



APK (SEGÚN REQUERIMIENTOS DEL PROYECTO)

Análisis y Desarrollo de Software
Juan Manuel Aldana Zambrano



21 DE MARZO DE 2024
JOSE MANUEL HUELVAS BLANCO
CENTRO DE HOTELERIA, TURISMO Y ALIMENTOS

Tabla de contenido

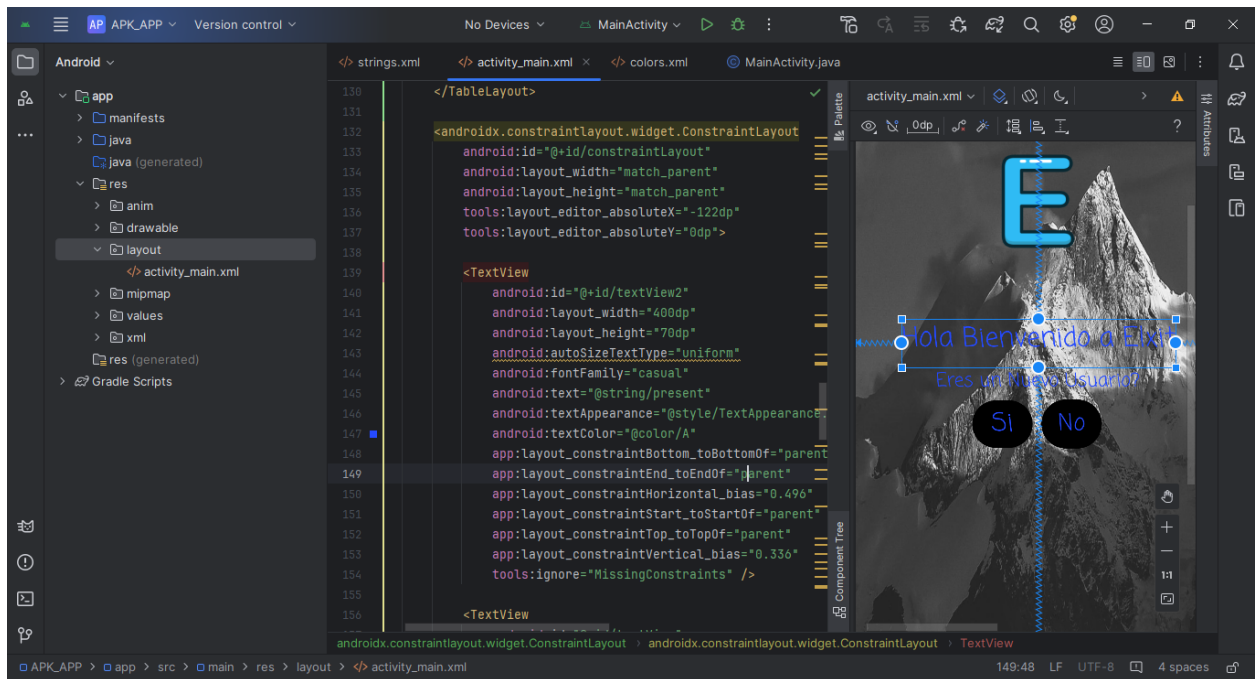
INTRODUCCION	2
Documentación de los Ambientes de Desarrollo y Pruebas	3
Carpeta Layout " <i>activity_main.xml</i> "	3
Archivo " <i>MainActivity.java</i> "	4
Carpeta "values" (" <i>strings.xml</i> , <i>colors.xml</i> ")	4
Carpeta mipmap y drawable	5
Funciones en los objetos en el archivo " <i>MainActivity.java</i> "	5
Animaciones (En una imagen)	6
Pruebas de la Aplicación	7
Pantalla de la APP	8
APP_APK	8

