

UNIVERSIDAD NACIONAL DE INGENIERÍA

Facultad de Ciencias

Escuela Profesional de Ciencia de la Computación

Programación en Dispositivos Móviles
Laboratorio N°3: *Ejecutando nuestra primera
aplicación*



Profesor : Castillo Cara, Manuel

Alumno : Arotoma Bacilio, Bitzer

<https://github.com/BitzerAr/Dispositivos-moviles>

Lima - Perú
3 de abril de 2017

Índice

1. Actividad 1	5
1.1. Crear un nuevo proyecto con los conceptos vistos en la teoría	5
1.1.1. Teclea “PrimeraAplicacion” en el campo Application Name. Android Studio y verifica que se crea una carpeta con este nombre	5
1.1.2. Elige el nombre del paquete “Package Name” siguiendo el estilo de Java conocido como convenio de dominio inverso: com.example.manwest.primeraaplicacion	5
1.1.3. Elige un valor para Minimum SDK. Antes de darle a “Next” haz click en “Help me choose”. Explica que ayuda tan importante nos dá	5
1.1.4. En la siguientes pantalla se pide los nombres para nuestro proyecto. Apuntar cada nombre.	5
1.1.5. Le damos a finalizar y vemos nuestro proyecto. Según los diferentes nombres puestos anteriormente localizarlos en nuestro proyecto.	5
1.2. Ahora ya puedes ejecutar la aplicación en el emulador que creaste en la tercera unidad. Para ello, selecciona tu proyecto “PrimeraAplicacion” con el cursor en el “Package Explorer” (la ventana de la izquierda) y pulsa el icono. Aparecerá el siguiente diálogo para que selecciones un la máquina virtual.	5
2. Actividad 2	6
2.1. Antes de ver los componentes haga un pequeño resumen de la estructura de proyecto que vimos en teoría con “PrimeraAplicacion”. Seguramente tengas la visión de Packages cámbiela a Android para verlos de forma muy resumida los ficheros.	6
2.2. AndroidManifest.xml: En la teoría hemos visto que este fichero identifica el paquete principal y las actividades. Ábralo en el editor y entienda su contenido.	6
2.3. Actividades: Desde la Actividad principal se carga el Layout o el diseño.	6
2.3.1. Lo primero que vemos en nuestra clase principal Java es que hay una herencia. Explique esa clase padre e identifique su paquete de importación explicándolo también.	6
2.3.2. Un aspecto importante es el método setContentView. Este método dibuja el fichero xml en pantalla. Identificar dónde se encuentra este fichero en nuestro proyecto. Fijarse que en esa dirección se encontrarán todas las interfaces gráficas.	7
2.3.3. Explicar los otros métodos que nos encontramos en dicho fichero	7
2.4. Layout: en esta carpeta nos encontramos los xml que definen nuestra interfaz gráfica de la aplicación.	7
2.4.1. En nuestra aplicación tenemos un Relative Layout. Explicar brevemente dos layout más.	7
2.4.2. También hemos aprendido que toda variable string se encuentra en su fichero. Verificar la ruta de estas variables.	7
2.4.3. Aunque ya lo veremos más adelante explicar los otros elementos de este fichero que no se explicaron en teoría.	7

2.5. Strings: otro aspecto a entender es el fichero strings.xml. Cambia la variable llamada en nuestro xml Activity y ejecute la aplicación. Comenta los cambios.	7
3. Actividad 3	8
3.1. Abra la carpeta física Androidd→ SDK→ Extras→ Intel y dentro podrá observar el archivo. Instálelo y verá que mejora notablemente el arranque. .	8
4. Actividad 4	8
4.1. A continuación vamos a crear una segunda actividad. Para crearla creamos una clase pública que herede de AppCompatActivity como hemos estado estudiando hasta ahora. Hay que crearla donde está la otra actividad, es decir, src → main → java → com.XXX.	8
4.2. Llamamos a nuestra clase “SegundaActividad”.	8
4.3. Para que sea considerada Actividad deberemos hacer que herede de AppCompatActivity.	8
4.4. Acordarse de realizar la importación de la librería correspondiente.	8
4.5. Implementar el método onCreate copiándolo de la anterior clase.	9
4.6. Verificar que nos solicita la importación de la librería “Bundle”.	9
4.7. Cambiamos el nombre de la interfaz gráfica, llamándose ahora “activity_segunda_actividad”. Comprobamos que nos aparece un error y es porque no reconoce el Layout, es decir, no existe esta interfaz.	9
4.8. Copiamos y pegamos el archivo xml de Layout para crear nuestra nueva interfaz.	9
4.9. En el archivo “activity_segunda_actividad.xml” cambiamos “tools:context” por “.SegundaActividad”.	9
4.10. Cambiamos el mensaje que muestra el TextView. Para ello tendremos que crear una nueva variable string y asignarle el nuevo valor en el fichero “strings.xml”.	9
4.11. Una vez creada nuestra segunda actividad, recordad que se debe de indicar en el archivo “AndroidManifest.xml”	9
4.11.1. Escribir el siguiente código dentro de la etiqueta “application”	10
4.11.2. Vemos que nos aparece un error, eso es porque no hemos creado dicho string. Procedemos a crearlo con, por ejemplo, el valor “Mi segunda Actividad”	10
4.12. En la pestaña “Design” creamos un button en el fichero “activity_primeras_aplicación.xml” que es el xml que teníamos en un primer momento. Para ello borramos el TextView de “Hello World!”	10
4.13. Vemos como en la pestaña Text se ha creado el código de <code>¡Button!</code> . Además nos aparece una advertencia que es que tenemos que definir dicho string. En esta ocasión vamos a cambiarlo a través de “Properties” en la vista “Design”.	10
4.13.1. Pulsamos sobre Button y buscamos la propiedad “text”.	10
4.13.2. Hacemos click en los puntos suspensivos y luego en New Resource → New string value.	10
4.13.3. Le indicamos el nombre e identificador. En mi caso he puesto “Pulse same”.	10

