28-11-2019

"Tomando repositorio de YouTube"

Proyecto Final de Android

José Luis Rodriguez Andrade

No. Control:

TS15110036

Materia:

Programación Móvil

Profesor:

Rafael Ramírez Rosillo

ITESI Extensión Tarimoro

Para el desarrollo de esta aplicación primeramente se necesitó tener instalado el Android studio y actualizado todas actualizaciones necesarias para su funcionamiento correcto e instalado un emulador, en este caso se utilizo el Genymotion el cual nos ofrece una gran cantidad de API´s y emuladores distintos para probar nuestras aplicaciones desarrolladas en Android.

Para iniciar con el desarrollo primero tenemos que tener abierto el android studio y un emulador de Android en este caso se emulara en la API 28 "*Samsung Galaxy S8 con Android en su versión 9.0*"

Materiales para utilizar

- Android Studio
- Genymotion
- Servicios de Google

Layout utilizados

- Activity_Main.xml (por default
  - Componentes:
    - TextView
    - ImageView
    - recycleView

1. iniciamos con un nuevo proyecto en blanco

2. 2. enseguida abrimos el build.gradle (module.app y agregamos el siguiente código dentro de (las dependencias) y sincronizamos, ilustración 1.

```
dependencies {
        implementation fileTree(include: ['*.jar'], dir: 'libs')
        implementation 'androidx.appcompat:appcompat:1.1.0
         implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
        testImplementation 'junit:junit:4.12'
        androidTestImplementation 'androidx.test.ext:junit:1.1.1'
        androidTestImplementation 'androidx.test.espresso:espresso-
        core:3.2.0'
        implementation files('libs/YouTubeAndroidPlayerApi.jar')

        // libreria para imagenes
        implementation 'com.github.bumptech.glide:glide:3.7.0';
    }
```

Ilustración 1, Dependencias del Build.gradle (Modulo: App)

En seguida abrimos el archivo del AndroidManifest.xml y agregamos los permisos de **internet** colocamos el siguiente código para los permisos, ilustración 2.

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

    <!--Se agregan os permisos de Internet-->
    <uses-permission android:name =
   "android.permission.INTERNET"> </uses-
permission>
```

Ilustración 2, Permiso de internet en el Manifests

3. Una vez la configuración los permisos nos dirigimos a la siguiente URL https://developers.google.com/youtube/android/player/downloads?hl=es para poder descargar el archivo YouTubeAndroidPlayerApi-1.2.2.zip, una vez descargado lo descomprimimos y abrimos la carpeta y solo, únicamente utilizaremos el archivo dentro de la carpeta lib el archivo *llamado YouTubeAndroidPlayerApi.jar*

4. Enseguida lo copeamos en la carpeta app > libs lo pegamos para ver estas carpetas cambiamos en el lado derecho del Android Studio de Android a Project, ilustración 3

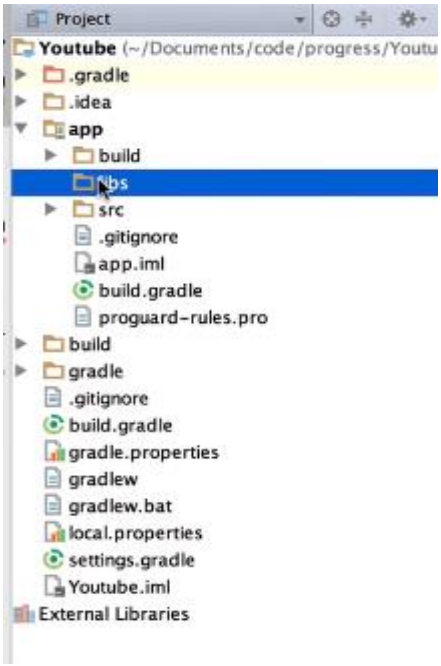

Ilustración 3, Librería Externa

5. Una vez agregada la librería de YouTube, creamos la vista para el video de YouTube que se mostrará en el contenedor. Para ello nos dirigimos al activity_main.xml que lo encontramos en app > res > layout una vez que regresemos la dirección de Project a Android en la parte izquierda del Android Studio, ilustración 4

