SIMULACIONES

IV + IoT + DL

Industria 5.0

Diego L. Aristizábal Ramírez
Profesor, Facultad de Ciencias, Departamento de Física
Universidad Nacional de Colombia
Medellín

MI PRIMERA APP

Con Android Studio

2

Temas
✓ INTRODUCCIÓN
✓ MI PRIMERA APP: HOLA MUNDO
✓ TAREA
✓ REFERENCIAS

INTRODUCCIÓN

En este módulo se empezará con el reconocimiento del ambiente de desarrollo Android Studio. Adicionalmente se desarrollará la primera aplicación la cual servirá para probar la correcta configuración de este **ambiente de desarrollo integrado** (IDE, por sus siglas en inglés[1]).

MI PRIMERA APP: HOLA MUNDO

Lo primero será instalar el IDE Android Studio, **android-studio-2025.1.3.7-windows**, el cual se puede obtener en el siguiente link,

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

Una vez instalado seguir los pasos que se indican a continuación para realizar la primera aplicación.

Paso 1:

Ejecutar **Android Studio**. Se despliega la interfaz de la Figura 1.

[1] Integrated Development Environment: IDE.

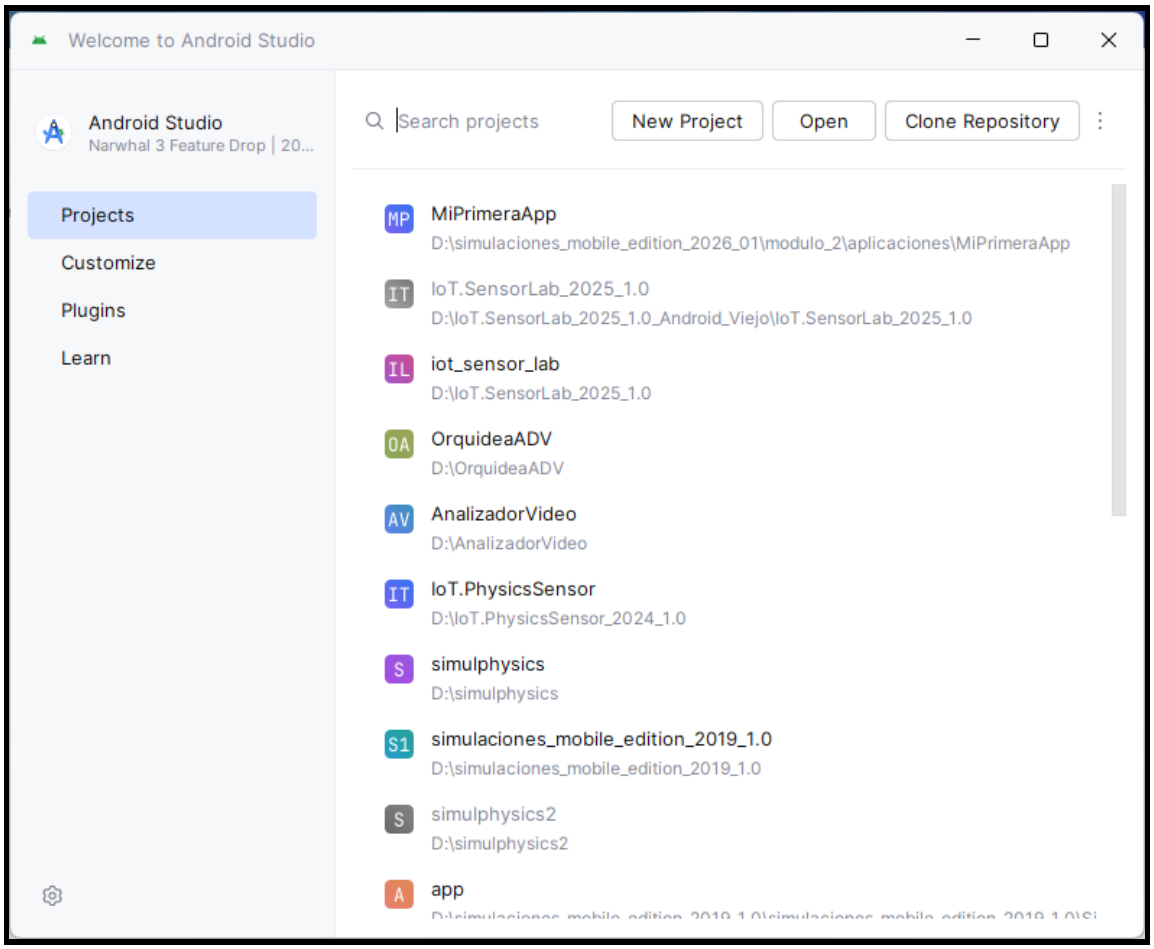

Figura 1

Paso 2:

Hacer clic en **New Project** para iniciar el proyecto. Se despliega la interfaz de la Figura 2.

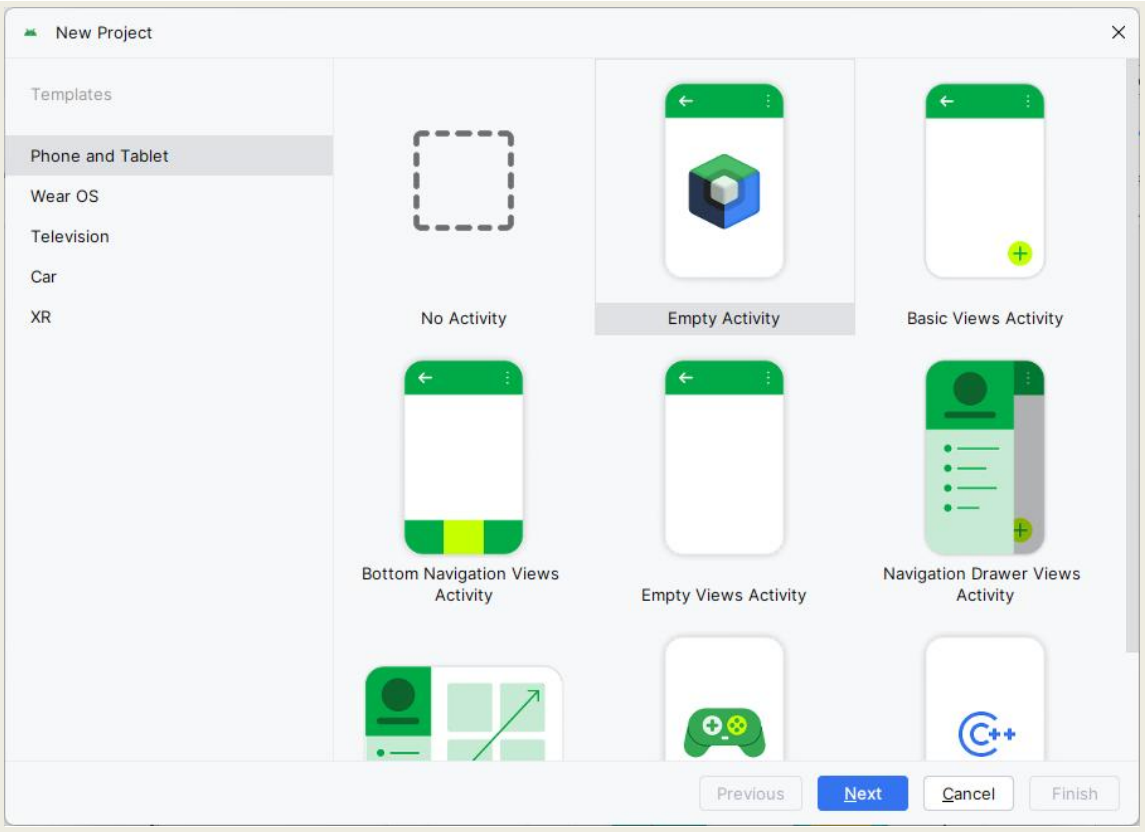

Figura 2

Paso 3:

En este curso se procurará que el código sea en su mayoría impuesto por el programador y minimizar la ayuda del **IDE**[2] para esto. Por ejemplo, las interfaces gráficas de usuario (**GUI**s, por sus siglas en inglés) se desarrollarán con código Java, a esto se le denomina, "**programaticamente**" ("**programmatically"**), es decir serán dinámicas. La otra forma de hacerlo es a través de archivos **XML** y en este caso serán estáticas, es decir se pueden cambiar sin tocar el código de la clase en Java. En definitiva, no se harán las **GUIs** con la ayuda del **IDE**, lo cual trae como consecuencia que se demorará más en el desarrollo de la aplicación, pero como ganancia se tendrá más control sobre el código lo cual facilitará que la aplicación se adapte mejor a las innumerables especificaciones en tamaños y resoluciones de los dispositivos Android.