4.13.4. En la propiedad “onClick” vamos a poner nombre al método nuevo al hacer click sobre “Púlsame”. En mi caso le he llamado “onClick”	11
4.13.5. En el fichero java “PrimeraAplicacion.java” añadimos el siguiente método:	11
4.13.6. Del código anterior explique cada línea su funcionalidad	11
4.13.7. Importamos las clases que nos sugiere.	11
5. Actividad 6	12
5.1. Ejecute la aplicación anterior de manera que esté su máquina virtual activada.	12
5.2. Explique la importancia del ícono para revisar toda la estructura de archivos.	13
5.3. Explique la importancia del ícono para revisar todos los hilos que se están ejecutando.	13
5.4. Explique la importancia del ícono para revisar el uso de memoria.	13
5.5. Explique la importancia del ícono para revisar el uso de Internet.	13
5.6. Explique la importancia del ícono para revisar la carga de trabajo.	13
5.7. Realice una llamada telefónica al equipo virtual.	14
5.8. Envíe un mensaje al dispositivo	14
6. Actividad 7	14
6.1. Hacemos click derecho en la carpeta “build” y pulsamos “Show in Explorer”. Vemos que muestra los archivos.	14
6.2. Abrimos build output que es la carpeta que contiene los archivos que se están ejecutando en el emulador.	15
6.3. Vemos una carpeta “apk” que es el ejecutable de Android. Explique los archivos que nos aparecen.	15
6.4. Para publicar en la tienda de Android deberá estar firmada nuestra aplicación.	15
6.5. En la Barra de Herramientas pulsamos “Build” y luego “Generate Signed APK...” que vemos que nos produce una aplicación certificada.	15
6.6. Para ello creamos un nuevo certificado. Rellenes los datos de dicho archivo. .	16
6.7. Como estoy preparando mi aplicación para ser publicada voy a elegir release. Explique los conceptos que aparecen.	16
6.8. Luego de unos minutos vemos que Gradle compila nuestro paquete con nuestro certificado.	17
6.9. Ahora entramos en la carpeta app de nuestros archivos y vemos el archivo “app-release.apk” que está listo para publicar.	17
6.10. Para publicar debemos entrar en la página de publicadores de Google y entrar en la sección “Distribute” y luego Developer Console.	17
6.11. En este proceso recordemos que hay que pagar 25US\$ que todavía no vamos a pagar.	17
6.12. Una vez dentro se subiría nuestro archivo apk que acabamos de generar. . .	17
7. Bibliografía	17

1. Actividad 1

1.1. Crear un nuevo proyecto con los conceptos vistos en la teoría

Luego de la instalacion del Android Studio, previamente habiendo instalado Java Development Kit (JDK) y Android SDK, se creo la primera aplicación.

1.1.1. Teclea “PrimeraAplicacion” en el campo Application Name. Android Studio y verifica que se crea una carpeta con este nombre

Se dio el nombre a nuestro proyecto para luego crearse la carpeta ”PrimeraAplicacion.” en la ruta AndroidStudioProjects que es la ruta por defecto de Android Studio.

1.1.2. Elige el nombre del paquete “Package Name” siguiendo el estilo de Java conocido como convenio de dominio inverso: com.example.manwest.primeraaplicacion

Se siguio el estilo java al asignar al Package Name, *com.example.bitzer.primeraaplicacion.*

1.1.3. Elige un valor para Minimum SDK. Antes de darle a “Next” haz click en “Help me choose”. Explica que ayuda tan importante nos dá.

”Help me choose” nos brinda una informacion sobre la cantidad de usuarios que tienen determinada version de API. El Minimum SDK que escogi es API 15: Android 4.0.3.

1.1.4. En la siguientes pantalla se pide los nombres para nuestro proyecto. Apuntar cada nombre.

En mi caso deje los nombres por defecto, en Activity Name: MainActivity y Layout Name: activity_main.

1.1.5. Le damos a finalizar y vemos nuestro proyecto. Según los diferentes nombres puestos anteriormente localizarlos en nuestro proyecto.

Al abrir se observan todos los archivos principales creados.