```
<!-- Contenedor video-->
<com.google.android.youtube.player.YouTubePlayerView
    android:id="@+id/youtube_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="220dp" />
```

Ilustración 4, Contenedor del Video

6. enseguida nos dirigimos al MainActivity.java que se encuentra java > com.example.youtube el ultimo nombre dependerá del nombre que se le dio al proyecto, colocamos el siguiente código el cual hace referencia a los elementos dentro del activity_main.xml (los cuales a continuación se mencionarán en el siguiente paso) , ilustración 5.

```
package com.example.youtube;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.widget.ImageView;
import android.widget.Toast;

import com.bumptech.glide.Glide; // Importación de la
libreria de Glide

import
com.google.android.youtube.player.YouTubeBaseActivity;
import
com.google.android.youtube.player.YouTubeInitializatio
nResult;
import
com.google.android.youtube.player.YouTubePlayer;
import
com.google.android.youtube.player.YouTubePlayerView;


public class MainActivity extends YouTubeBaseActivity
implements YouTubePlayer.OnInitializedListener,
YouTubePlayer.PlaybackEventListener {

    // Se crea una variable donde contiene la clave
generada y guardada desde los servicios de google
    String claveYoutube=
"AIzaSyCfKOUTc9zx_1_4FdMMRw63vDOMH4MMLoU";

    // se crear una variable
    YouTubePlayerView youtubePlayerView;

    @Override
    protected void onCreate(Bundle savedInstanceState)
{
```

Ilustración 5, Código del MainActivity.java

7. A continuación, se agrega el siguiente código en el Activity_main.xml, el cual se hizo referencia al anterior paso #6, ilustración 6.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout

xmlns:android="http://schemas.android.com/apk/res/andr
oid"
    xmlns:app="http://schemas.android.com/apk/res-
auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/fondo"
    tools:context=".MainActivity">

    <!-- Titulo-->
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="14dp"
        android:text="Repositorio de Youtube"
        android:textAlignment="center"
        android:textColor="#ffffff"
        android:textSize="30sp" />

    <!-- Contenedor video ESTE FUE AGREGADO
ANTERIORMENTE -->

<com.google.android.youtube.player.YouTubePlayerView
        android:id="@+id/youtube_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="220dp" />
    <!-- Contenedor video ESTE FUE AGREGADO
ANTERIORMENTE -->

    <!-- imagen 1-->
    <ImageView
        android:id="@+id/imageView1"
        android:layout_width="517dp"
        android:layout_height="157dp"
        android:layout_marginTop="60dp"
        android:src="@drawable/image1" />
```

Ilustración 6, Código del Activity Main.xml

8. Enseguida colocaremos el color de fondo agregando un archivo de extensión xml, lo llamaremos fondo.xml en la carpeta app > res > Drawable haciendo clic derecho sobre la carpeta Drawable como se muestra en la siguiente ilustración, ilustración 7.

