

1. Lee el siguiente artículo:

<http://www.zdnet.com/article/debunking-four-myths-about-android-google-and-open-source/>

Contesta las siguientes preguntas:

a) ¿Sobre qué mitos habla el artículo?

El artículo habla de cuatro mitos: el primero es de si los servicios de Android define google, el segundo mito es de si Android no de código abierto, el tercero de si google cobra tarifas de licencia para los servicios móviles de google y el cuarto mito sobre si Android no es Linux

b) ¿Por qué existe la duda si Android es código abierto?

Porque para Richard M. Stallman, Android no permite que los usuarios sean libre

c) ¿Cuál es tu conclusión sobre este artículo?

Todos los mitos que se mencionan en este artículo son falsos. Android es de código abierto y se puede utilizar en dispositivos, aplicaciones y servicios.

2. Android Studio

Descarga e instala Android Studio, el IDE oficial para el desarrollo de aplicaciones Android. La URL de descarga de Android Studio es:

<https://developer.android.com/studio/index.html>

Par la instalación de Android Studio en Ubuntu se emplea los siguientes comandos:

****INSTRUCCIONES****

//instalacion de JDK

sudo apt install default-jre

sudo apt install openjdk-8-jre-headless

//es opcional

sudo apt install openjdk-9-jre-headless

//instalacion de ADB y FASTBOOT

sudo apt install android-tools-adb

sudo apt install adb

sudo apt install android-tools-fastboot

sudo apt install fastboot

//librerias necesarias para android studio

sudo apt-get install libc6:i386 libncurses5:i386 libstdc++6:i386 lib32z1 libbz2-1.0:i386

//Descomprimiendo Android Studio y copiandolo a la carpeta opt

sudo unzip android-studio-ide-162.3934792-linux.zip -d /opt

//moviendonos a la carpeta de android studio

cd /opt/android-studio/bin

//ejecutando android studio

./studio.sh

El paquete descargado incluye todas las herramientas necesarias para el desarrollo de aplicaciones:

- ☐ Android Studio IDE
- ☐ Herramientas Android del SDK
- ☐ Herramientas de la plataforma Android
- ☐ La última versión de Android
- ☐ La última imagen del sistema para el emulador

Contesta lo siguiente:

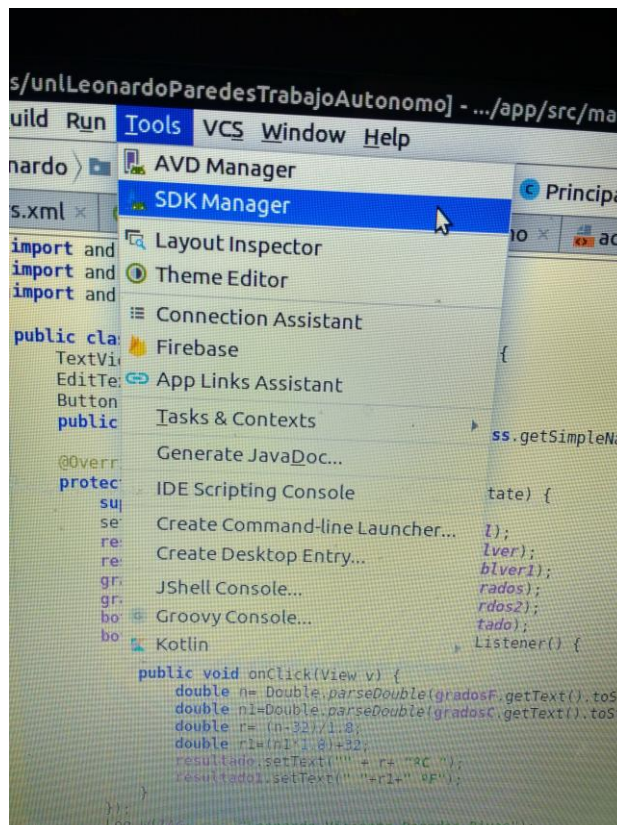
d) ¿En qué sistema operativo has instalado el IDE Android Studio? Si has tenido algún

En el Sistema Operativo de Ubuntu 16.04

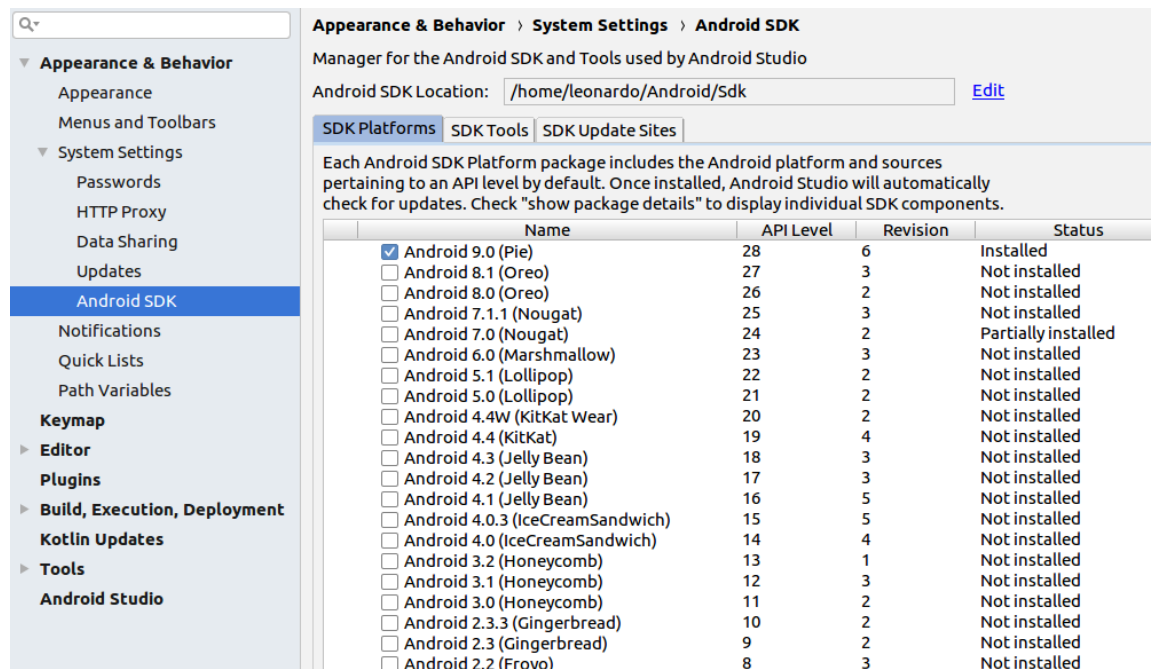
“Incidencias TA1”.

e) Periódicamente aparecen nuevas versiones de Android y con ellas nuevas versiones del SDK oficial de Android. Si hubiera un nuevo cambio de versión, describe cómo acceder a la ventana que permite actualizar la versión más reciente del SDK y como se llama esta herramienta.