1.2. Ahora ya puedes ejecutar la aplicación en el emulador que creaste en la tercer unidad. Para ello, selecciona tu proyecto “PrimeraAplicacion” con el cursor en el “Package Explorer” (la ventana de la izquierda) y pulsa el icono. Aparecerá el siguiente diálogo para que selecciones un la máquina virtual.

En mi caso usare mi dispositivo movil.

2. Actividad 2

- 2.1.** Antes de ver los componentes haga un pequeño resumen de la estructura de proyecto que vimos en teoría con “PrimeraAplicacion”. Seguramente tengas la visión de Packages cámbiela a Android para verlos de forma muy resumida los ficheros.

En la estructura del proyecto, se observa 3 carpetas fundamentales, en la cual cada una tiene ficheros principales.

Nombraremos las carpetas y sus archivos principales:

- **manifest:** Se encuentra únicamente el AndroidManifest.xml, que se encarga del control de la aplicación.
- **java:** Se encuentran 3 carpetas, donde se encontrara toda la parte lógica realizada en Java, esta el archivo MainActivity.java.
- **res:** Se encuentran los recursos del proyecto, encontramos: drawable (imágenes), layout (parte visual), menu, minimap (íconos) y values(un tipo valores globales).

- 2.2.** **AndroidManifest.xml:** En la teoría hemos visto que este fichero identifica el paquete principal y las actividades. Ábralo en el editor y entienda su contenido.

En la etiqueta principal se define el nombre de espacios de android y en el nombre del package define la aplicación; la etiqueta application contiene los valores por defecto en nuestra aplicación como la opción de hacer backup del sistema, el identificador del recurso, identificador de texto que dará nombre a la aplicación y el identificador al recurso que especifica el tema por defecto de todas las actividades; y por ultimo en la etiqueta activity muestra el controlador que va a interactuar con una pantalla de la interfaz gráfica.

- 2.3.** **Actividades:** Desde la Actividad principal se carga el Layout o el diseño.

- 2.3.1.** Lo primero que vemos en nuestra clase principal Java es que hay una herencia. Explique esa clase padre e identifique su paquete de importación explicándolo también.

La clase padre es AppCompatActivity , que soporta la utilización de la Action Bar en nuestras aplicaciones (barra del menú) es una extensión de Activity. En el se encuentra el método onCreate(), donde el usuario interactúa con las actividades mediante el método setContentView que para el usuario se presenta como actividades.

El paquete de importación es *android.support.v7.app.AppCompatActivity* son librerías que proporcionan configuraciones específicas y pueden ser incluidas en cualquier aplicación independiente de la versión.

- 2.3.2.** Un aspecto importante es el método `setContentView`. Este método dibuja el fichero xml en pantalla. Identificar dónde se encuentra este fichero en nuestro proyecto. Fijarse que en esa dirección se encontrarán todas las interfaces gráficas.

Este fichero esta especificado en `/src/main/res/layout/activity_main.xml`, la llamada a este indica a Android que debe establecer como interfaz gráfica de dicha actividad que esta definida en la ruta indicada, además se encuentran otros aspectos imágenes, color, dimensiones.

- 2.3.3. Explicar los otros métodos que nos encontramos en dicho fichero**

Encontramos el método `onCreate`, llamado cada vez que se cargue la actividad, en su interior se encuentra `setContentView` que ya se explico en el item anterior.

- 2.4. Layout: en esta carpeta nos encontramos los xml que definen nuestra interfaz gráfica de la aplicación.**

- 2.4.1. En nuestra aplicación tenemos un Relative Layout. Explicar brevemente dos layout más.**

- Table Layout: Dispone los elementos de forma tabular se utiliza la etiqueta `<Table-Row>` para insertar nueva linea.
- Linear Layout: `LinearLayout`, los elementos están contenidos en el mismo se alinearán linealmente horizontal o verticalmente.

- 2.4.2. También hemos aprendido que toda variable string se encuentra en su fichero. Verificar la ruta de estas variables.**

La ruta es `R.values.strings.xml` que equivale a `/res/values/strings.xml`.

- 2.4.3. Aunque ya lo veremos más adelante explicar los otros elementos de este fichero que no se explicaron en teoría.**

En la carpeta Layout en el fichero `activity_main.xml` se muestra la etiqueta `TextView` la cual es un cuadro de texto para el usuario la cual es modificable.

- 2.5. Strings: otro aspecto a entender es el fichero strings.xml. Cambia la variable llamada en nuestro xml Activity y ejecute la aplicación. Comenta los cambios.**

Las variables `strings.xml` son llamadas en el `AndroidManifest.xml` como `@string/app_name`, al cambiar de nombre las llamadas a un nombre que no existe generará errores en el `AndroidManifest.xml`. **No resource found that matches the given name(at 'label' with value '@string/app_nombre')**

3. Actividad 3

- 3.1. Abra la carpeta física Android → SDK → Extras → Intel y dentro podrá observar el archivo. Instálelo y verá que mejora notablemente el arranque.

No compatible para linux.

4. Actividad 4

- 4.1. A continuación vamos a crear una segunda actividad. Para crearla creamos una clase pública que herede de AppCompatActivity como hemos estado estudiando hasta ahora. Hay que crearla donde está la otra actividad, es decir, src → main → java → com.XXX.