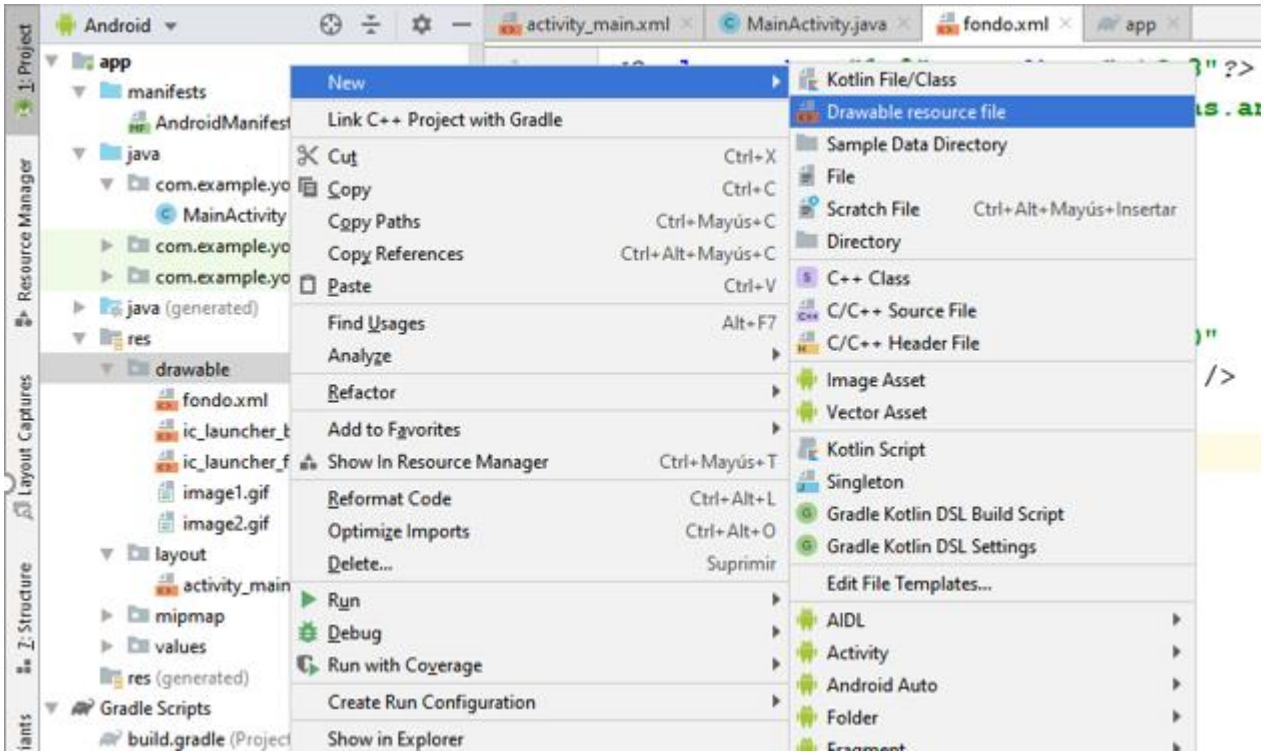

Ilustración 7, utilización Google map para clave & huella digital

9. Una vez ya agregado lo anterior, abrimos el navegador y colocamos lo siguiente *console developer* o colocando la URL *https://console.developers.google.com/apis* en el primer enlace y para crear un nuevo proyecto nos vamos a *YoutubeAPI > Nuevo proyecto*, ilustración 8.

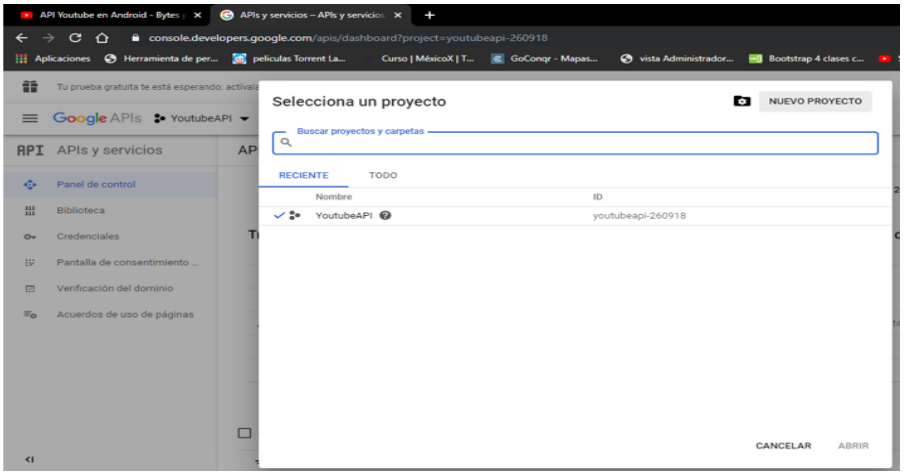

Ilustración 8, Búsqueda de la API Google

10. Enseguida se abrirá una pantalla de llenado en el cual le pondremos el nombre referente al proyecto creado en Android Studio con un agregado API y damos clic en crear y se espera a que termine de cargar, , ilustración 9 y 10.