Realizada la aclaración anterior se procederá a seleccionar **No Activity** y hacer clic en el botón **Next** lo cual desplegará la GUI de la a Figura 3. Llenar los campos respectivos. El nombre de la aplicación debe comenzar con mayúsculas. En este caso se le denominó

[2] **IDE** (Integrated Development Environment): **Entorno de Desarrollo Integrado**

MiPrimeraApp. Al dominio de la compañía se le denominó **com.curso_simulaciones**.Llenar el campo que elige el directorio donde se hospedará la aplicación. Luego el campo del lenguaje, en nuestro caso **Java**. Luego elegir la mínima API que soportará la app, en nuestro caso elegimos la API 24 que nos garantiza que se nuestra app se ejecutará aproximadamente en el 98.6 % de los dispositivos ANDROID actuales. Hacer clic en **Finish** y se despliega la interfaz de la Figura 4 (el proceso puede demorar un tiempo apreciable para la primera app mientras se acomoda bien el IDE).

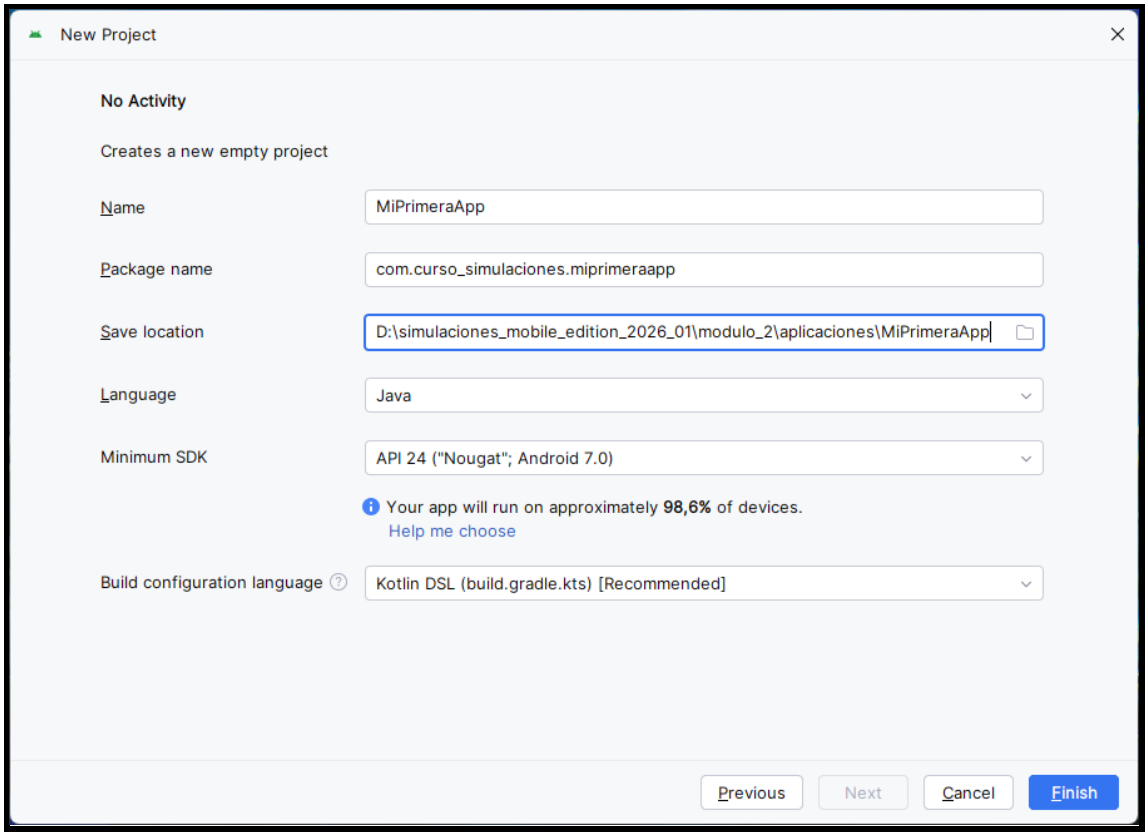

Figura 3

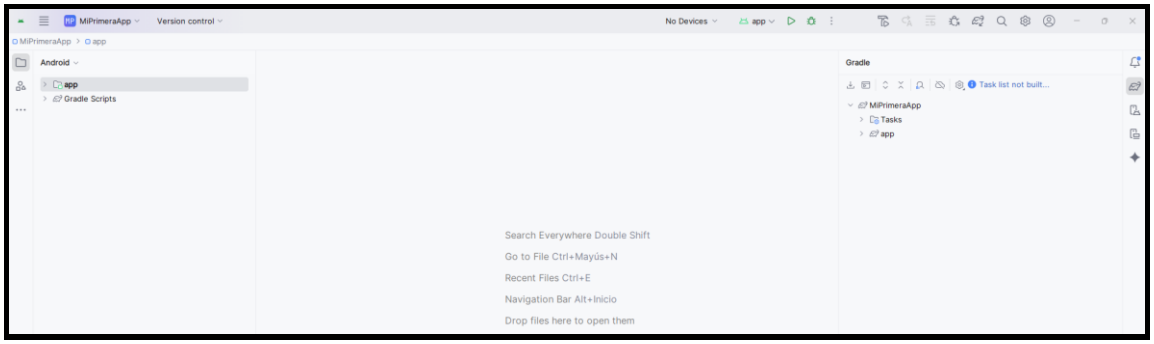

Figura 4

Paso 4:

Ubicarse sobre **java/com/curso_simulaciones>miprimeraapp**, y hacer clic derecho. Se despliega la interfaz de la Figura 5,

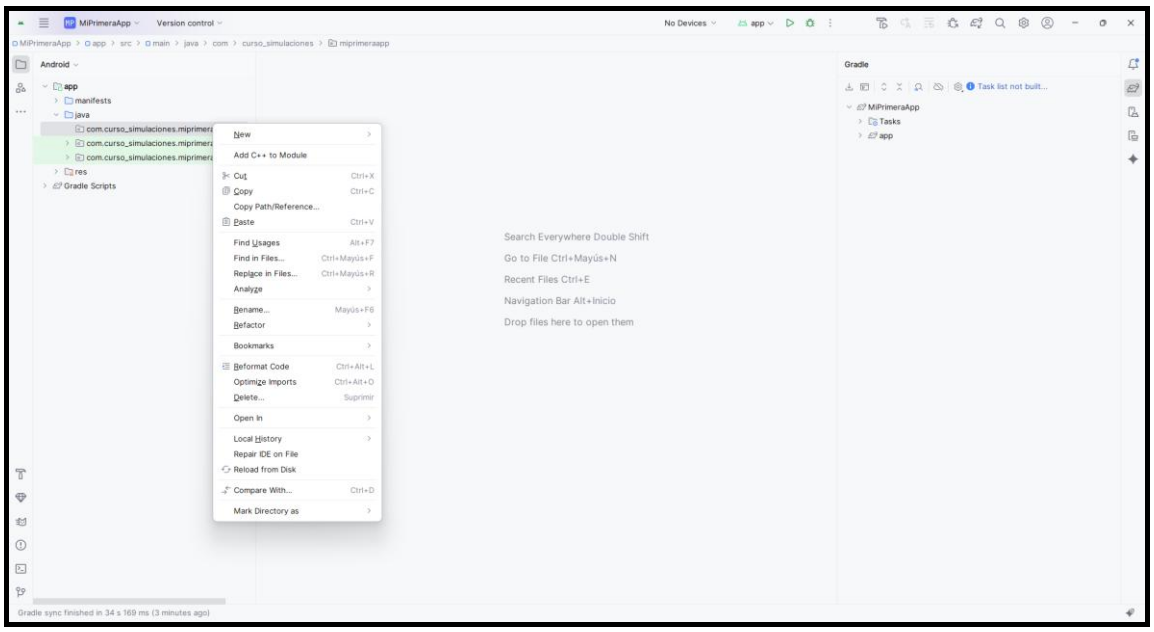

Figura 5

Seleccionar **New > Java Class**. Se despliega la interfaz de la Figura 6. En el campo escribir el nombre de la clase que se creará, en este caso se eligió el nombre **ActividadPrincipalMiPrimeraApp**. Hacer doble clic en **Class**. Se despliega la interfaz de la Figura 7.