4.2. Llamamos a nuestra clase “SegundaActividad”.

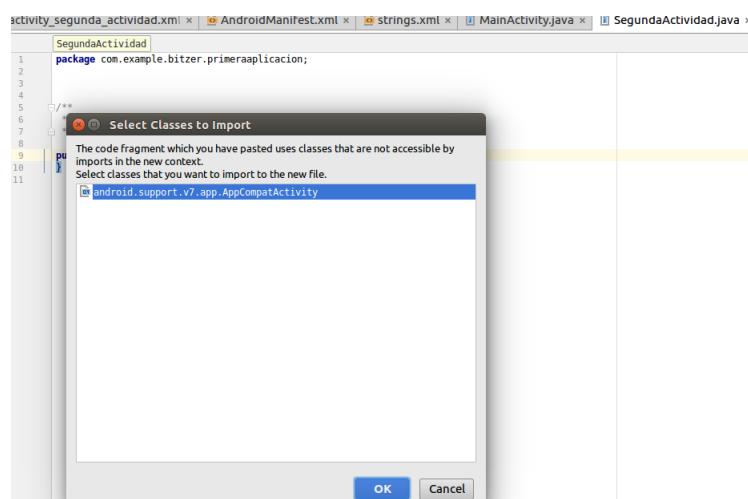
Inicialmente se crea la clase sin importar ninguna librería.

4.3. Para que sea considerada Actividad deberemos hacer que herede de AppCompatActivity.

Al hacer el extends AppCompatActivity, nos manda una alerta para importar librería.

4.4. Acordarse de realizar la importación de la librería correspondiente.

La importación de la librería Android Studio nos solicita importarla mandando una alerta.



- 4.5.** Implementar el método `onCreate` copiándolo de la anterior clase.
- 4.6.** Verificar que nos solicita la importación de la librería “`Bundle`”.

Al igual que el caso anterior nos manda una alerta para importar dicha librería.

- 4.7.** Cambiamos el nombre de la interfaz gráfica, llamándose ahora “`activity_segunda_actividad`”. Comprobamos que nos aparece un error y es porque no reconoce el Layout, es decir, no existe esta interfaz.

Se sombra de rojo y pide crear el layout llamado `activity_segunda_actividad.xml`.

```
▼ /home/bitzer/AndroidStudioProjects/PrimeraAplicacion/app/src/main/java/com/example/bitzer/primeraaplicacion/SegundaActividad.java
  ! error: cannot find symbol variable activity_segunda_actividad
Execution failed for task ':app:compileDebugJavaWithJavac'.
  > Compilation failed; see the compiler error output for details.
```

- 4.8.** Copiamos y pegamos el archivo xml de Layout para crear nuestra nueva interfaz.
- 4.9.** En el archivo “`activity_segunda_actividad.xml`” cambiamos “`tools:context`” por “`.SegundaActividad`”.

`tools:context=com.example.bitzer.primeraaplicacion.SegundaActividad`

- 4.10.** Cambiamos el mensaje que muestra el `TextView`. Para ello tendremos que crear una nueva variable string y asignarle el nuevo valor en el fichero “`strings.xml`”.

Creamos la nueva variable en `/res/values/strings.xml` y para invocarlos usamos `android:text="@string/app_nombre`, la variable que cree es
`<string name = "app_nombre" > SegundaActividad </string >`.

- 4.11.** Una vez creada nuestra segunda actividad, recordad que se debe de indicar en el archivo “`AndroidManifest.xml`”.

Se indica aqui ya que es aqui donde se configura la aplicación, en este caso se agrega la actividad.

4.11.1. Escribir el siguiente código dentro de la etiqueta “application”.

```
<activity
    android:name=".SegundaActividad"
    android:label="@string/seg_actividad" >
</activity >
```

4.11.2. Vemos que nos aparece un error, eso es porque no hemos creado dicho string. Procedemos a crearlo con, por ejemplo, el valor “Mi segunda Actividad”

```
<string name="seg_actividad" >Mi Segunda Actividad</string>
```

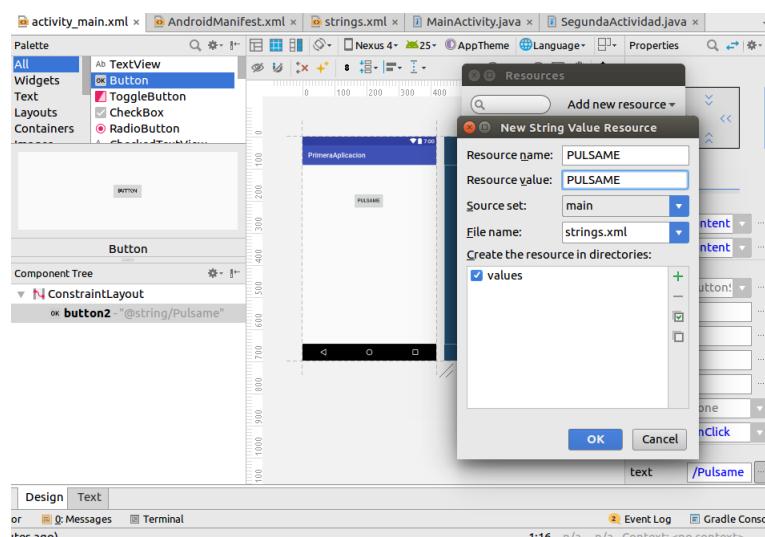
4.12. En la pestaña “Design” creamos un button en el fichero “activity_primeras_aplicación.xml” que es el xml que teníamos en un primer momento. Para ello borramos el TextView de “Hello World!”