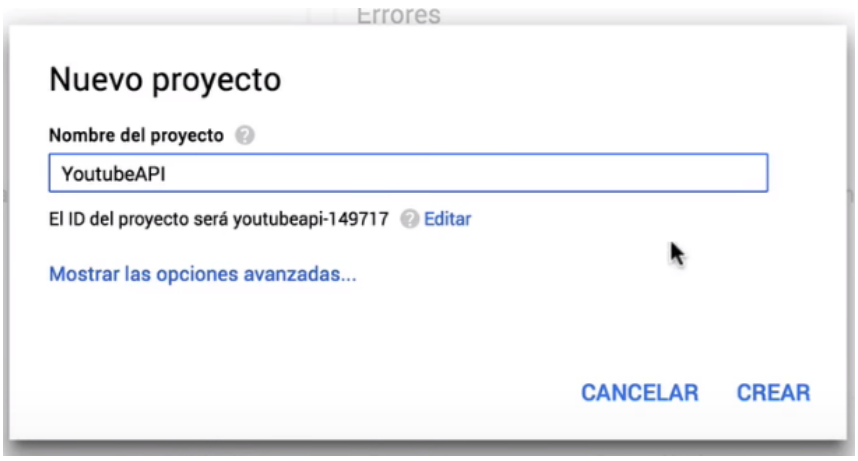

Ilustración 9, creación de Nuevo Proyecto en Google API

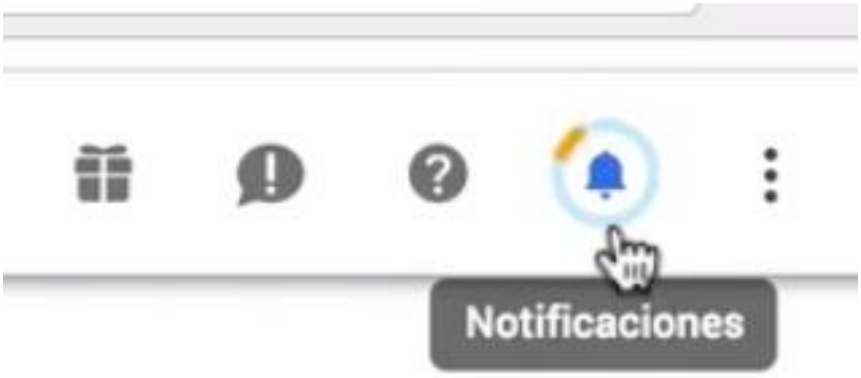

Ilustración 10, Caga de Guardado

11. A continuación, seleccionamos del menú izquierdo en *credenciales* y hacemos un clic en crear credencial > clave de API y finalmente en restringir API, ilustración 11 y 12.