INTRODUCCION

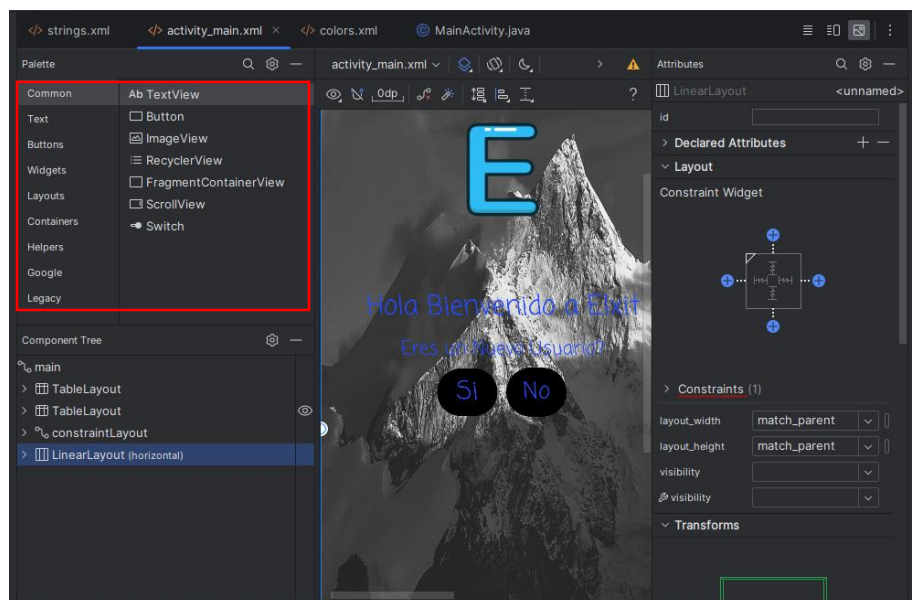
En el presente documento se busca codificar los módulos del proyecto con base a en los requerimientos del sistema orientados en dispositivos móviles bajo la plataforma Android. Por eso se creará una APK (Android Application Pckage), los cuales son ejecutables diseñados para Android y es uno de los términos que más se repiten en cualquier proyecto de apps. Se trata de un archivo ejecutable que contiene todos los datos que se necesitan para instalar y hacer funcionar una aplicación Android. Estos archivos APK puede ser un arma de doble filo, por una parte, puede ayudar a instalar aplicaciones que no están en Google Play, y por el otro puede que acabe siendo un virus o malware instalado en el dispositivo móvil si no se tiene cuidado. Esta actividad se centra en el diseño de las nuevas tecnologías emergentes y disruptivas como lo es el desarrollo de la aplicación para dispositivos móviles con énfasis en Android así, como también de obtener conocimientos en tecnología Blockchain (Cadena de bloques) y Machine Learning (Aprendizaje de Maquina) etc.

Documentación de los Ambientes de Desarrollo y Pruebas

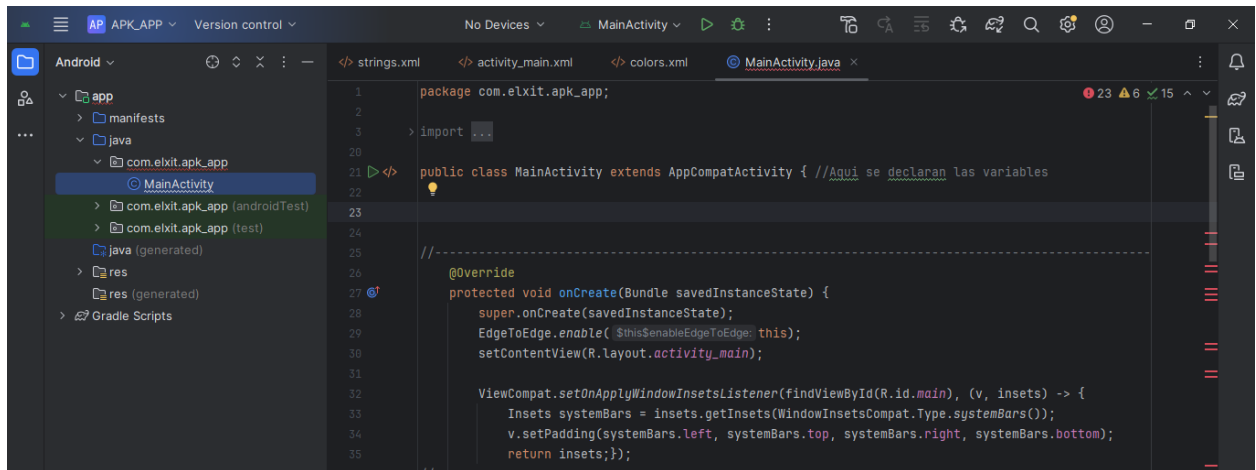
Carpeta Layout “activity_main.xml”



En este archivo es donde se mostrará todo en la pantalla principal y en donde se agregarán las imágenes, textos, botones, etc. En este Código se muestra el siguiente texto del primer pantallazo en el “layout” con un “TexteView”, pero es más fácil agregar cada objeto arrastrando las opciones que se muestran en el segundo pantallazo.



