de la aplicación

1.1 Las carpetas del proyecto

Vamos a pasar a describir los contenidos de la carpeta `\res` anterior (ver figura 1) por orden de aparición.

`\drawable`

Contiene todas las imágenes del juego en los formatos habituales (*jpg*, *png*, *gif*) y con los tamaños adecuados para ser mostrados en la pantalla de nuestro dispositivo móvil específico, que hemos seleccionado en el menú correspondiente `run` de la barra superior de menús del nuestro *app*.

```

1 <?xml version="1.0" encoding="utf-8"?>
2   <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3       android:orientation="vertical"
4       android:layout_width="match_parent"
5       android:layout_height="wrap_content"
6       style="@style/Background_">
7       <ImageView
8           android:layout_marginTop="100dp"
9           android:layout_width="wrap_content"
10          android:layout_height="wrap_content"
11          android:src="@drawable/areusca"
12          android:layout_gravity="center"/>
13       <TextView
14           style="@style/Texto_introductorio"
15           android:text="@string/welcome1"
16           android:textAllCaps="false"
17           android:fontFamily="sans-serif"
18           android:textSize="35dp"
19           android:textColor="#003366"/>
20       <TextView
21           style="@style/Texto_introductorio"
22           android:text="@string/welcome2"
23           android:id="@+id/go"
24           android:textAllCaps="false"
25           android:fontFamily="sans-serif"
26           android:textSize="35dp"
27           android:textColor="#003366"/>
28       <LinearLayout
29           android:layout_width="match_parent"
30           android:layout_height="match_parent"
31           android:orientation="vertical">
32           <Button
33               style="@style/Botones_menu"
34               android:text="Jugar"
35               android:id="@+id/btn_jugar"
36               android:background="#bada55"
37               android:padding="15dp"
38               android:layout_gravity="center"/>
39           <Button
40               style="@style/Botones_menu"
41               android:text="Resultados"
42               android:id="@+id/btn_resultados"
43               android:background="#7aaee6"
44               android:padding="15dp"
45               android:layout_gravity="center"/>
46           <Button
47               style="@style/Botones_menu"
48               android:text="Otros_juegos"
49               android:id="@+id/btn_ajustes"
50               android:background="#ff88ae"
51               android:padding="15dp"
52               android:layout_gravity="center"/>
53       </LinearLayout>
54   </LinearLayout>

```

Tabla 2: Archivo actividad_menu.xml

\layout

Contiene todos los formatos de diseño que van a utilizar las clases de la aplicación. Tenemos que definir un archivo xml para cada una de las clases que utilizan estos recursos, que en el caso de la aplicación del presente ejemplo se denominan:

actividad_menu.xml, actividad_principal.xml, despedida.xml, presentacion.xml y resultado.xml. Uno de estos archivos de diseño sería el de la vista del menú principal o página de inicio de la aplicación (ver figura 2), que hemos denominado actividad_menu.xml (o MenuActividad en el archivo *Android Manifest*), cuyo contenido se muestra en la tabla 2.

\menu

Aquí se genera automáticamente el siguiente archivo `menu_main.xml`:

```

1 <menu xmlns:android="http://schemas.android.com/apk/res/android"
2     xmlns:tools="http://schemas.android.com/tools" tools:context=".MenuActividad">
3     <item android:id="@+id/menu_inicial"
4         android:title="Menu_inicial"
5         android:orderInCategory="100"
6         android:showAsAction="never" />
7 </menu>

```

\mipmap

Aquí se ubican los distintos iconos o “logos” con que aparecerá nuestra *app* en aquellos dispositivos en los que se despliegue. Por defecto, si no incluimos otra imagen representativa de nuestra *app*, nos creará imágenes con el “marciano verde” de Android, para 4 posibles resoluciones de imagen:

- `ic_launcher.png` (48 px x 48 px) (mdpi)
- `ic_launcher.png` (72 px x 72 px) (hdpi)
- `ic_launcher.png` (96 px x 96 px) (xhdpi)
- `ic_launcher.png` (144 px x 144 px) (xxhdpi)