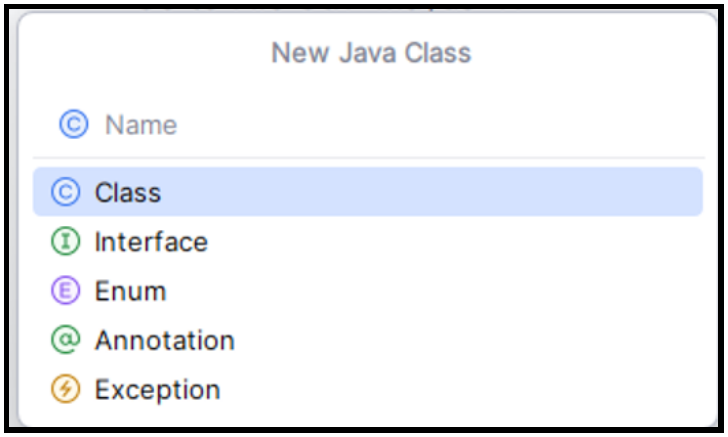

Figura 6

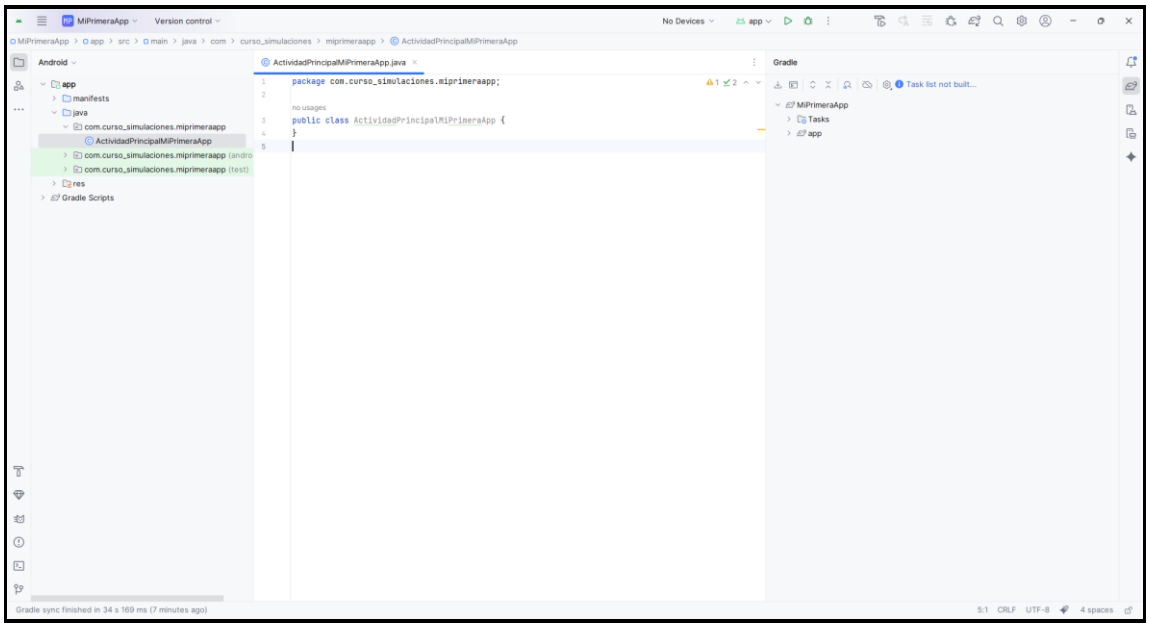

Figura 7

Paso 5:

Agregar el siguiente código, Figura 8:

```
extends Activity
```

Importar los recursos necesarios de la API de Android que contenga la clase **Activity**, Figura 8. Esto se logra haciendo clic en la tecla **Enter** mientras se presiona simultáneamente la tecla **Alt**. Hecho esto **Activity** cambia de color rojo a gris, Figura 9.