Archivo “MainActivity.java”

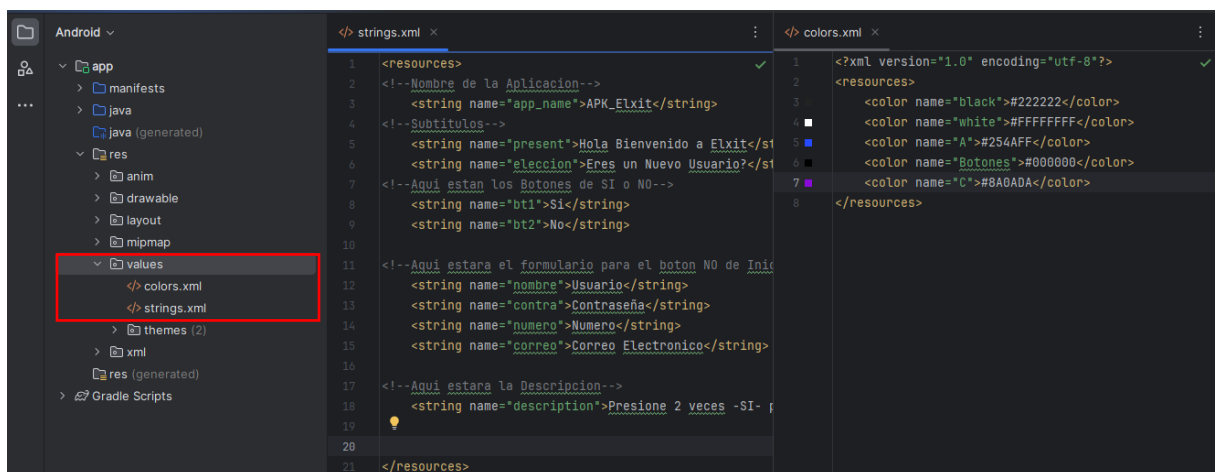


The screenshot shows the MainActivity.java file in an IDE. The left sidebar displays the project structure with 'com.elxit.apk_app' selected. The main editor area shows the following Java code:

```
1 package com.elxit.apk_app;
2
3 import ...
4
5 public class MainActivity extends AppCompatActivity { //Aqui se declaran las variables
6
7 //-----
8
9 @Override
10 protected void onCreate(Bundle savedInstanceState) {
11     super.onCreate(savedInstanceState);
12     EdgeToEdge.enable(this);
13     setContentView(R.layout.activity_main);
14
15     ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
16         Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
17         v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
18         return insets;});
19 }
```

Este archivo es prácticamente el más importante de la app ya que con esto inicia el layout de mi aplicación y en donde los objetos pueden tener diferentes funciones.

Carpeta “values” (“strings.xml, colors.xml”)