4.13. Vemos como en la pestaña Text se ha creado el código de `¡Button!`. Además nos aparece una advertencia que es que tenemos que definir dicho string. En esta ocasión vamos a cambiarlo a través de “Properties” en la vista “Design”.

4.13.1. Pulsamos sobre Button y buscamos la propiedad “text”.

4.13.2. Hacemos click en los puntos suspensivos y luego en New Resource → New string value.

4.13.3. Le indicamos el nombre e identificador. En mi caso he puesto “Pulsame”.



4.13.4. En la propiedad “onClick” vamos a poner nombre al método nuevo al hacer click sobre “Púlsame”. En mi caso le he llamado “onClick”

4.13.5. En el fichero java “PrimeraAplicacion.java” añadimos el siguiente método:

```
public void onClick(View view){  
    Intent intent = new Intent(this, SegundaActividad.class);  
    startActivity(intent);  
    finish()  
}
```

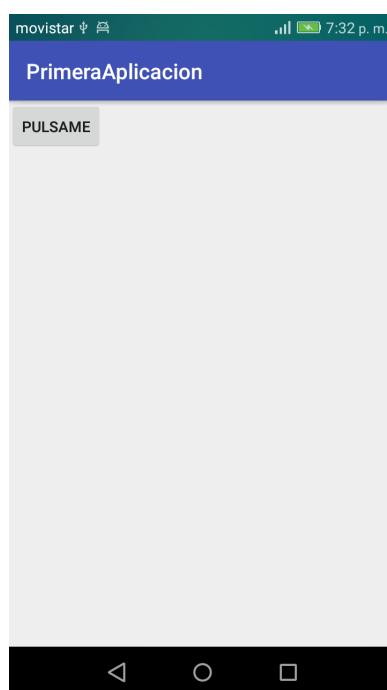
Esto es Para poder observar lo realizado an el item anterior.

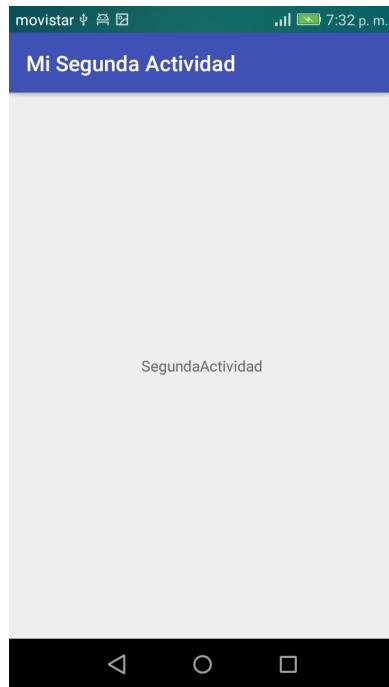
4.13.6. Del código anterior explique cada línea su funcionalidad

- **Public void onClick(View view)**, la declaración de la función teniendo como parámetro la segunda vista creada anteriormente.
- **Intent intent = new intent(this,SegundaActividad.class);**, Crea el objeto que toma como parámetros en su método constructor un contexto que viene a ser nuestra actividad en la que estamos y un .class que sería la clase que vamos a lanzar; define la intención de pasar entre actividades.
- **startActivity(intent);** Inicia el intent que cambia de activity.
- **Finish();** cierra esta actividad, la actividad que lanzo a la segunda.