Figura 8

Figura 9

Paso 6:

Continuar completando el código de la clase: para esto utilizar el código que se presenta a continuación. Hacer el proceso de importación de los recursos necesarios de la API de Android para implementar este código: proceder tal como se hizo en el paso anterior (**Alt+Enter** ubicándose en donde aparezca código en rojo).

```
public class ActividadPrincipalMiPrimeraApp  extends Activity {


  private TextView cadena;

  protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);


     /*llamada al método para crear los elementos de la interfaz  gráfica
       de usuario (GUI)*/
     crearElementosGui();


    /*para informar cómo se debe adaptar la GUI a la pantalla del
    dispositivo*/
    ViewGroup.LayoutParams parametro_layout_principal = new
    ViewGroup.LayoutParams(ViewGroup.LayoutParams.MATCH_PARENT,
    ViewGroup.LayoutParams.MATCH_PARENT);

   /*pegar al contenedor la GUI: en el argumento se está llamando al método
     crearGui()*/
     this.setContentView(crearGui(), parametro_layout_principal);


    }


    //crear los objetos de la interfaz gráfica de usuario (GUI)
    private void crearElementosGui(){

        cadena=new TextView(this);
        cadena.setTextSize(TypedValue.COMPLEX_UNIT_SP, 12);
        cadena.setTextColor(Color.YELLOW);
        cadena.setText("HOLA MUNDO");

    }


    //organizar la distribución de los objetos de la GUI usando administradores de
diseño
    private LinearLayout crearGui(){
```

```
        //administrador de diseño
        LinearLayout linear_principal = new LinearLayout(this);
        linear_principal.setOrientation(LinearLayout.VERTICAL);
        linear principal.setGravity(Gravity.CENTER_HORIZONTAL);
        linear_principal.setGravity(Gravity.FILL);
        linear_principal.setBackgroundColor(Color.BLUE);


        //pegar el objeto cadena (es tipo TextView)
        linear principal.addView(cadena);


        return linear_principal;

    }


    }
```


Paso 7:

Ahora es necesario agregar código en el **archivo manifiesto.** Hacer clic en el archivo **app.manifests> AndroidManifest.xml.** En el código desplegado agregar código de tal forma que el archivo quede como el siguiente. Ver Figura 10.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.curso_simulaciones.miprimeraapp">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MiPrimeraApp"
        tools:targetApi="31" >

    <activity

       android:name="com.curso_simulaciones.miprimeraapp.ActividadPrincipalMiPrimeraApp"
       android:exported="true">

        <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
       </intent-filter>
    </activity>