Para poder actualizar el SDK en Android Studio le damos a la opción **Tools > SDK Manager**,



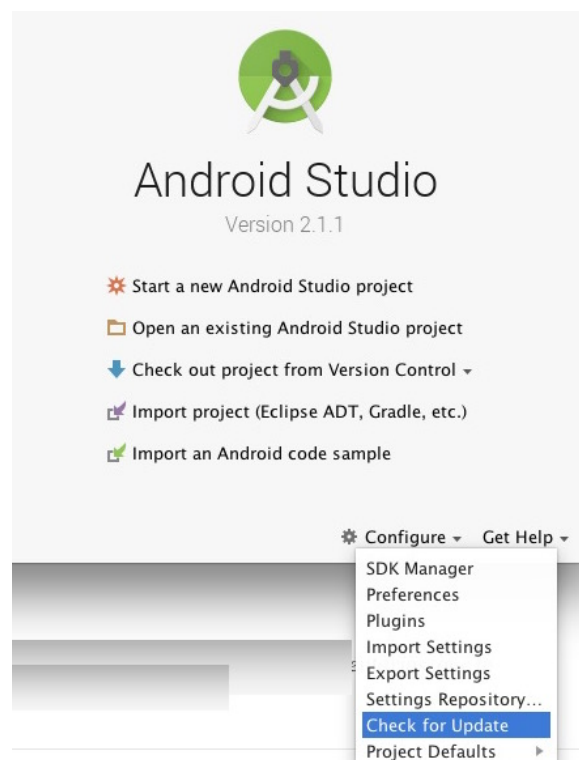
posteriormente nos aparece una ventana llamada Default Settings. Dentro de esta ventana hay una opción llamada android SDK. Esta opción es la que permitirá actualizar el sdk



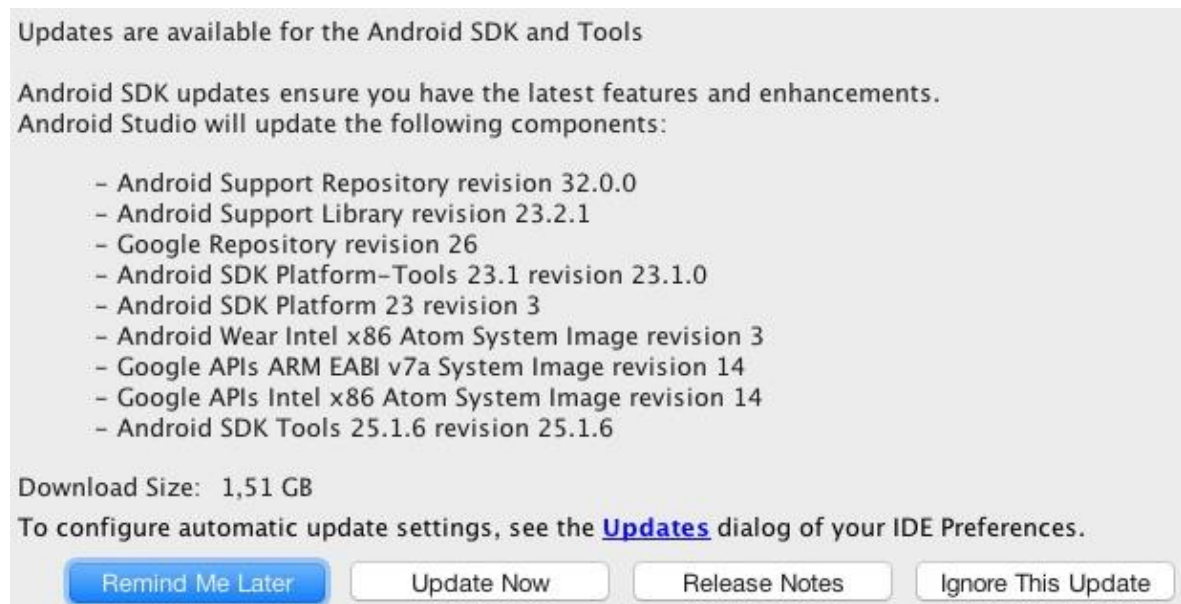
f) Instala Google USB driver (si es Windows), Intel x86 Emulator Accelerator (si el procesador del ordenador es Intel) y Google Play Services. Adjunta una captura de pantalla y explica para que sirven cada una.

g) Android Studio también se va actualizando cada cierto tiempo. Describe que pasos se deben seguir para obtener la última actualización y donde podríamos ver qué novedades incorpora.

Si se abre Android Studio (con todos los proyectos cerrados), se verá que en la parte inferior dice "Configure". Una vez dentro, en Configure > Check for Update, Android Studio se encargará de ver si hay una nueva actualización.



Si hay una nueva versión disponible, nos aparecerá para descargar, por lo que solo hay que seguir los pasos haciendo clic en “**Update Now**” como vemos en la siguiente imagen.



3. Emulador

En el ciclo de desarrollo de las aplicaciones se codifica, compila y ejecuta en un entorno de pruebas. Las apps se pueden testear en un dispositivo físico o en un Android Virtual Device (AVD). Los AVDs se administran mediante la herramienta “Android Virtual Device Manager” que puede ser utilizada como una herramienta de línea de comandos o mediante una interfaz gráfica desde el propio IDE. Mi recomendación es utilizar el dispositivo para el que se está desarrollando la app, caso contrario, configurar con una especificación de hardware que coincida con la del dispositivo físico que se quiere simular.

h) Mediante Android Studio, crea un dispositivo virtual Android. Este dispositivo virtual debe tener las siguientes características:

- ☐ Última versión Android disponible con la API de Google
- ☐ La cámara delantera debe ser la de la webcam de su computador
- ☐ Almacenamiento interno de 400MB
- ☐ SD card de 256MB

Indica el tipo de dispositivo, versión del sistema operativo, número de API, memoria, etc) y adjunta capturas de pantalla de las configuraciones. Una vez configurado, arranca el emulador y escribe en el debate “Incidencias TA1” si has tenido algún problema durante el proceso, caso contrario, adjunta una captura de pantalla del emulador funcionando correctamente. Si tienes problemas de hardware también puedes utilizar Genymotion como alternativa.

i) Opcionalmente configura los drivers de un dispositivo móvil con sistema operativo Android para que el IDE lo reconozca cuando se conecte al ordenador. Indica que dispositivo has conectado, adjunta capturas de pantalla y/o fotografías y explica los pasos que has seguido.

Para que el IDE lo reconozca en primer lugar se necesita las opciones de desarrollador de nuestro celular, para ello vamos a acceder a lo que son los **ajustes > acerca del teléfono > y en la opción que dice número de compilación se le va a dar 7 veces click**. Una vez hecho eso nos habilitara una opción en los ajustes llamada opciones de desarrollo, cuya opción debe estar activa. Dentro de opciones de desarrollo debemos habilitar la depuración por USB. Una vez hecho esto se procede a conectar el celular, posteriormente nos sale un mensaje en el celular que nos dice si permitimos la depuración por USB, le damos a aceptar y a continuación hacemos correr la aplicación en donde se va a seleccionar el dispositivo móvil donde se va a ejecutar. En este paso es necesario instalar los drivers para que Android Studio reconozca nuestro dispositivo para poder emularlo. Una vez hecho todos estos pasos ya se puede ejecutar cualquier aplicación en nuestro dispositivo celular

4. Depuración de la aplicación

j) ¿Cómo se ejecuta una app en modo depuración?

Existen dos formas de realizarlo. La primera es dándole en el menú a la opción Run > Debugg app y la otra forma es dándole click  al botón señalado y para que esto funcione

nuestro dispositivo celular debe estar conectado a nuestra computadora

k) ¿Cómo se podría conectar en modo depuración una aplicación que ha sido lanzada previamente en modo Run?