4.13.7. Importamos las clases que nos sugiere.

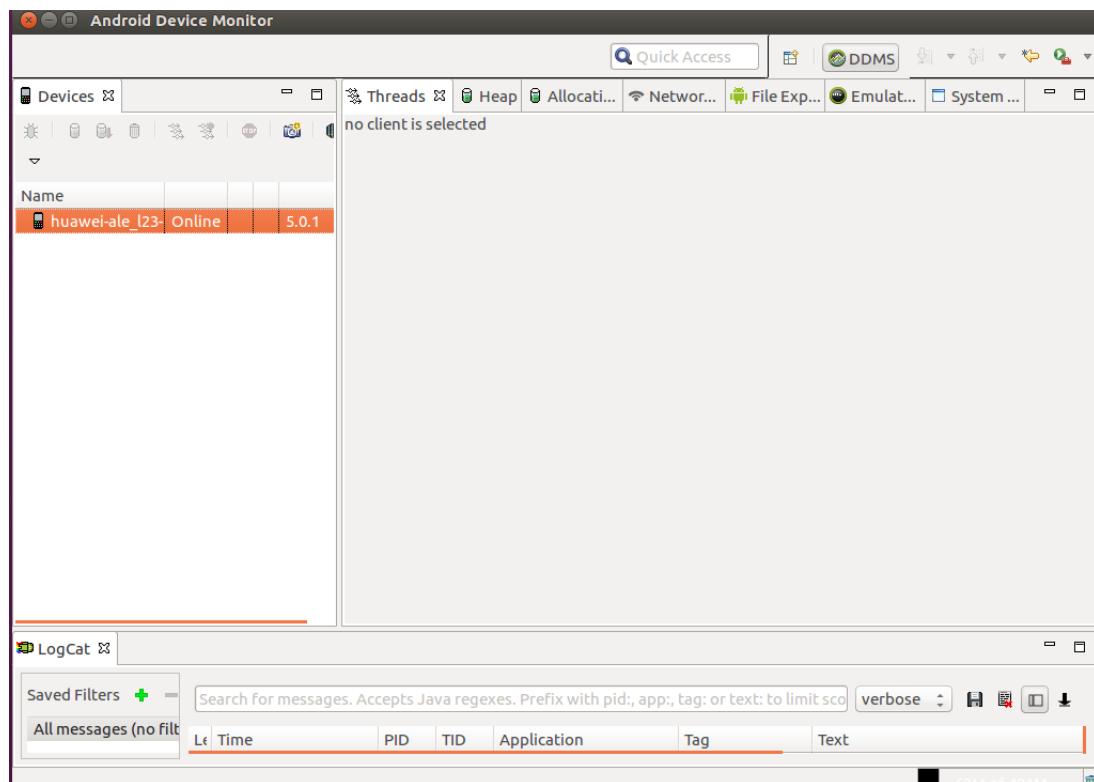
Se importaron de forma automática.





5. Actividad 6

5.1. Ejecute la aplicación anterior de manera que esté su máquina virtual activada.



5.2. Explique la importancia del icono para revisar toda la estructura de archivos.

Android device monitor es una herramienta independiente ya que no requiere de la instalación de un entorno, esta proporciona herramientas de depuración y análisis de la aplicación.

5.3. Explique la importancia del icono para revisar todos los hilos que se están ejecutando.

Un hilo viene a ser la ejecución asociada a una aplicación, el objetivo de esto es realizar múltiples tareas a la vez para dar esa percepción al usuario es una unidad de ejecución asociada a una aplicación. Es la estructura de la programación concurrente, la cual tiene como objetivo dar la percepción al usuario que el sistema que ejecuta realiza múltiples tareas a la vez, la importancia es monitorizar las operaciones realizadas en cada hilo.

5.4. Explique la importancia del icono para revisar el uso de memoria.

La importancia es realizar un seguimiento de los objetos que se están asignando a la memoria, este seguimiento es en tiempo real, la información que se proporciona tras este seguimiento tiene por finalidad evaluar el uso de memoria.

5.5. Explique la importancia del icono para revisar el uso de Internet.

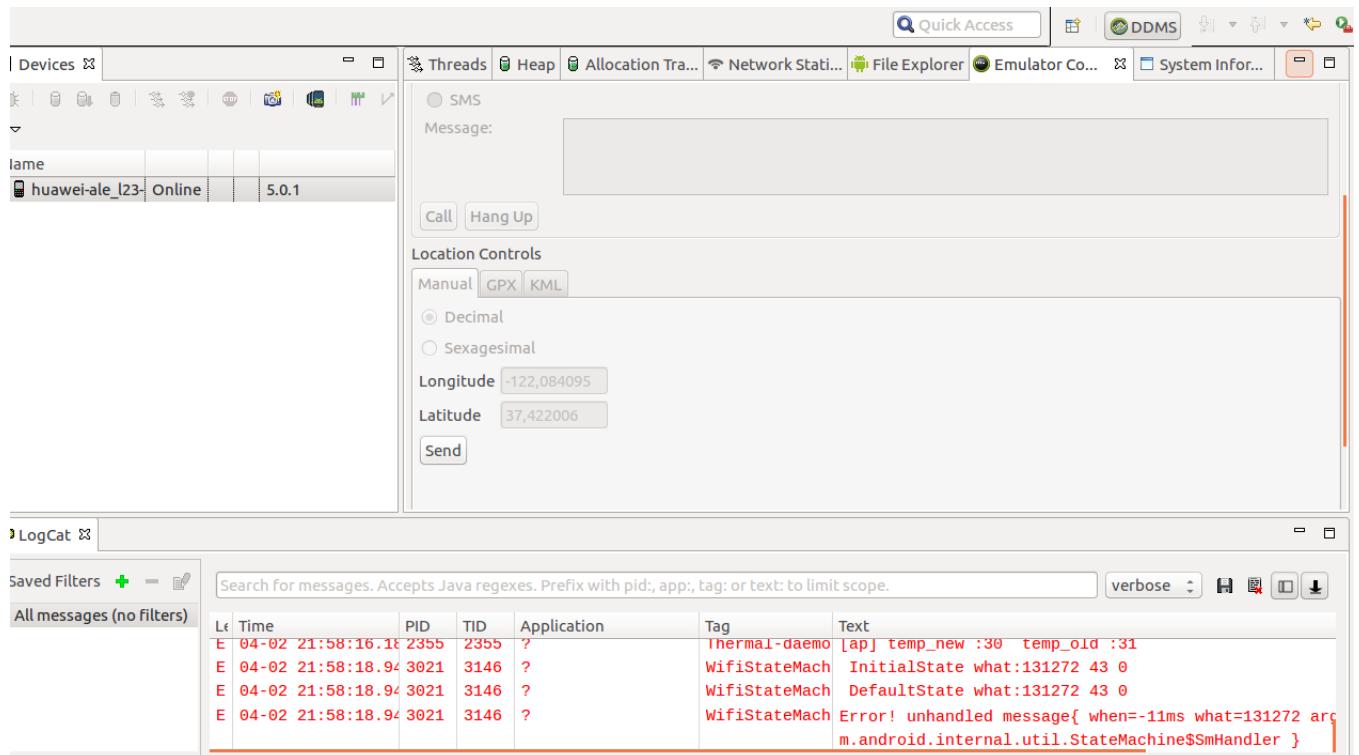
El uso de esta herramienta, puede controlar cómo y cuándo sus transferencias de datos de aplicaciones y optimizar el código subyacente de manera apropiada, tambien distingue entre los diferentes