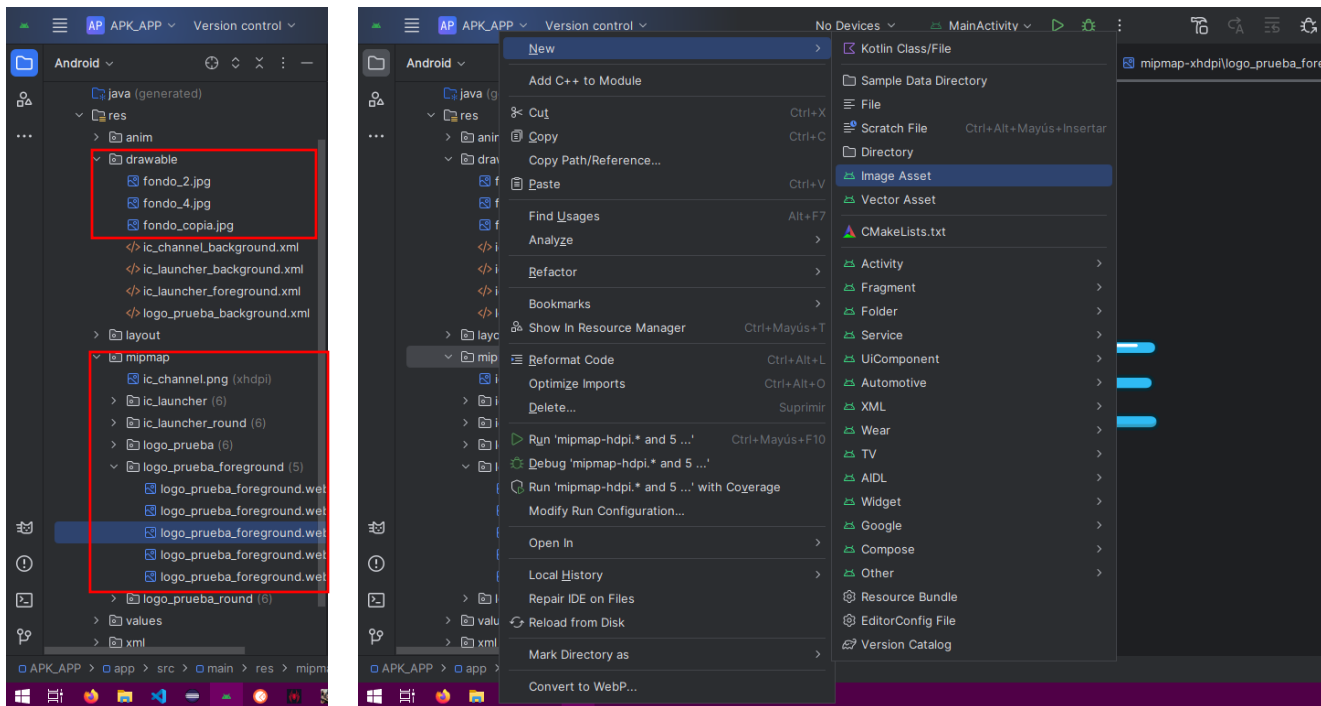
The screenshot shows the 'values' folder in an IDE. The left sidebar displays the project structure with 'values' selected. The main editor area shows the following XML code for strings.xml and colors.xml:

```
1 <resources>
2 <!--Nombre de la Aplicacion-->
3 <string name="app_name">APK_Elxit</string>
4 <!--Subtitulos-->
5 <string name="present">Hola Bienvenido a Elxit</string>
6 <string name="eleccion">Eres un Nuevo Usuario</string>
7 <!--Aqui estan los Botones de SI o NO-->
8 <string name="bt1">Si</string>
9 <string name="bt2">No</string>
10
11 <!--Aqui estara el formulario para el boton NO de Inicio-->
12 <string name="nombre">Usuario</string>
13 <string name="contra">Contraseña</string>
14 <string name="numero">Numero</string>
15 <string name="correo">Correo Electronico</string>
16
17 <!--Aqui estara la Descripcion-->
18 <string name="description">Presione 2 veces -SI-
19
20
21 </resources>
```

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3 <color name="black">#222222</color>
4 <color name="white">#FFFFFF</color>
5 <color name="A">#254AFF</color>
6 <color name="Botones">#000000</color>
7 <color name="C">#8A0ADA</color>
8 </resources>
```

En el archivo de “strings.xml” es donde se pueden guardar todos los textos los cuales pueden ser reutilizados para el archivo “activity_main.xml”, al igual que el archivo “colors.xml” en donde se guardan los colores que se reutilizaran en mi app y para poder reutilizar estos valores se les asigna un nombre a cada “string” como por ejemplo “name=“app_name”.

Carpeta mipmap y drawable



Dentro de la carpeta “mipmap” en mi caso inserto el logo que utilizare para mi aplicación y también como imagen. Para insertar imágenes que utilizare dentro de mi app las importo dentro de la carpeta drawable y se pueden insertar, dando click derecho en la carpeta y selecciona “Image Asset”.

Funciones en los objetos en el archivo “MainActivity.java”

```

20
21 public class MainActivity extends AppCompatActivity { //Aquí se declaran las variables
22
23     2 usages
24     private ImageView elxiting; //Se importa la imagen desde activity_main
25
26     Button si;
27     1 usage
28     Button no;
29     3 usages
30     TextView correo2;
31     3 usages
32     TextView contra2;
33
34     3 usages
35     TextView usuario;
36     3 usages
37     TextView contra;
38     3 usages
39     TextView numero;
40     3 usages
41     TextView correo;

```