\raw

Aquí ubicaremos los registros de la base de datos, en formato texto, que llamaremos `database.sqlite`:

```

1 CREATE TABLE `preguntas` (
2     _id INTEGER AUTO INCREMENT PRIMARY KEY,
3     `pregunta` VARCHAR(256),
4     `respuesta_correcta` VARCHAR(32) NOT NULL,
5     `respuesta_incorrecta_1` VARCHAR(32) NOT NULL,
6     `respuesta_incorrecta_2` VARCHAR(32) NOT NULL,
7     `respuesta_incorrecta_3` VARCHAR(32) NOT NULL,
8     `respuesta_incorrecta_4` VARCHAR(32) DEFAULT NULL,
9     `respuesta_incorrecta_5` VARCHAR(32) DEFAULT NULL,
10    `respuesta_incorrecta_6` VARCHAR(32) DEFAULT NULL,
11    `respuesta_incorrecta_7` VARCHAR(32) DEFAULT NULL,
12    `dificultad` CHARACTER NOT NULL DEFAULT 'e',
13    `categoria` CHARACTER NOT NULL DEFAULT 'Z',
14    `pregunta_tipo` VARCHAR(32) NOT NULL
15 );
16
17 -- historia      - A
18 -- geografia     - B
19 -- literatura    - C
20 -- arte         - D
21 -- deportes     - E
22 -- ciencia      - F
23 -- tecnologia   - G
24 -- astronomia   - H
25 -- otras        - Z
26
27 -- incluir algunas preguntas para depuracion
28 INSERT INTO `preguntas` ('pregunta', 'respuesta_correcta', 'respuesta_incorrecta_1',
29 'respuesta_incorrecta_2', 'respuesta_incorrecta_3', 'categoria', 'pregunta_tipo')
30 VALUES ("El agua es esencial para la vida: ¿en qué porcentaje forma parte del peso total
31 del cuerpo humano?", "60-80%", "10-30%", "40-50%", "90-110%", 'G', '0');
32
33 --resto de preguntas

```

\values

En esta carpeta, además del archivo `dimens.xml` que contiene los márgenes de las pantallas por defecto y pantallas con más de 820dp de anchura disponible¹, se incluyen los 2 recursos siguientes:

1. `strings.xml`: contiene las cadenas globales que se utilizan en los archivos de la carpeta `layout` para las distintas pantallas de la aplicación.
2. `styles.xml`: contiene definiciones de los estilos que vamos a utilizar en nuestra aplicación para los diferentes *widgets*, botones, texto en la pantalla de presentación, etc.

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3   <string name="Version">Version 2.0 (c) Qualiastech</string>
4   <string name="app_name">Juego de Agua</string>
5   <string name="menu_inicial">Menu Inicial</string>
6   <string name="despedida">Enhorabuena has respondido a todas las preguntas!</string>
7   <string name="despedida1">Has respondido correctamente: \n</string>
8   <string name="despedida2">Soy ARENUSCA de nuevo, muchas gracias por tu atencion. \n </string>
9   <string name="despedida3"> Me has ayudado muchisimo, espero que tu tambien hayas aprendido!
10  \n</string>
11  <string name="play" >Jugar</string>
12  <string name="stats" >Resultados</string>
13  <string name="settings" >Otros juegos</string>
14  <string name="welcome1" >Hola, \n Soy ARENUSCA y me gustaria retarte a que me demuestres
15  cuanto sabes del agua &#8230;</string>
16  <string name="welcome2">Te atreves?</string>
17  <string name="tip" color="#F62217" textSize="33dp">Entonces, &#8230; esta envenenada el
18  agua del embalse? \n \n No, porque el agua no esta estancada, esta en continua renovacion,
19  le esta entrando y saliendo agua permanentemente .
20  </string>
21 </resources>

```

Tabla 3: Archivo `strings.xml` de la aplicación *Juego de Agua*