5.6. Explique la importancia del icono para revisar la carga de trabajo.

La importancia es de supervisar las asignaciones para cada uno de los recursos de modo que puede equilibrar sus cargas de trabajo de forma eficaz.

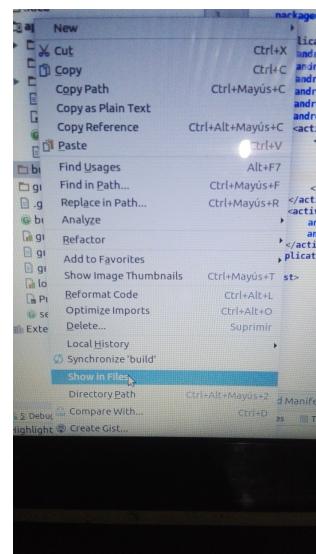
5.7. Realice una llamada telefónica al equipo virtual.

5.8. Envíe un mensaje al dispositivo

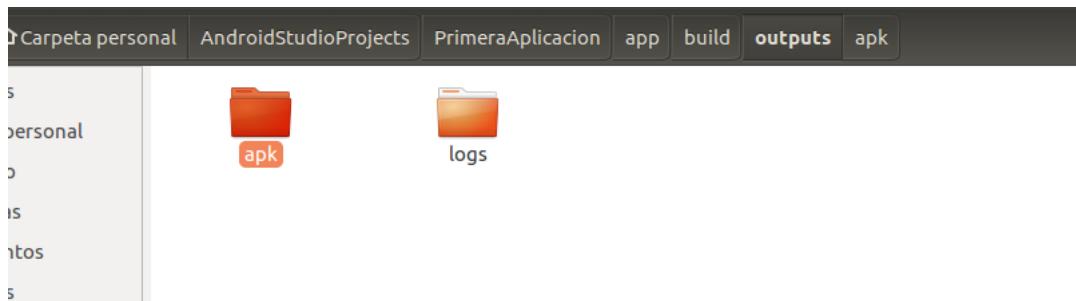


6. Actividad 7

6.1. Hacemos click derecho en la carpeta “build” y pulsamos “Show in Explorer”. Vemos que muestra los archivos.



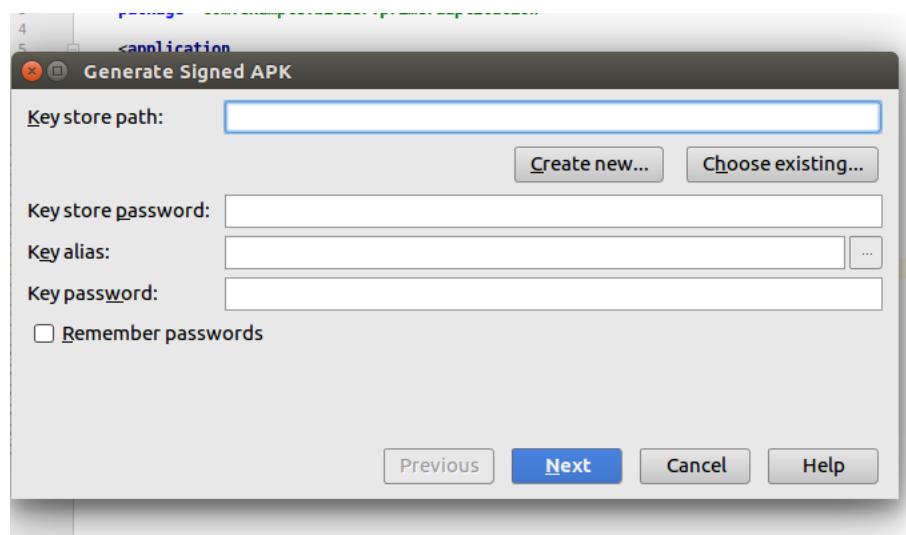
- 6.2. Abrimos build output que es la carpeta que contiene los archivos que se están ejecutando en el emulador.