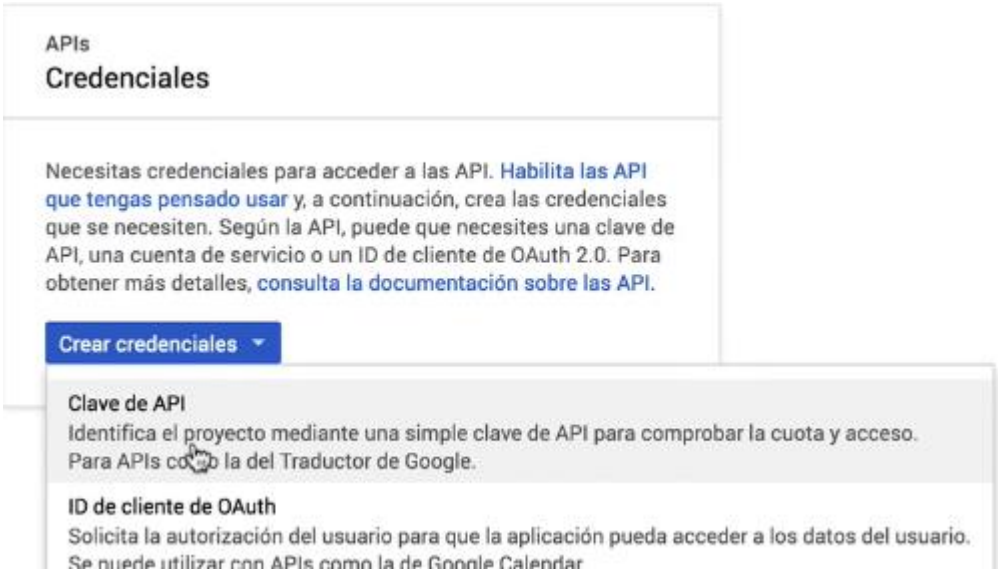

Ilustración 11, Creación de credenciales 1

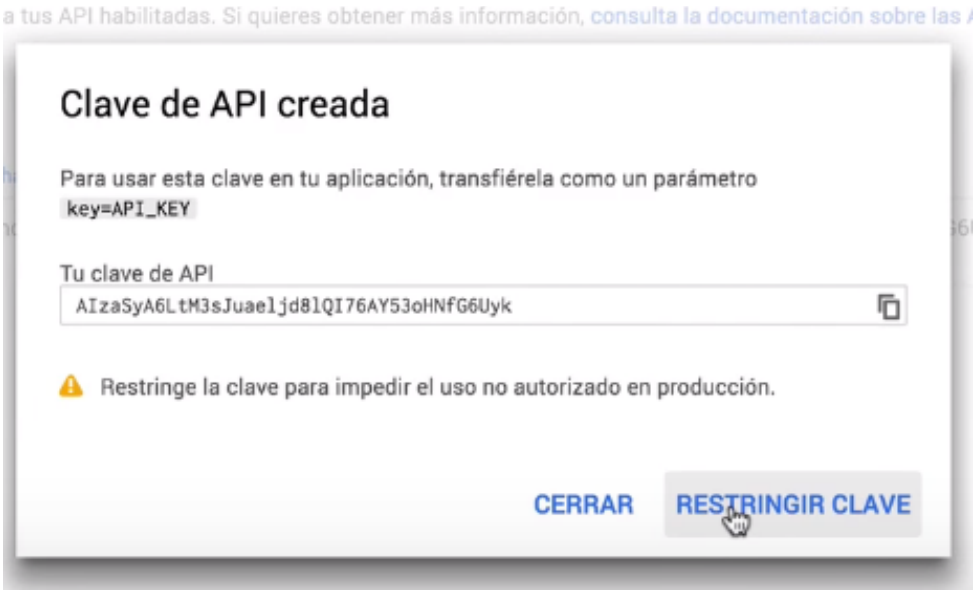

Ilustración 12, Creación de credenciales 2

12. Una vez restringida la clave seleccionamos en Aplicaciones para Android y para debajo de la misma pagina agregamos una nueva Huella Digital, **en este caso se creó en Android Studio**

