Actividad 3.2. Splash Screen

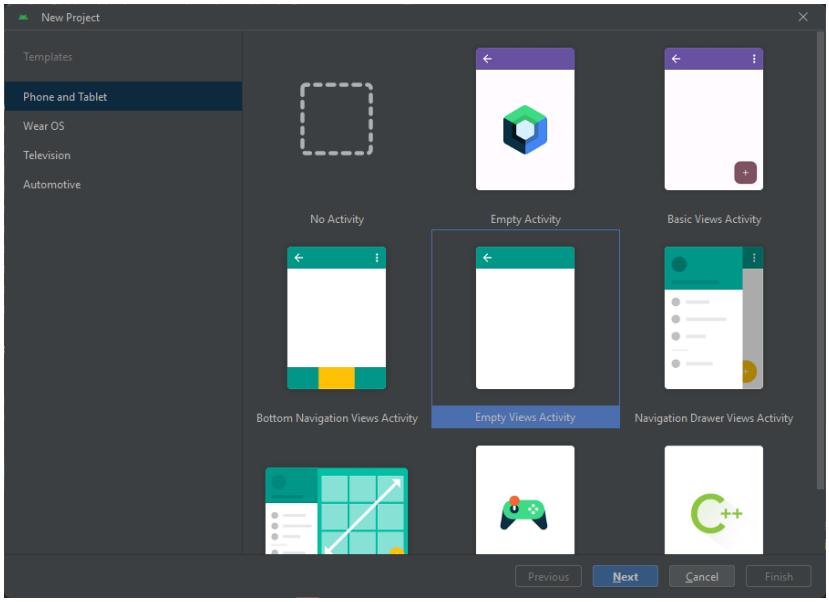
Splash Screen es la primera pantalla visible para el usuario cuando se inicia la aplicación. La pantalla de bienvenida es una de las pantallas más vitales de la aplicación, ya que es la primera experiencia del usuario con la aplicación.

Al abrir la aplicación hay un momento de "nada" ya que la aplicación aún no está en memoria a esto se le llama "cold start", el administrador de ventanas intenta dibujar una UI de marcador de posición utilizando elementos del tema de la aplicación como windowBackground. Entonces, en lugar de mostrar el valor predeterminado windowBackground (generalmente blanco o negro).

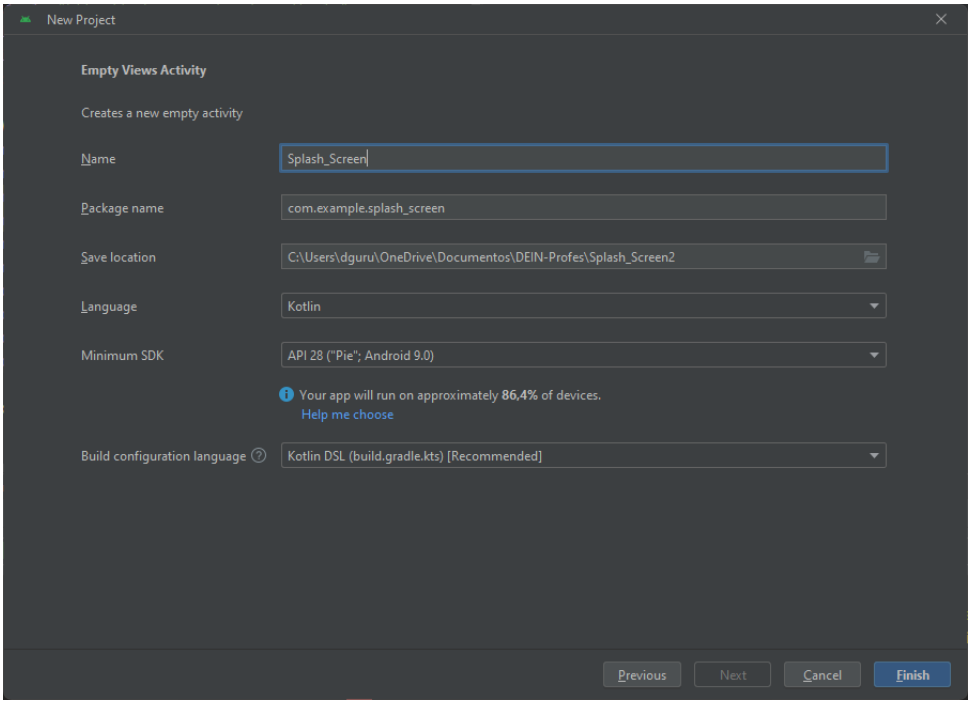
Esta puede ser cambiada a un dibujo personalizado que muestre su pantalla de inicio. De esta manera, la pantalla de inicio solo se muestra cuando es necesario y no ralentiza a los usuarios.

Vamos a crear una "ventana de presentación" y posteriormente, a los 5 sg en este caso, que aparezca la ventana principal.