```

//////////////////////////////////// ACCION AL PRESIONAR EL BOTON Y MOSTRAR UN MENSAJE
correo2 = findViewById(R.id.correo2);
contra2 = findViewById(R.id.contra2);
final boolean[] invisible = {true};

si = (Button) findViewById(R.id.si);
no = (Button) findViewById(R.id.no);
si.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

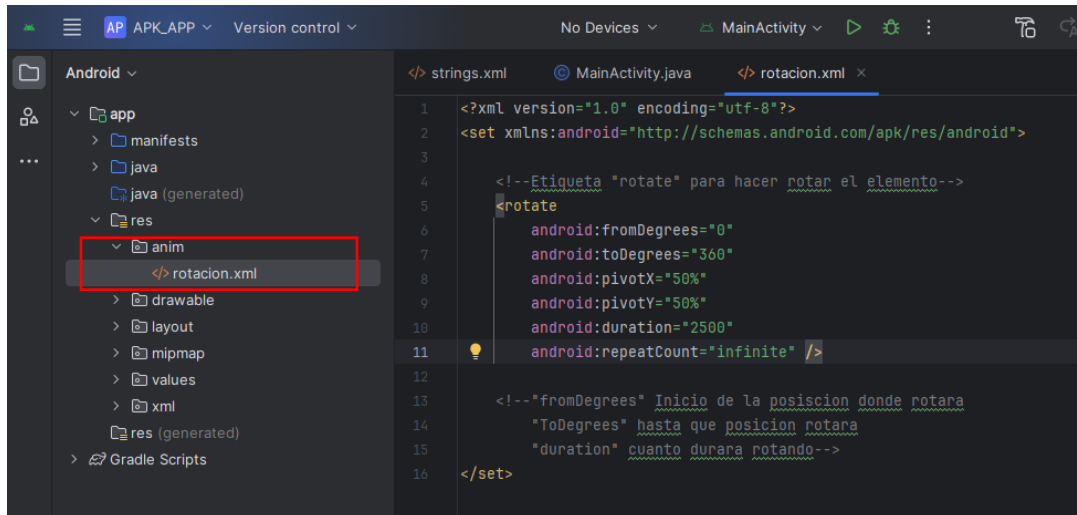
        Toast.makeText(getApplicationContext(), texto "Registrarse", Toast.LENGTH_SHORT).show();
        //Mostrar un mensaje al presionar el boton