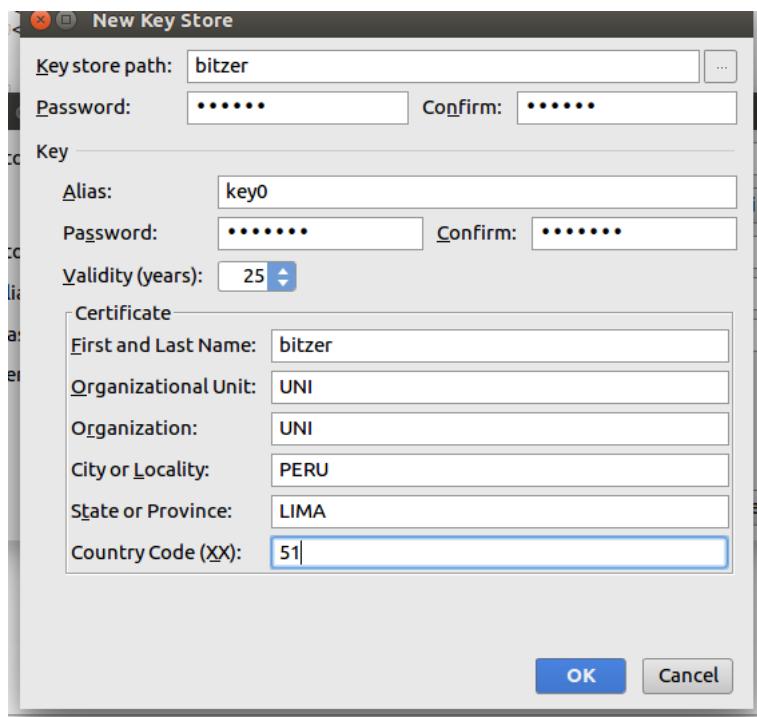
- 6.3. Vemos una carpeta “apk” que es el ejecutable de Android. Explique los archivos que nos aparecen.

En la carpeta apk hay un archivo app-debug.apk, un .apk es el formato de archivo utilizado para la instalación de software en el sistema operativo Android.

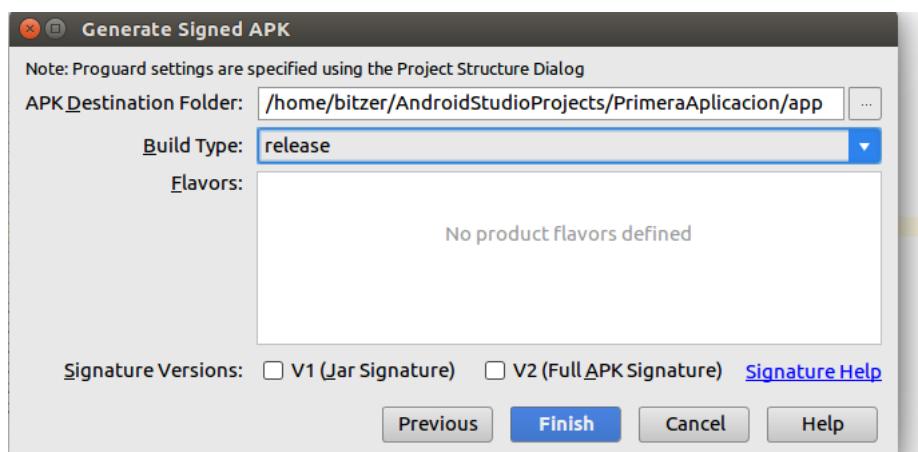
- 6.4. Para publicar en la tienda de Android deberá estar firmada nuestra aplicación.
- 6.5. En la Barra de Herramientas pulsamos “Build” y luego “Generate Signed APK...” que vemos que nos produce una aplicación certificada.



6.6. Para ello creamos un nuevo certificado. Rellenes los datos de dicho archivo.



6.7. Como estoy preparando mi aplicación para ser publicada voy a elegir release. Explique los conceptos que aparecen.



Selecciona un destino para el APK firmado, luego el tipo de compilación(jar o full apk); cuando se complete el proceso encontrarás el APK firmado para poder distribuirlo.

- 6.8.** Luego de unos minutos vemos que Gradle compila nuestro paquete con nuestro certificado.
- 6.9.** Ahora entramos en la carpeta app de nuestros archivos y vemos el archivo “app-release.apk” que está listo para publicar.



- 6.10.** Para publicar debemos entrar en la página de publicadores de Google y entrar en la sección “Distribute” y luego Developer Console.
- 6.11.** En este proceso recordemos que hay que pagar 25US\$ que todavía no vamos a pagar.
- 6.12.** Una vez dentro se subiría nuestro archivo apk que acabamos de generar.

7. Bibliografía

- <https://developer.android.com/topic/libraries/support-library/setup.html?hl=es>
- <http://www.sgoliver.net/blog/desarrollando-una-aplicacion-android-sencilla-android-studio/>
- <https://developer.android.com/guide/topics/ui/layout/relative.html>
- <http://programandolo.blogspot.pe/2013/09/el-archivo-androidmanifestxml.html>
- <https://developer.android.com/studio/run/managing-avds.html?hl=es-419>