    </application>

    </manifest>
```

Figura 10

El objetivo de este código agregado es informar cuál es la clase que arrancará la ejecución de la aplicación.

Ya se está en condiciones de proceder a **compilar** y **ejecutar** la aplicación.

Paso 11:

Se procederá ahora a compilar la aplicación y a ejecutarla. Esto se puede hacer usando un emulador de dispositivo Android o el dispositivo mismo. En el curso se adoptará el segundo camino ya que son pruebas reales en el hardware. Para lograr esto es necesario cambiar la configuración del dispositivo móvil ANDROID (tableta o celular) al denominado "**opción de desarrollador**". Para lograr esto seguir las siguientes instrucciones:

A partir de Android 4.2 (Jelly Bean) las **opciones para desarrolladores** vienen por defecto ocultas. Google ha preferido que sólo los desarrolladores tengan acceso a esas opciones para evitar que un usuario medio entre en su menú y active o desactive opciones que provoquen un mal funcionamiento del terminal.

Activar las opciones de desarrollo es muy fácil, tan sólo se tiene que ir a **Ajustes > Información del dispositivo** y **pulsar siete veces sobre el número de compilación**. Una vez hecho saldrá el mensaje **¡Ahora eres un desarrollador!** y mostrará en los Ajustes ese apartado.

Luego proceder a activar **Depuración USB**. En estos momentos se puede conectar el dispositivo por el puerto USB del PC y Android Studio **lo reconocerá**.

Paso 12:

Conectar el dispositivo por un puerto USB del PC. Proceder a la compilación y ejecución haciendo clic en el botón de la barra superior de herramientas de Android Studio que está señalado con un círculo rojo en la Figura 11.