en el mismo proyecto un activity, pero de Google maps para poder tomar la clave del proyecto una vez copeado la llave generada nos presionamos en donde se encuentra la huella *control* + *z* para eliminar el activity creado, ilustración 13

```
<resources>
    <!--
    TODO: Before you run your application, you need a
Google Maps API key.

    To get one, follow this link, follow the
directions and press "Create" at the end:

https://console.developers.google.com/flows/enableapi?
apiid=maps_android_backend&keyType=CLIENT_SIDE_ANDROID
&r=2B:1A:44:86:59:5B:79:DD:83:9B:78:D4:A0:CF:D4:BB:14:
F7:9D:D1%3Bcom.example.myapplication

    You can also add your credentials to an existing
key, using these values:

    Package name:

2B:1A:44:86:59:5B:79:DD:83:9B:78:D4:A0:CF:D4:BB:14:F7:
9D:D1

    SHA-1 certificate fingerprint:

2B:1A:44:86:59:5B:79:DD:83:9B:78:D4:A0:CF:D4:BB:14:F7:
9D:D1

    Alternatively, follow the directions here:

https://developers.google.com/maps/documentation/andro
id/start#get-key

    Once you have your key (it starts with "AIza"),
replace the "google_maps_key"
    string in this file.
    -->
    <string name="google_maps_key"
templateMergeStrategy="preserve"
translatable="false">YOUR_KEY_HERE</string>
</resources>
```

Ilustración 13, Copiado de la clave / Huella Digital

13. A continuación, copeamos el nombre del proyecto, el cual se encuentra en la carpeta java, y colocamos el nombre del archivo *“com.example.**nombre del proyecto”,*** enseguida ***presionamos el botón guardar***

14. Una vez terminado guardada quedara asi, ilustración 14

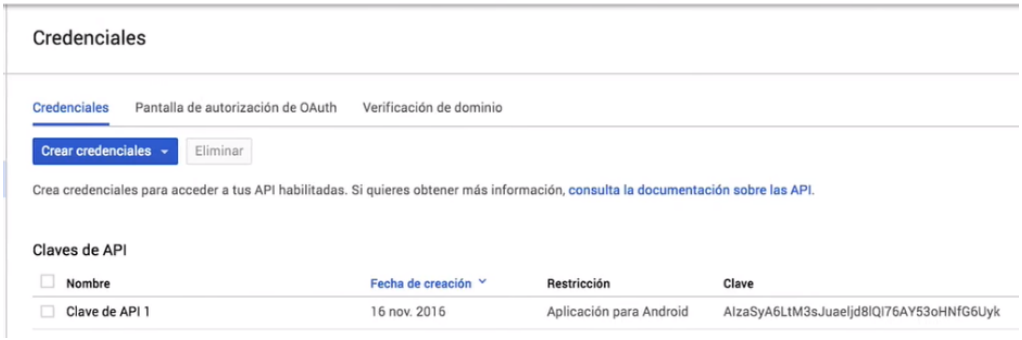

Ilustración 14, credencial guardada correctamente

15. Enseguida, nos dirigimos a biblioteca en el menú y elegiremos la opción APIs de YouTube > *YouTube para API* o YouTubeData API, ilustración 15.

Ilustración 15, YouTube Data API

16. A continuación, habitamos YouTube Data API v3, ilustración 16.

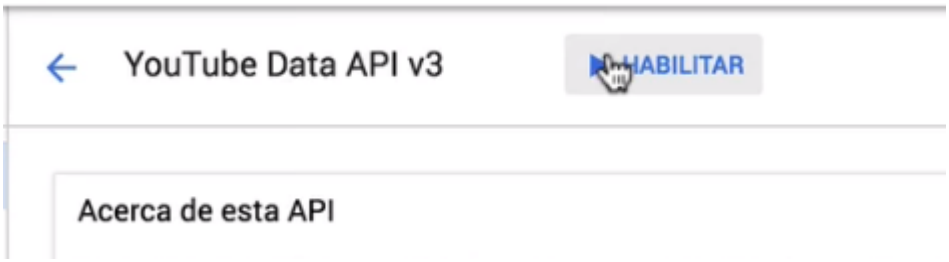

Ilustración 16, Habilitar YouTube Data API v3

17. Finalmente, habilitado el servicio, nos regresamos a credenciales, copeamos la clave, ilustración 17 la cual ira pegada en la variable creada anteriormente en ***String clave***, ilustración 18.