Deteniendo la aplicación y dándole al boto Debugg app mencionado anteriormente

l) ¿Qué son y para qué sirven los breakpoints?

Los Breakpoints son varios tipos de puntos de interrupción que activan diferentes acciones de depuración. El más común es un punto de interrupción de línea que pausa la ejecución de tu app en una línea de código específica. Mientras la app está pausada, puedes examinar variables, evaluar expresiones y luego continuar la ejecución línea por línea para determinar las causas de los errores en el tiempo de ejecución.

m) Ejecuta la aplicación con un breakpoint en el código. Una vez detenida explica para

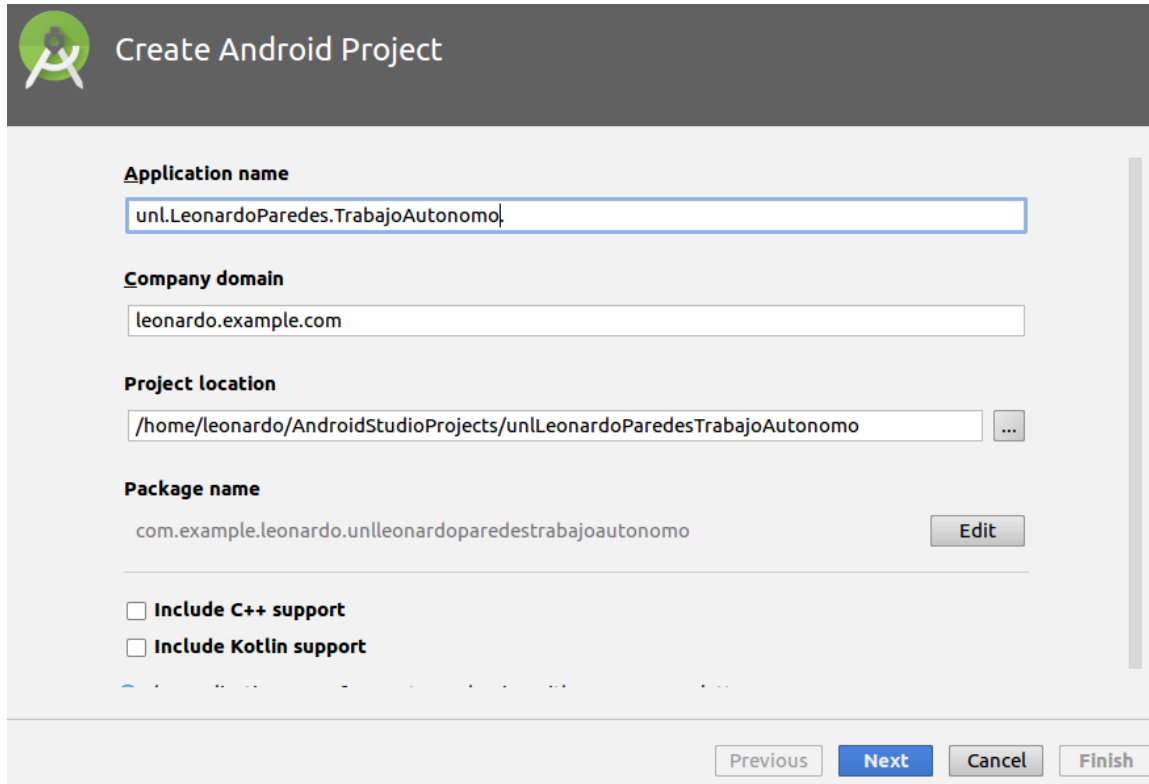


que sirve cada uno de estos botones y qué comando por teclado tienen asignados.

- 1) Ejecutar app con cobertura
- 2) Detener app
- 3) Saltar
- 4) Entrar
- 5) Puntos de interrupción
- 6) Silenciar punto de interrupción

5. Primera app Android

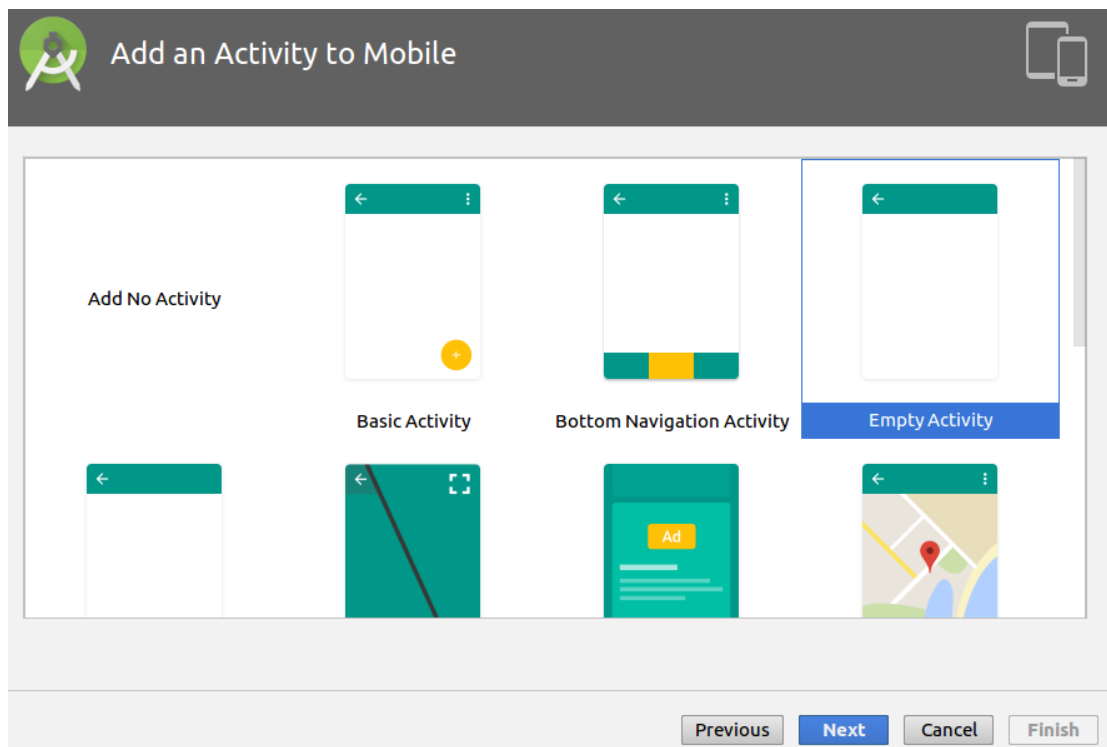
n) Crea un proyecto Android utilizando Android Studio (tipo de proyecto “Empty Activity”, API 16). El nombre de paquete de Java debe ser unl.nombre1apellido1.nombreApp. Explica todos los pasos y parámetros seleccionados para la creación del proyecto y describe el porqué de tu elección.



The screenshot shows the 'Create Android Project' dialog in Android Studio. The title bar includes the Android Studio logo and the text 'Create Android Project'. The dialog contains several input fields and checkboxes:

- Application name:** A text field containing 'unl.LeonardoParedes.TrabajoAutonomo'.
- Company domain:** A text field containing 'leonardo.example.com'.
- Project location:** A text field containing '/home/leonardo/AndroidStudioProjects/unlLeonardoParedesTrabajoAutonomo' with a browse button (three dots) to its right.
- Package name:** A text field containing 'com.example.leonardo.unlleonardoparedestrabajoautonomo' with an 'Edit' button to its right.
- Include C++ support:** A checkbox that is unchecked.
- Include Kotlin support:** A checkbox that is unchecked.