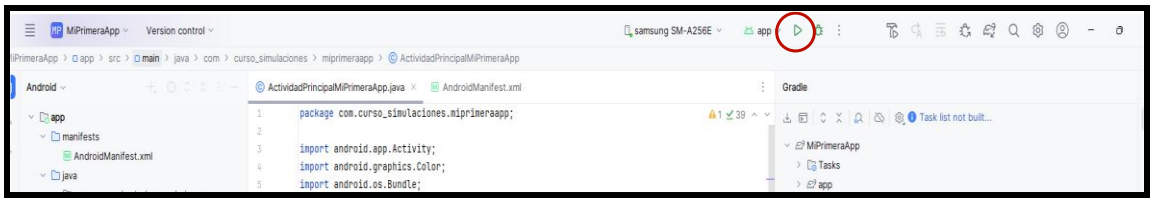

Figura 11

Pasados unos segundos en el dispositivo móvil deberá aparecer el resultado de la ejecución del programa, Figura 12. Además, en el menú de aplicaciones del dispositivo debe aparecer el icono de la aplicación, Figura 13 (ver dentro del rectángulo rojo), en este caso es el icono de ANROID (más adelante se mostrará cómo cambiarlo por uno diseñado).

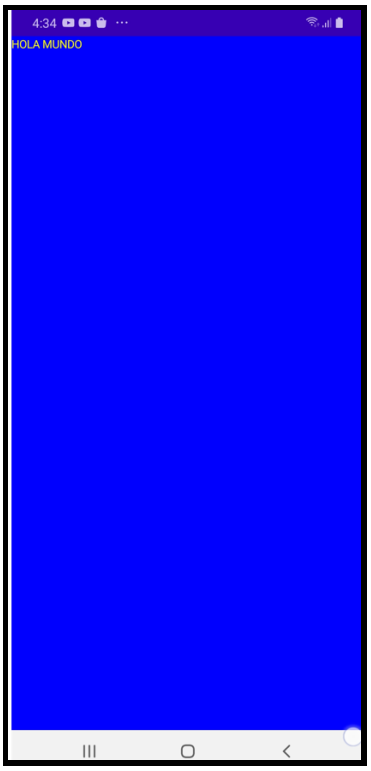

Figura 12

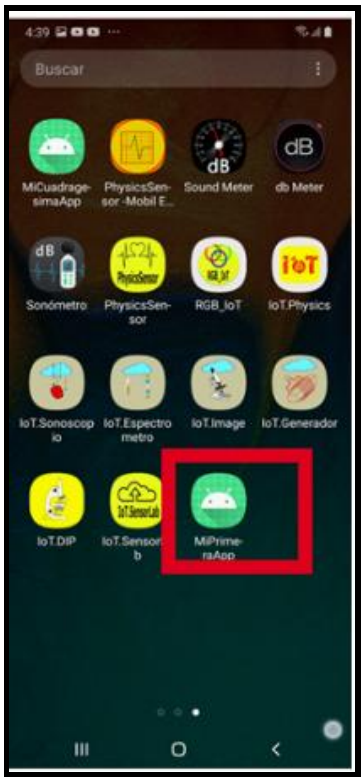

Figura 13

Paso 13: Generar el ejecutable

Ir a la opción de generar el apk, Figuras 14 y 15.

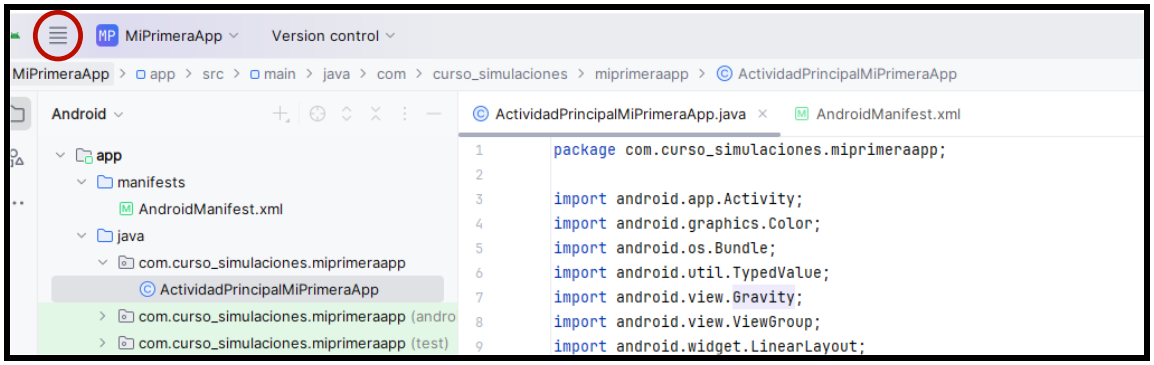

Figura 14

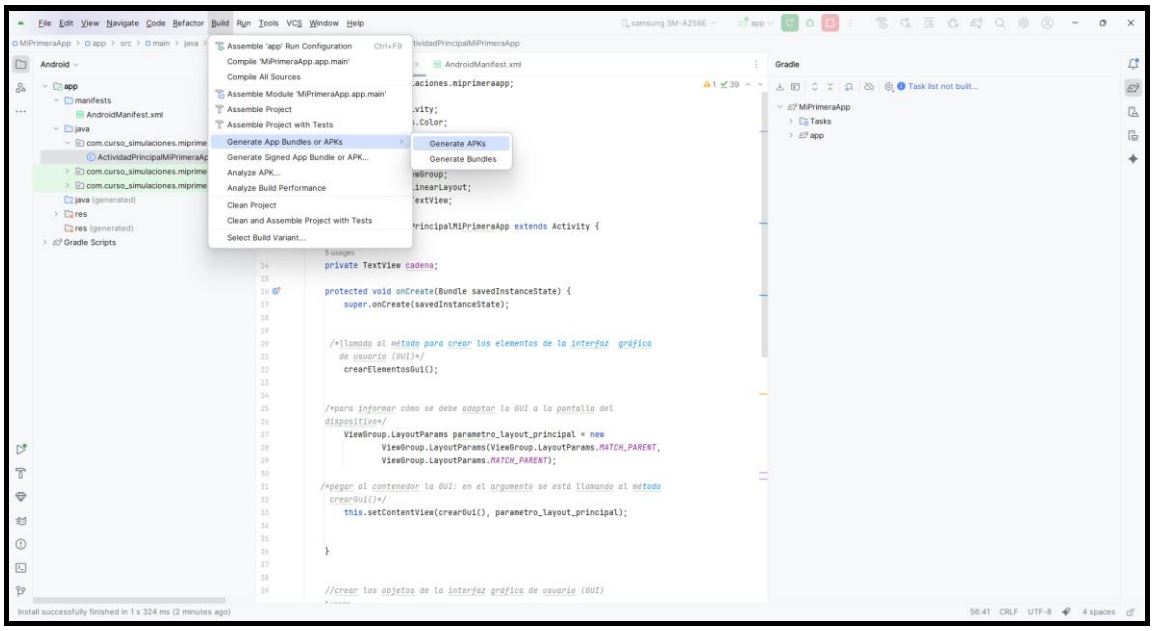

Figura 15

Una vez generado este archivo se encontará en la carpeta,

```
App > build > outputs > apk >debug >
```

Con el nombre **app-debug.apk**

Si se desea se puede cambiar el nombre (es lo mejor), por ejemplo,

`Mi_primera_app_version_2022_1.0.apk`

Este archivo se pasa al dispositivo móvil ANDROID en el que se desea instalar. Una vez este allí, se ejecuta el archivo y se van aceptando los permisos requeridos. **Listo!**

Se recomienda siempre instalarlo en diferentes dispositivos para que pase las pruebas de buen funcionamiento, esto es: que las interfaces gráficas se vean en todos de la misma forma, que la app no se bloquee en algunos, etc.

TAREA

Estudiar y desarrollar el módulo #3.

FIN

REFERENCIAS

- ✓ Franco G. A (2000)., **Curso de Lenguaje Java, Universidad** del País Vasco. http://www.sc.ehu.es/sbweb/fisica/curso.htm. Consultada [Septiembre de 2025].

- ✓ Android Studio. https://developer.android.com/studio/. Consultada [Septiembre de 2025].