Ilustración 17, Clave generada por YouTube API

```
public class MainActivity extends YouTubeBaseActivity
implements YouTubePlayer.OnInitializedListener,
YouTubePlayer.PlaybackEventListener {


// Se crea una variable donde contiene la clave
generada y guardada desde los servicios de Google

 String claveYoutube=
"AIzaSyCfKOUTc9zx_1_4FdMMRw63vDOMH4MMLoU";
```

Ilustración 18, Clave colocada en el proyecto

18. Finalmente configuramos los servicios de Google ya que con los emuladores de GENYMOTION vienen limpios de cualquier tipo de aplicación básica de un teléfono, para eso nos dirigimos a nuestro emulador descargado, (en este caso se utilizó la API 28 con Android en su versión 9.0).

19. A continuación, una vez iniciado nuestro emulador nos dirigimos al lado derecho superior nos apareceré un icono antes de la batería ilustrasción19, el cual presionamos un clic y se nos abrirá una ventana la cual son los servicios de Google aceptamos y se descargara automáticamente

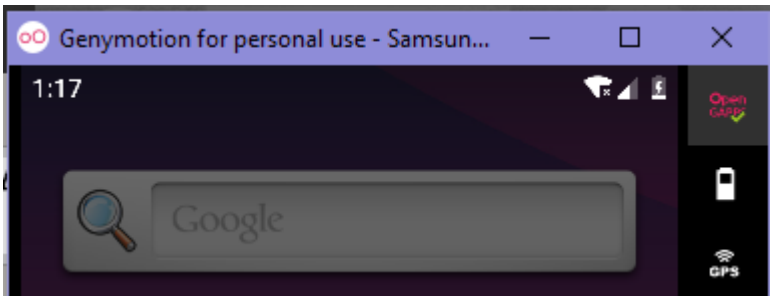

Ilustración 19, Inicio de instalación de Servicios de Google Play

20. Una vez instalados los servicios de Google instalamos la app de YouTube para que el repositorio (NUESTRA APP) funcione correctamente

21. Finalmente ejecutamos el proyecto desde el Android Studio y se nos abrirá en el emulador la app funcionando con una conexión de red o wifi, ilustración 20.

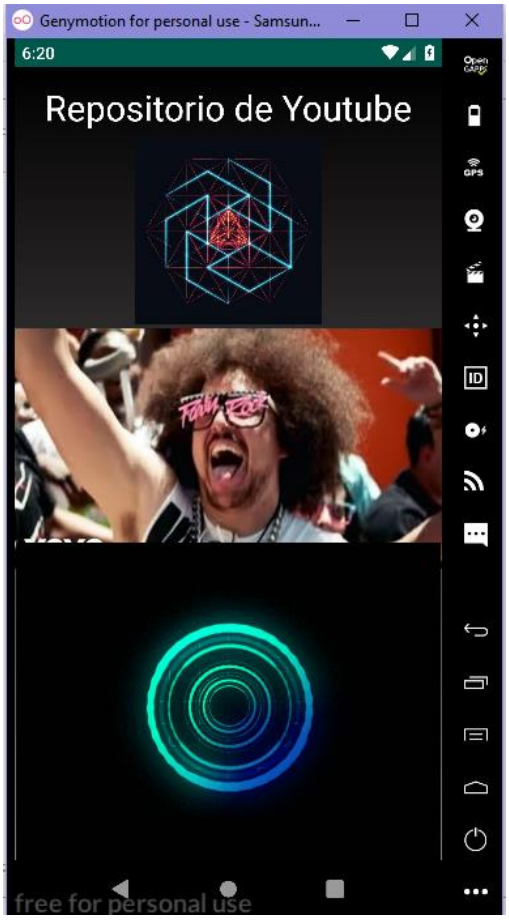

Ilustración 20, Finalización de proyecto, (vista de Diseño)