        if(invisible[0]){ //Condicion en donde la variable invisible esta en "true"
            // si "invisible" es verdadero entonces no se mostrara el formulario
            correo2.setVisibility(View.INVISIBLE);
            contra2.setVisibility(View.INVISIBLE);
            invisible[0] = false; // Cambia a "false"
        } else{
            //En caso de que la variable sea "false" se mostrara el formulario
            correo2.setVisibility(View.VISIBLE);
            contra2.setVisibility(View.VISIBLE);
            invisible[0] = true; //Se cambia el valor a "true"
        }
    }
}

```

Aquí importe los objetos como “*TextView*” (textos) o “*Button*” (botones) desde el layout de mi app y se buscan los objetos gracias al “id” de cada objeto. Aquí hago una condición en donde al dar click a un botón se hace visible el formulario (en este caso).

Animaciones (En una imagen)

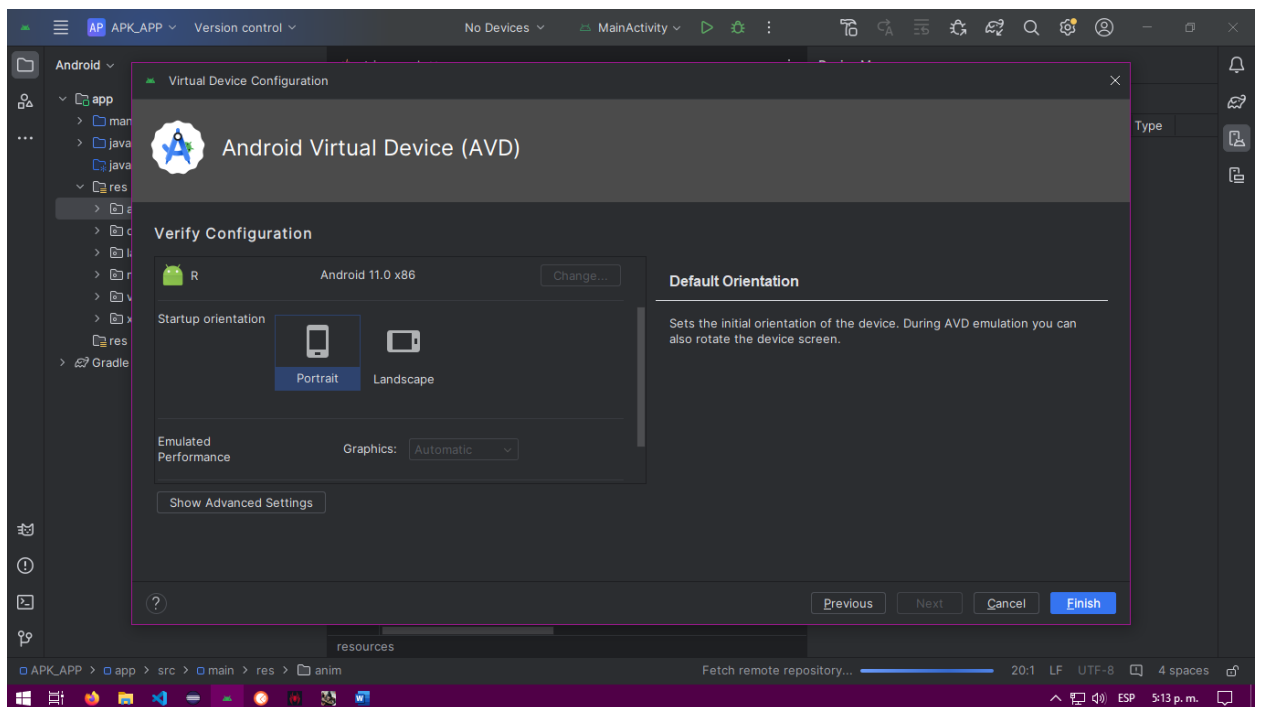
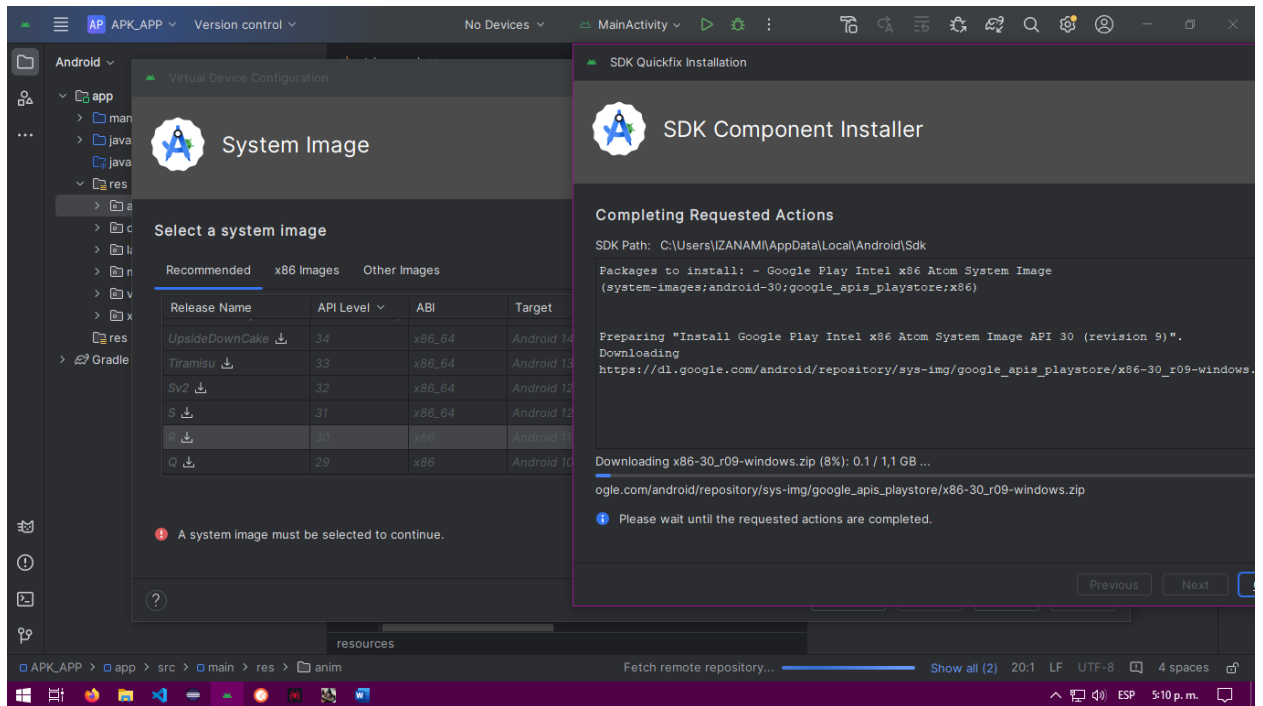