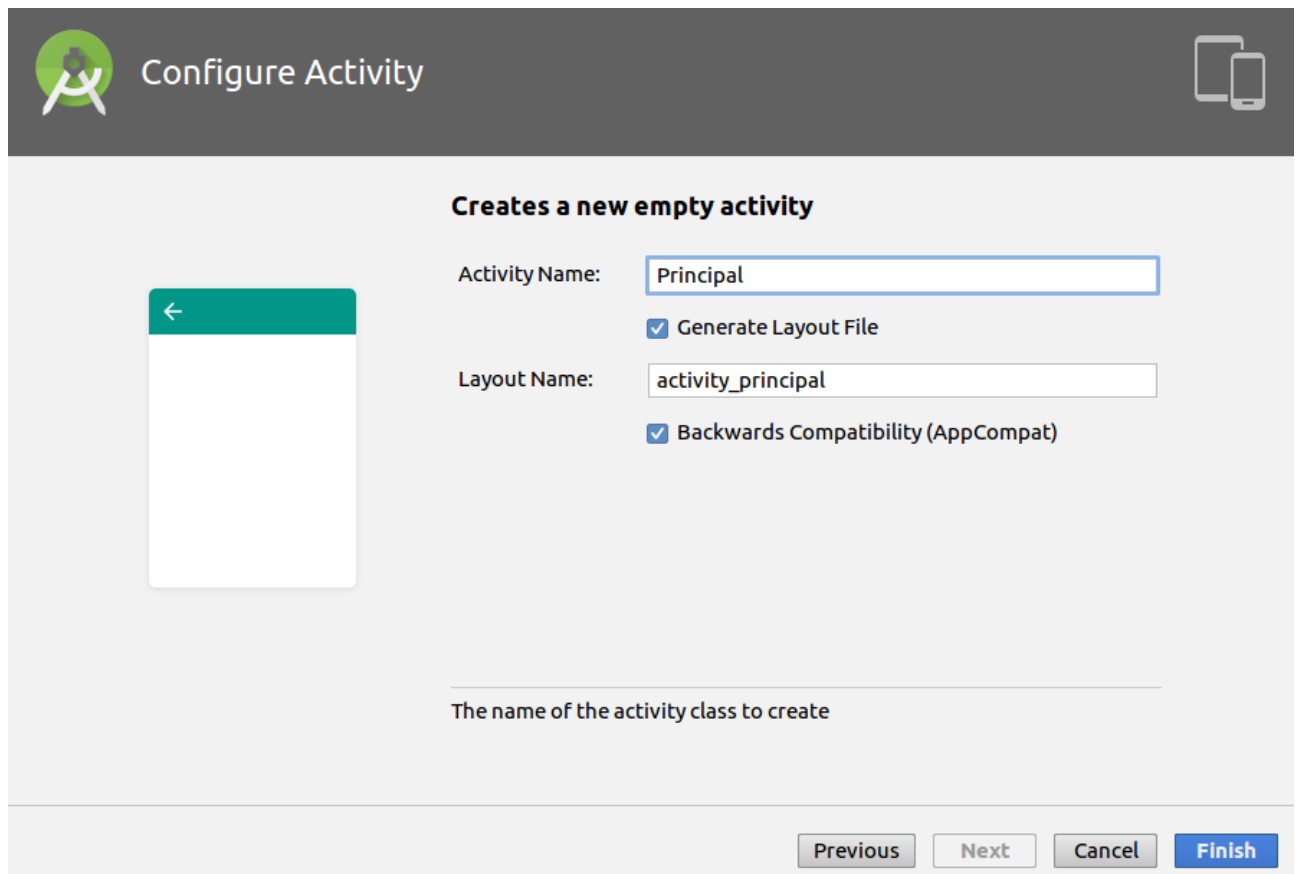
At the bottom right, there are four buttons: 'Previous' (disabled), 'Next' (active), 'Cancel', and 'Finish' (disabled).



The screenshot shows the 'Add an Activity to Mobile' dialog in Android Studio. The title bar includes the Android Studio logo, the text 'Add an Activity to Mobile', and a mobile device icon. The dialog displays a grid of activity templates:

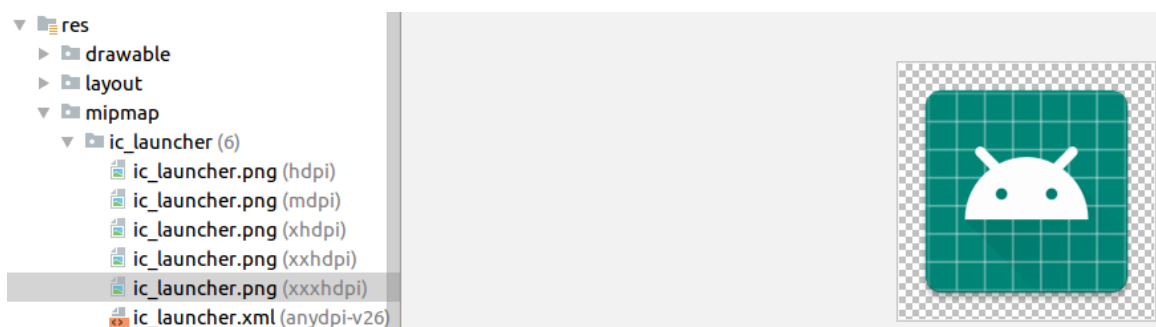
- Add No Activity:** A text label.
- Basic Activity:** A template with a green header, a white body, and a yellow plus button at the bottom right.
- Bottom Navigation Activity:** A template with a green header, a white body, and a yellow and green bottom navigation bar.
- Empty Activity:** A template with a green header and a white body, highlighted with a blue border and a blue background.
- Other templates:** A grid of four more templates showing various UI elements like a map, an ad, and a list.

At the bottom right, there are four buttons: 'Previous' (disabled), 'Next' (active), 'Cancel', and 'Finish' (disabled).