```

1 <resources>
2   <style name="Background_"
3     parent="android:Theme.Holo.Light.DarkActionBar">
4     <item name="android:background">@drawable/background_juego</item>
5   </style>
6   <style name="Texto_introductorio">
7     <item name="android:textAppearance">?android:attr/textAppearanceLarge</item>
8     <item name="android:layout_width">match_parent</item>
9     <item name="android:layout_height">wrap_content</item>
10    <item name="android:textSize">35dp</item>
11    <item name="android:textColor">#003366</item>
12  </style>
13  <style name="Botones_menu"
14    parent="@android:style/TextAppearance">
15    <item name="android:layout_width">match_parent</item>
16    <item name="android:layout_height">match_parent</item>
17    <item name="android:layout_marginLeft">100dp</item>
18    <item name="android:layout_marginRight">100dp</item>
19    <item name="android:layout_marginTop">5dp</item>
20    <item name="android:layout_marginBottom">0dp</item>
21    <item name="android:textSize">50dp</item>
22    <item name="android:textStyle">bold</item>
23    <item name="android:textAllCaps">false</item>
24    <item name="android:fontFamily">sans-serif</item>
25    <item name="android:visibility">visible</item>
26  </style>

```

Tabla 4: Archivo `styles.xml` de la aplicación *Juego de Agua*

¹por ejemplo, dispositivos de 7" y 8" en modo apaisado, que poseen sobre 960dp y 1280dp, respectivamente

1.2 Algunas clases de la aplicación

La base de datos anteriormente mencionada se leerá desde la clase-Java DBHelper de nuestro proyecto, utilizando para ello la instrucción `execSQL` dentro de un método de creación.

```

1 import android.content.Context;
2 import android.database.SQLException;
3 import android.database.sqlite.SQLiteDatabase;
4 import android.database.sqlite.SQLiteOpenHelper;
5 import android.util.Log;
6 import java.io.File;
7 import java.io.IOException;
8 import java.util.Scanner;
9 /**
10  * Created by Manuel on 17/12/2015.
11  */
12 public class DBHelper extends SQLiteOpenHelper {
13     public static final String NOMBRE_BD="Juego-preguntasDB";
14     public static final int VERSION_ACTUAL_BD=2;
15     protected SQLiteDatabase db;
16     protected Context ctx;
17
18     public DBHelper(Context context) {
19         super(context, NOMBRE_BD, null, VERSION_ACTUAL_BD);
20         this.ctx = context;
21         this.open();
22     }
23     public void onCreate(SQLiteDatabase sqLiteDatabase) {
24         Log.d("debug", "Creando_la_Base_de_Datos");
25         StringBuilder sb = new StringBuilder();
26
27         Scanner sc = new Scanner( this.ctx.getResources().openRawResource(R.raw.database) );
28         while(sc.hasNextLine()) {
29             sb.append(sc.nextLine());
30             sb.append('\n');
31             if (sb.indexOf(";") != -1) {
32                 sqLiteDatabase.execSQL(sb.toString());
33                 sb.delete(0, sb.capacity());
34             }
35         }
36     }
37     public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i2) {
38         Log.d("debug", "Actualizando_la_base_de_datos_desde_la_version_" + i + "_a_la_version_" + i2);
39     }
40     public void open() {
41         Log.d("debug", "Abriendo_la_base_de_datos");
42         this.db = this.getWritableDatabase();
43     }
44     public void close() {
45         Log.d("debug", "Cerrando_la_base_de_datos");
46         this.db.close();
47     }
48     public void addSampleData() {
49         try {
50             this.db.execSQL("DELETE_FROM `preguntas`");
51             this.db.execSQL("INSERT INTO `preguntas` (`pregunta`,`respuesta_correcta`,
52             respuesta_incorrecta_1`,`respuesta_incorrecta_2`,`respuesta_incorrecta_3`,
53             categoria`,`pregunta_tipo`) +
54             "VALUES(\"El agua es esencial para la vida: En que porcentaje forma parte
55             del peso total del cuerpo humano?`,`60-80%`,`10-30%`,`40-50%`,`
56             90-110%`,`G`,`0`);");
57             //RESTO DE PREGUNTAS
58         }
59         catch (Exception e) {
60             e.printStackTrace();
61             Log.d("debug", e.toString());
62         }
63     }
64 }

```

La clase Java `ActividadPrincipal` obtiene una nueva pregunta de la base de datos llamando al método `getPreguntas(...)` de una instancia de `DBPref` en la siguiente instrucción:

```
1 Cursor cuestiones =
2     this.db.getPreguntas(DBPref.Categoria.TECNOLOGIA, DBPref.Dificultad.FACIL, PREGUNTAS);
```

La clase `DBPref` actúa como un adaptador de la instancia de la base de datos `sqliteDatabase`, que aparece en la clase `DBHelper` para que pueda ser utilizada por `ActividadPrincipal`:

```
1 import android.content.ContentValues;
2 import android.content.Context;
3 import android.database.Cursor;
4 /**
5  * Created by Manuel on 17/12/2015.
6  */
7 public class DBPref extends DBHelper {
8     public static enum Categoria {
9         HISTORIA('A'),
10        //...
11        TECNOLOGIA('G'),
12        OTROS('Z');
13        public final char C;
14        Categoria(char c) {
15            this.C = c;
16        }
17    }
18    public static enum Dificultad {
19        FACIL('e'),
20        MEDIA('m'),
21        DIFICIL('h');
22        public final char D;
23        Dificultad(char d) {
24            this.D = d;
25        }
26    }
27    public DBPref(Context contexto) {
28        super(contexto);
29    }
30
31
32    public void addRegistro(Pregunta pregunta) {
33        ContentValues valores = new ContentValues();
34        valores.put("pregunta", pregunta.getPregunta());
35        valores.put("respuesta_correcta", pregunta.getRespuesta());
36
37        String[] respuestas_erroneas = pregunta.getRespuestasErroneas();
38        for(int i = 0; i < respuestas_erroneas.length; i++) {
39            valores.put("respuesta_incorrecta_" + i, respuestas_erroneas[i]);
40        }
41        db.insert("preguntas", null, valores);
42    }
43    public void addRegistros(Pregunta[] preguntas) {
44        for(int i = 0; i < preguntas.length; i++) {
45            this.addRegistro(preguntas[i]);
46        }
47    }
48    public Cursor getPreguntas(Categoria c, Dificultad d, int limit) {
49        return this.db.rawQuery("SELECT _pregunta, _respuesta_correcta,
50        _respuesta_incorrecta_1, _respuesta_incorrecta_2, " + "
51        _respuesta_incorrecta_3, _pregunta_tipo FROM _preguntas `
52        WHERE _Categoria=_? AND _Dificultad=_?" + "ORDER_BY_RANDOM() _LIMIT_?",
53        new String[]{String.valueOf(c.C), String.valueOf(d.D), String.valueOf(limit)});
54    }
55    //...
56 }
```


1.3 Despliegue de la aplicación móvil

Un app correctamente diseñado, desarrollado y probado ha de ser desplegado en un espacio público para que los potenciales usuarios de la aplicación puedan acceder a ella. El despliegue de una aplicación de este tipo se puede realizar utilizando los servicios de un Cloud o en un SW.

En el caso de apps desarrolladas para dispositivos con Android se puede hacer el despliegue mediante un archivo específico (.apk) en *Play Store* de Google.

La generación del archivo apk y ejecución de la aplicación en dispositivos móviles (teléfonos y tablets) que asuman el sistema operativo Android con un API igual o superior a 21 (versión ≥ 4.4 de Android), se puede realizar en el IDE *Android Studio* directamente, siguiendo el siguiente esquema general:

1. En modo *debug*, generar el *keystore* (para guardar las claves de la aplicación) y el archivo apk
2. Borrar el apk
3. Abrir un terminal o consola (cmd de Windows) y ejecutar la siguiente orden:

```
keytool -list -v -keystore  
C:\<tu directorio para guardar keystores de  
Android Studio>\nombreKey.jks
```
4. Copiar la secuencia hexadecimal completa que aparece con el nombre SHA
5. Dar de alta en *Google Console Developer* el SHA anterior
6. Regresando a nuestro proyecto en Android Studio, ir a *Project Structure/signing* y seleccionar el *keystore* generado en el paso (1)
7. En *Project Structure/Flavors/signing config* seleccionar el *keystore* del paso (6)
8. Cambiar al modo *release*
9. En el menú *Build Project*, realizar un *clean/rebuild* y probar a ejecutar (run)
10. Utilizando el *keystore* del paso (1), generar el denominado *signed apk* que es el desplegable para dispositivos móviles

Créditos

- <http://doc.qt.io/qt-5/deployment-android.html>
- <http://www.vogella.com/tutorials/Android/article.html>
- Android developer homepage <http://developer.android.com/index.html>
- Wiki “Android bug tracker” <https://code.google.com/p/android/issues/list>
- Deploying an Android app <http://docs.twinklmedia.com/m/twinkl-publisher-app/1/243432-deploying-an-android-app-in-google-play>