```
////////// ANIMACION DE ROTACION DE LA IMAGEN "E"

elxiting = findViewById(R.id.elxiting); //Se importa el objeto por su "ID"
Animation rotationAnimation = AnimationUtils.loadAnimation( context: this, R.anim.rotacion);
//Se carga la animacion desde el archivo "rotacion.xml"
elxiting.startAnimation(rotationAnimation); //Se inicializa la funcion
```

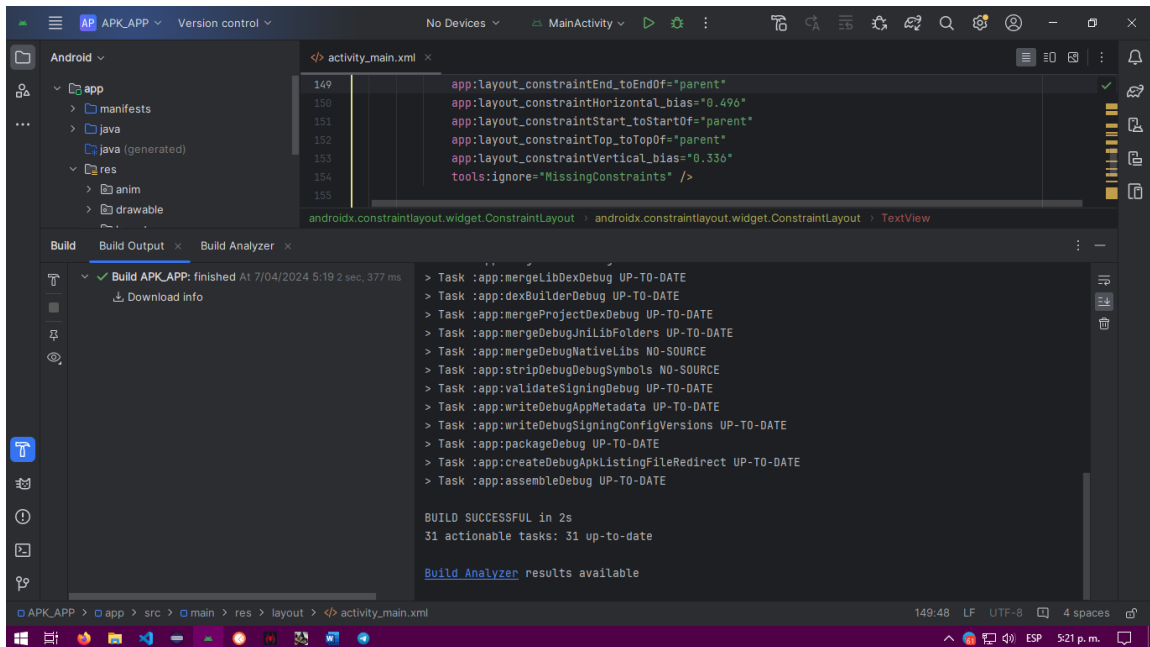
Aquí cree una carpeta para insertar mis animaciones, en este caso la animación que quise hacer fue de rotación. Se realiza primero que función va a tener este objeto, su punto de partida, hacia que lado va a rotar, su velocidad, etc. Luego se importa esa animación en el archivo principal de la app “*MainActivity.java*” para que la función se cumpla dentro de la aplicación.

Pruebas de la Aplicación



Se instala el emulador del Android que se requiera y sea universal para que pueda correr la aplicación.

Pantalla de la APP



Aquí se esta cargando el emulador de la app.

APP_APK