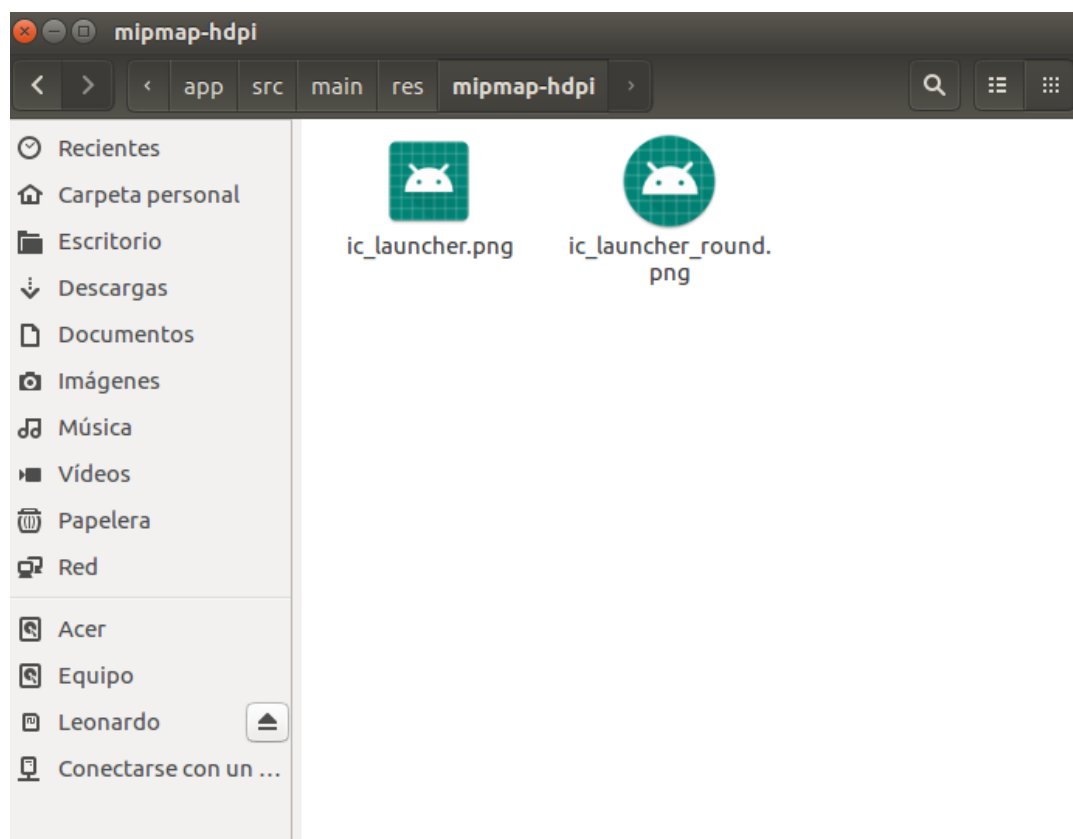
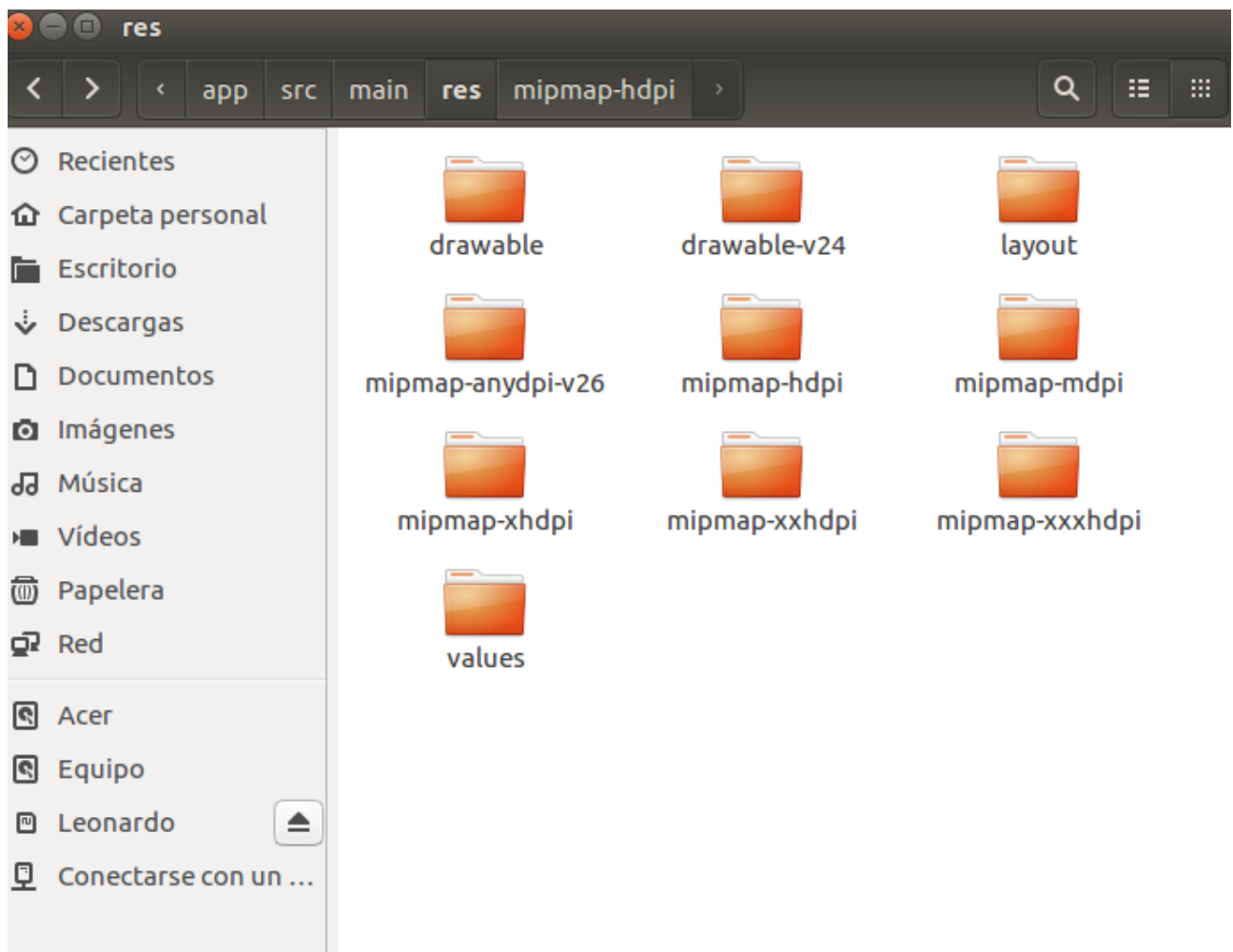


o) Una vez creado el proyecto, describe donde encontramos los siguientes elementos:

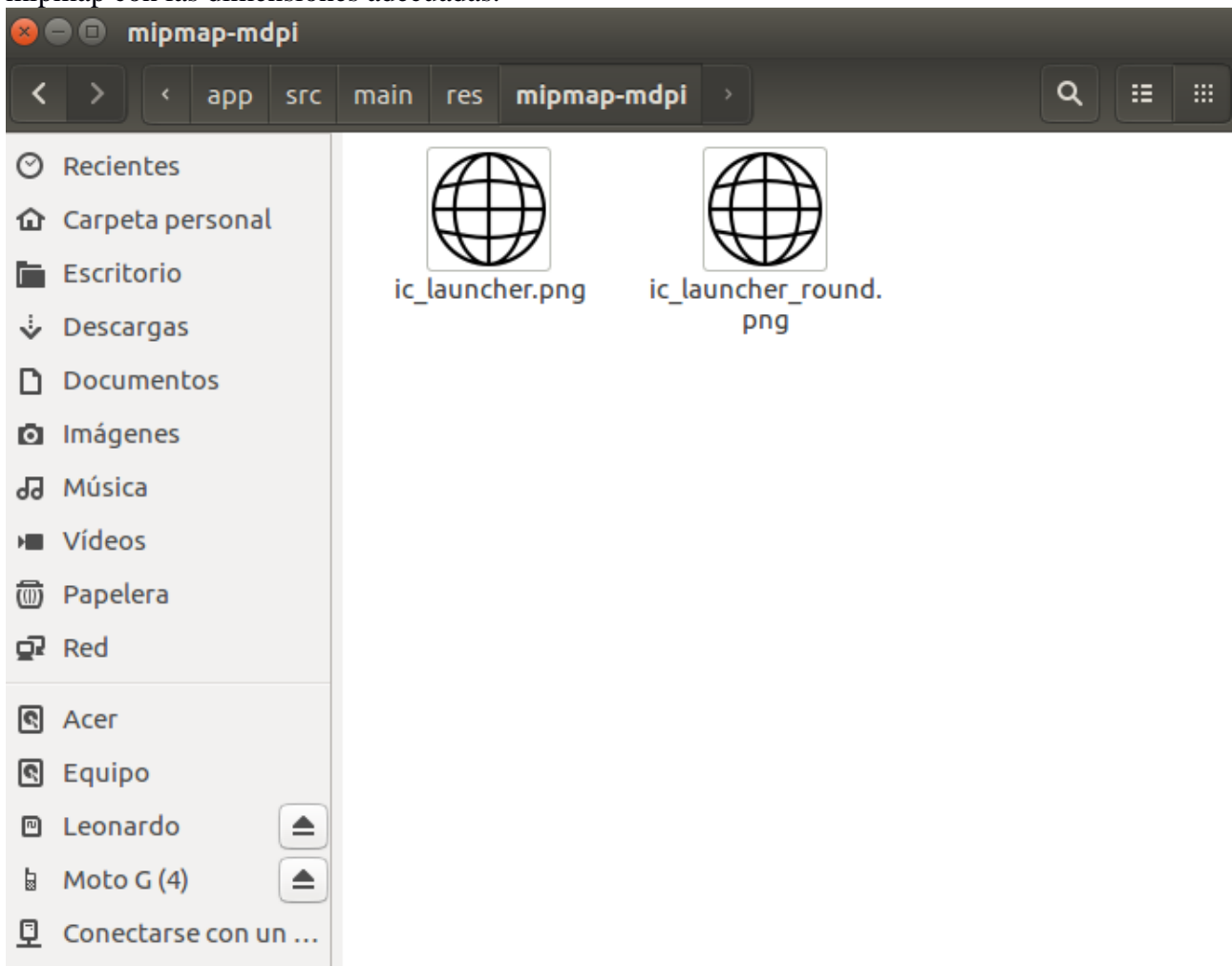
- El icono de la app
El icono de la aplicación la podemos encontrar accediendo a la **carpeta del proyecto > app > src > main > res > mipmap** como se muestra en la siguiente imagen.



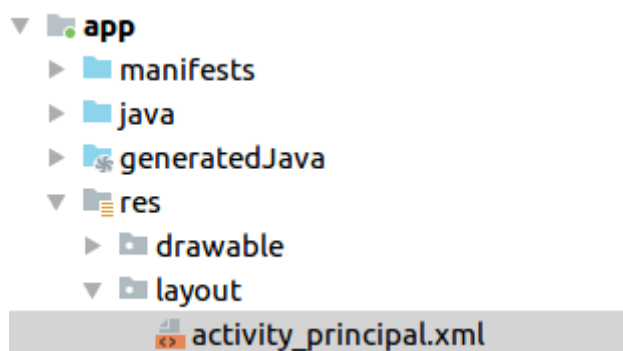
Para cambiar el icono de la aplicación se accede a la carpeta como el paso anterior. Una vez situados en esta carpeta, hay que cambiar el icono de la aplicación en varias carpetas y con dimensiones diferentes. Se le puede poner el mismo nombre `ic_launcher.png`.



A continuación descargamos una imagen en formato png y se pega en cada una de las carpetas mipmap con las dimensiones adecuadas.



- Las plantillas de la aplicación
Estas se encuentran accediendo al **app > layout**



- El archivo de configuración AndroidManifest.xml

Es un archivo de configuración donde podemos aplicar las configuraciones básicas de nuestra app. Su configuración puede realizarse a través de una interfaz gráfica, pero es recomendable conocer la sintaxis ya que en muchas ocasiones será más fácil y rápido hacerlo desde el propio xml. El android manifest está situado en la raíz de cada aplicación y hace lo siguiente:

- Nombra el paquete de Java para la aplicación. El nombre del paquete sirve como un identificador único para la aplicación.
- Describe los componentes de la aplicación: las actividades, los servicios, los receptores de difusión y los proveedores de contenido de los que se compone la aplicación. Nombra las clases que implementan cada uno de los componentes y publica sus capacidades (por ejemplo, qué mensajes de Intent pueden manejar). Estas declaraciones permiten que el sistema Android sepa cuáles son los componentes y bajo qué condiciones se pueden lanzar.
- Determina qué procesos alojarán los componentes de la aplicación.
- Declara qué permisos debe tener la aplicación para acceder a partes protegidas de la API e interactuar con otras aplicaciones.
- También declara los permisos que otros deben tener para interactuar con los componentes de la aplicación.
- Enumera las clases de Instrumentación que proporcionan información de perfil y otra información a medida que se ejecuta la aplicación. Estas declaraciones están presentes en el manifiesto solo mientras la aplicación se está desarrollando y probando; Se eliminan antes de que se publique la aplicación.
- Declara el nivel mínimo de la API de Android que requiere la aplicación.
- Enumera las bibliotecas con las que se debe vincular la aplicación.

- **Los strings de la app**

Un String en Android, es un recurso de texto que nos va permitir poder contar con valores dentro de nuestra aplicación, estos valores pueden ser utilizando en cualquier parte de la aplicación.

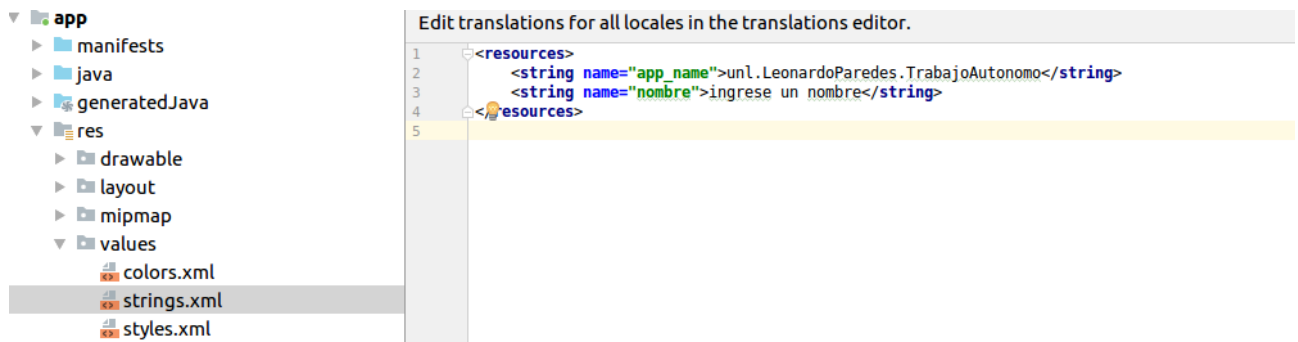
Existen tres tipos de recursos que pueden proporcionar strings para tu aplicación

- Cadena simple: Es un recurso del tipo XML que representa a una sola cadena de texto.
- Arreglo de cadenas: Recurso del tipo XML que proporciona un arreglo (array) de cadenas de texto.
- Cantidad de cadenas (plurales): Recurso XML que representa cadenas de pluralización.

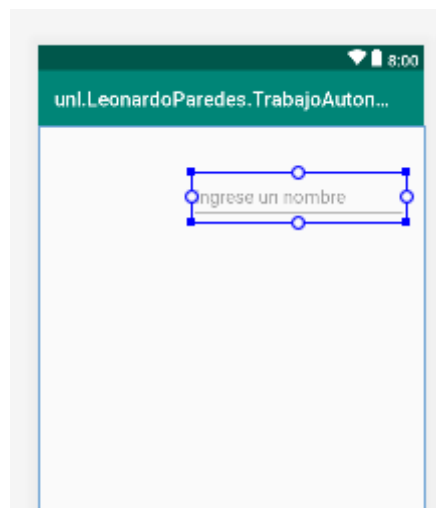
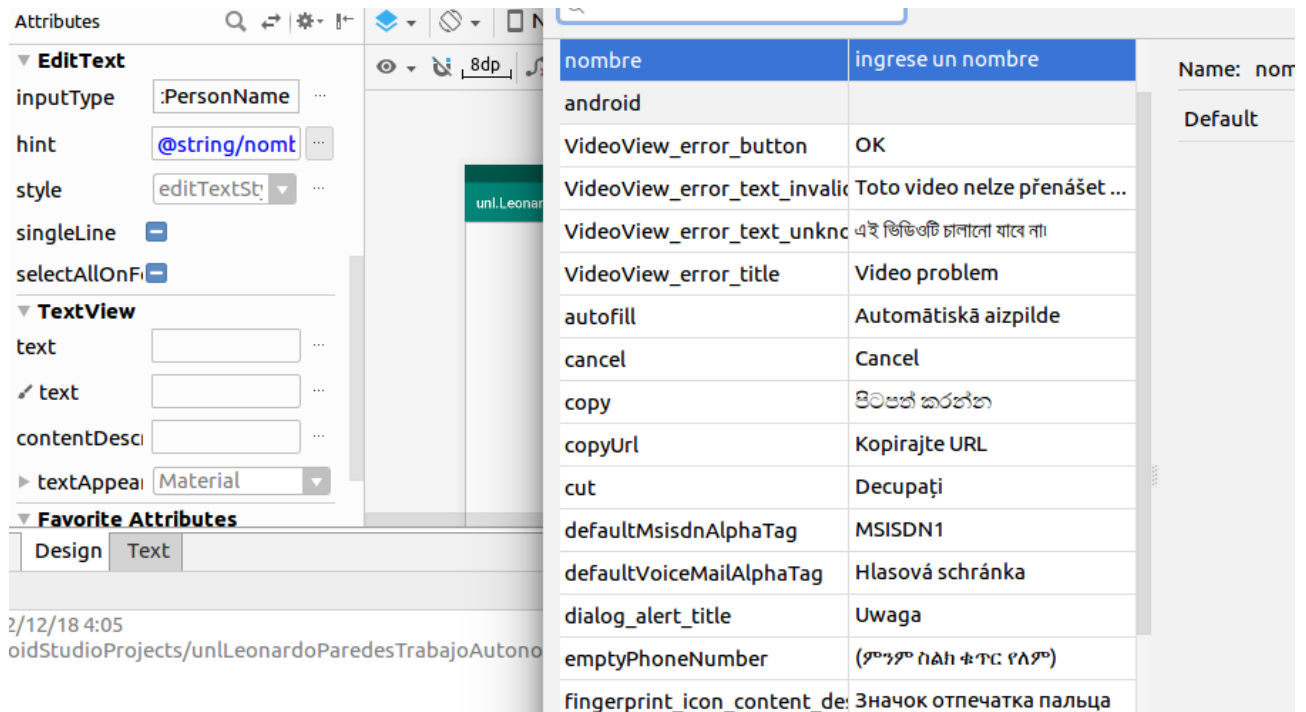
En cualquiera de estos tipos es posible aplicar métodos de marcado y estilo de textos,

Todos los tipos antes mencionados se declaran en el **archivo de recursos strings.xml ubicado en la carpeta de recursos de nuestro proyecto.**

Para acceder se entra al **app > values > string.xml**, dentro del documento xml y creamos un nuevo recurso con name: “**nombre**” y con un mensaje “**Ingrese un nombre**” como se observa en la imagen.



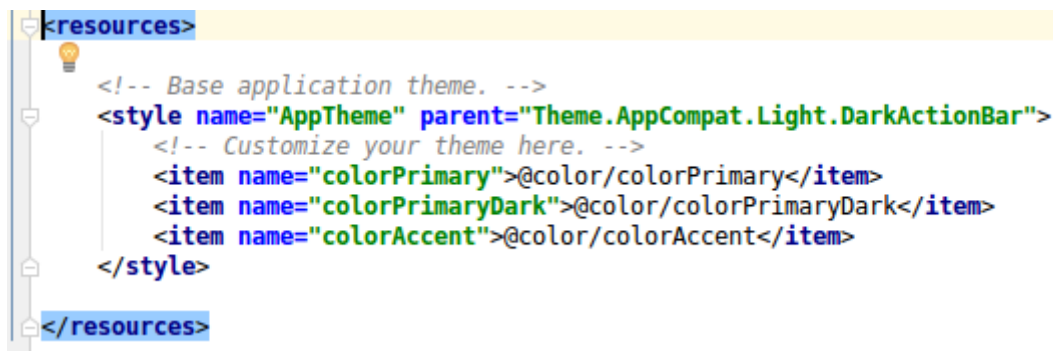
Posteriormente se lo enlaza a un Text Plain en otro xml mediante un Hint a través del name que se puso al recurso creado anteriormente.



p) Explica que definen y para qué sirven los siguientes archivos:

styles.xml: es una colección de propiedades que especifican la apariencia y el formato de una View o ventana. Un estilo puede especificar propiedades, como altura, relleno, color de fuente, tamaño de fuente, color de fondo y mucho más. Los estilos se definen en un recurso XML que está separado del XML que especifica el diseño.

En Android, los estilos comparten una filosofía similar a las hojas de estilo CSS del diseño web: permiten separar el diseño del contenido.



```
<resources>

    <!-- Base application theme. -->
    <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>
    </style>

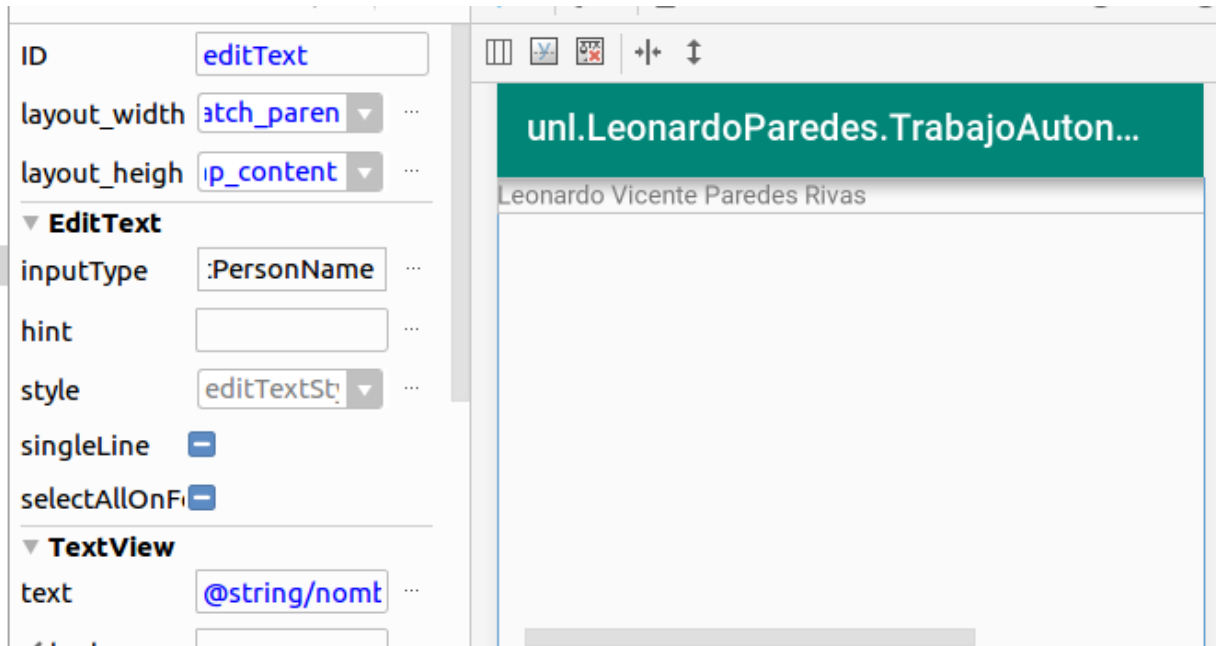
</resources>
```

MainActivity.xml; es la actividad principal de nuestra aplicación, en esa clase definiremos los métodos y llamadas a otras clases, así como el funcionamiento del *layout*.

build.gradle (el que corresponde a la aplicación): es un sistema de compilación que reúne en un solo las mejores prestaciones de otros sistemas de compilación. Está basado en JVM (Java Virtual Machine), lo que significa que puedes escribir tu propio script en java, y que Android Studio lo entenderá y lo usará.

Lo mejor de gradle es que es un plugin, lo que facilita su actualización y su exportación de un proyecto a otro. Esto significa que puedes tener tu propio lenguaje de programación y automatizar el proceso de compilación en un solo paquete (de la misma manera que un jar en caso de java) y poder distribuirlo al resto del mundo.

q) Cambia el texto del TextView y escribe tus nombres y tus apellidos. Adjunta una captura de pantalla y explica cuáles son las formas de asignar cadenas de texto a los elementos TextView.



r) Explica que tipo de Logs (Log) hay en Android y añade uno al final del método onCreate() del archivo MainActivity.java. Como etiqueta (Tag) pon el nombre de la clase y como texto del mensaje tus nombre y apellidos. Pega a continuación la línea de Log que has utilizado.

La clase Log te permite crear mensajes de registro que aparecen en el monitor de logcat. Por lo general, debes usar los siguientes métodos de registro, que se ordenan de mayor a menor prioridad:

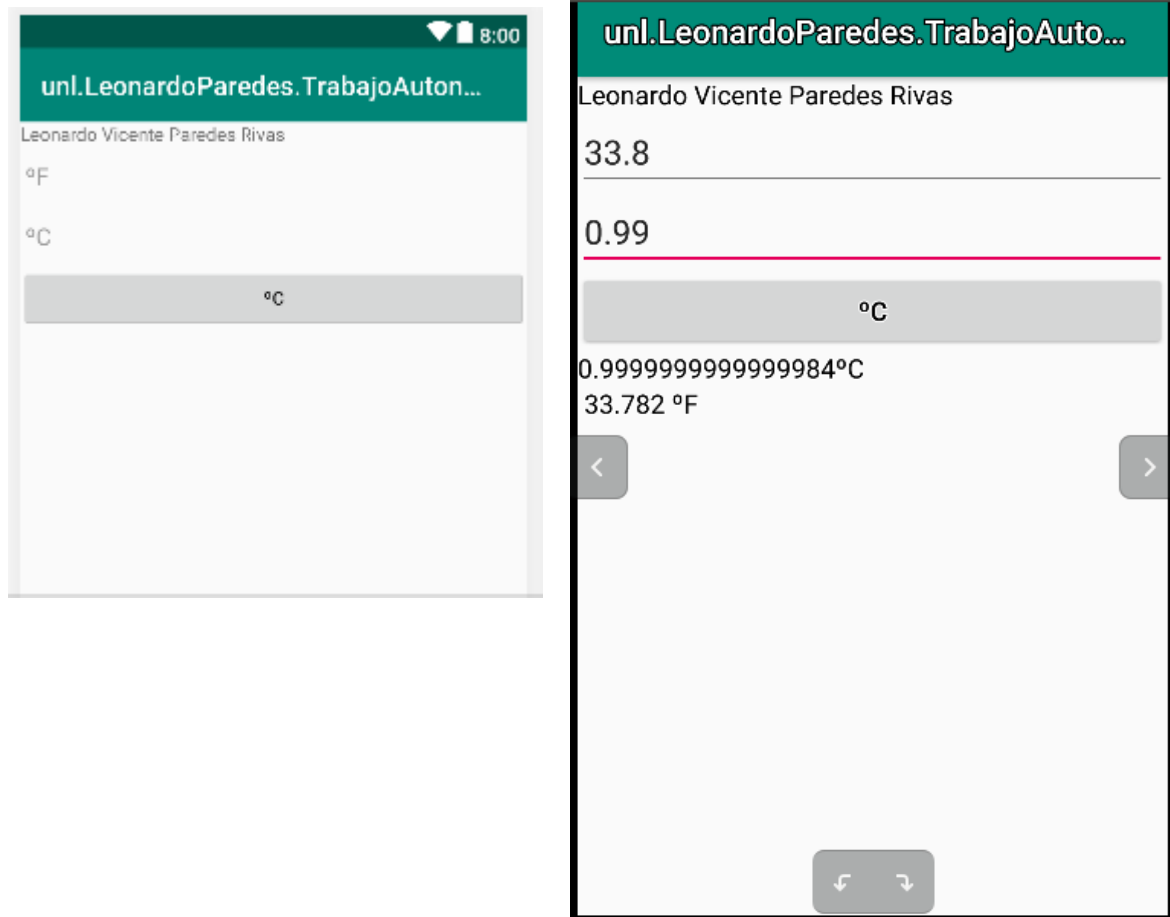
- Log.e(String, String) (error)
- Log.w(String, String) (advertencia)
- Log.i(String, String) (información)
- Log.d(String, String) (depuración)
- Log.v(String, String) (detalle)

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_principal);

    // Log.e(String, String) (error)
    // Log.w(String, String) (advertencia)
    // Log.i(String, String) (información)
    // Log.d(String, String) (depuración)
    // Log.v(String, String) (detalle)

    boton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            double n= Double.parseDouble(gradosF.getText().toString());
            double n1=Double.parseDouble(gradosC.getText().toString());
            double r= (n-32)/1.8;
            double r1=(n1*1.8)+32;
            resultado.setText(" " + r+ " ºC ");
            resultado1.setText(" "+r1+" ºF");
        }
    });
    Log.w(TAG, msg: "Leonardo Vicenete Paredes Rivas");
}
```

s) En la Main Activity agrega las views necesarias para realizar la conversión de grados Fahrenheit a Centígrados y viceversa.



t) Suba su proyecto a su cuenta de GitLab, GitHub, Bitbucket, etc, la visibilidad debe ser pública, escriba a continuación la URL de su proyecto Android.