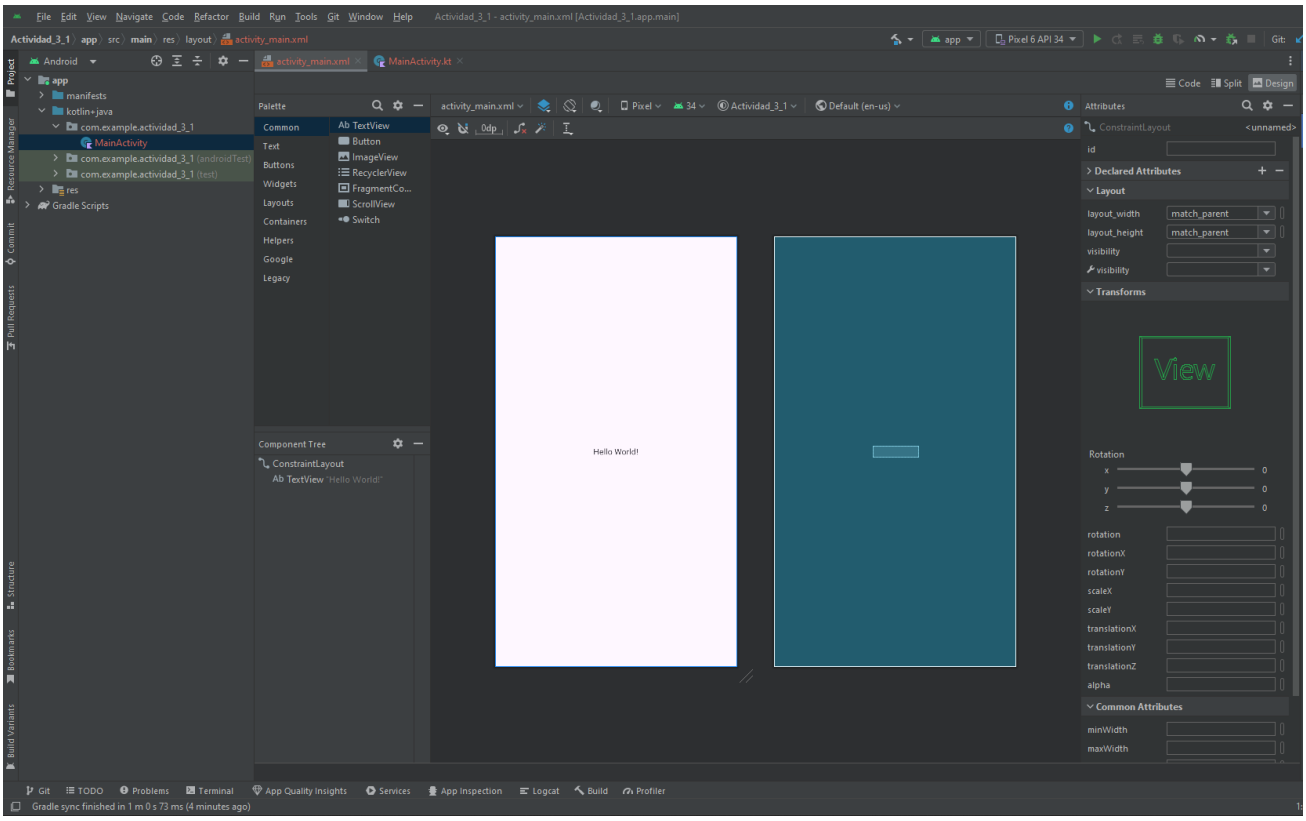
1. **Primero crea un nuevo proyecto en Android Studio del tipo "Empty Views Activity"**



Llámale "**Splash_Sreen**", el Lenguaje "Kotlin" y aunque tenemos diversas plataformas y APIs en las que utilizar nuestra aplicación, nosotros nos centraremos en aplicaciones para teléfonos y tablets, en cuyo caso tan sólo tendremos que seleccionar la API mínima (es decir, la versión mínima de Android) que soportará la aplicación. Android 9.0. como versión mínima (API 28):

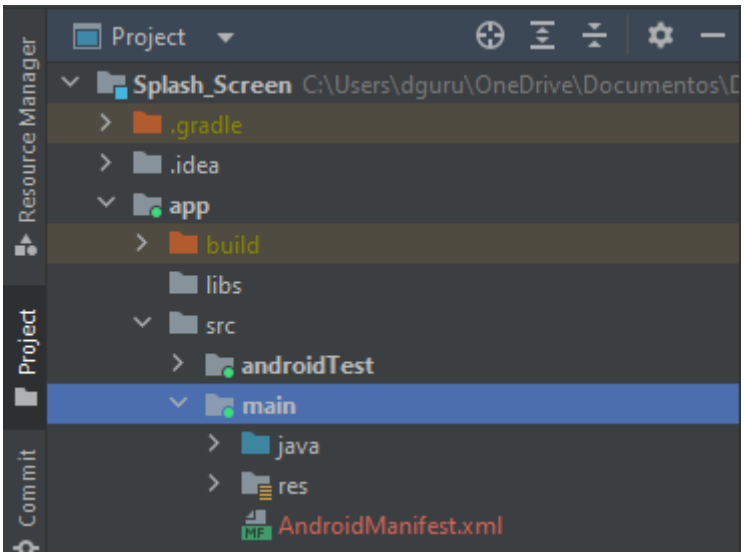


Una vez configurado todo pulsamos el botón **Finish** y Android Studio creará por nosotros toda la estructura del proyecto y los elementos indispensables que debe contener. Si todo va bien aparecerá la pantalla principal de Android Studio con el nuevo proyecto creado:



En la parte izquierda, podemos observar todos los elementos creados inicialmente para el nuevo proyecto Android, sin embargo, por defecto los vemos de una forma un tanto peculiar que podría llevarnos a confusión. Para entender mejor la estructura del proyecto vamos a cambiar momentáneamente la forma en la que Android Studio nos la muestra. Para ello, pulsaremos sobre la lista desplegable situada en la parte superior izquierda, y cambiaremos la vista de proyecto al modo **"Project".**

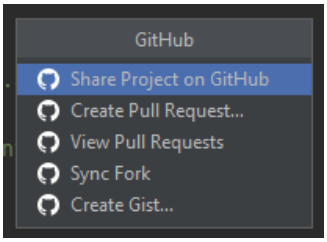
Tras hacer esto, la estructura del proyecto cambia un poco de aspecto y pasa a ser como se observa en la siguiente imagen:

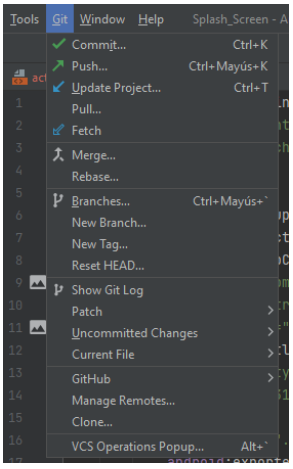


En los siguientes apartados describiremos los elementos principales de esta estructura.

Lo primero que debemos distinguir son los conceptos de **proyecto y módulo**. La entidad proyecto es única, y engloba a todos los demás elementos. Dentro de un proyecto podemos incluir varios módulos, que pueden representar aplicaciones distintas, versiones diferentes de una misma aplicación, o distintos componentes de un sistema (aplicación móvil, aplicación servidor, librerías, ...). En la mayoría de los casos, trabajaremos con un proyecto que contendrá un sólo módulo correspondiente a nuestra aplicación principal. Por ejemplo, en este caso que estamos creando tenemos el proyecto "Splash_Screen" que contiene al módulo "app" que contendrá todo el software de la aplicación de ejemplo.

Configúrate y recuerda ir guardando el Proyecto mientras los vas realizando (y configura GIT).

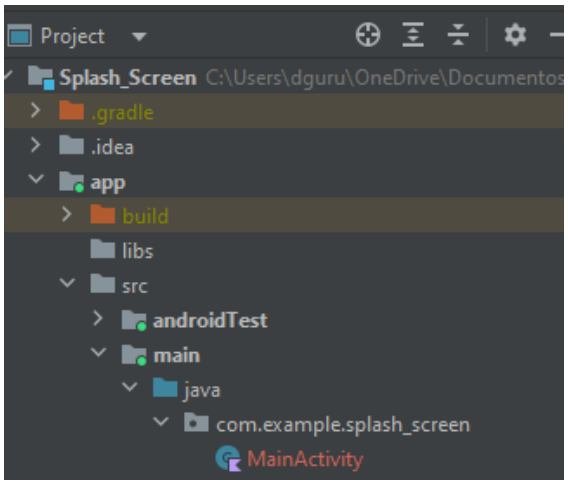




2. Carpetas, archivos y contenidos principales de nuestro módulo principal.

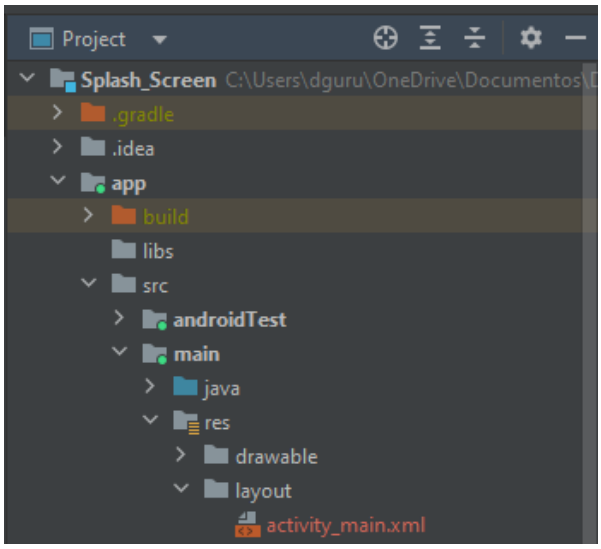
1. ***Carpeta /app/src/main/java***

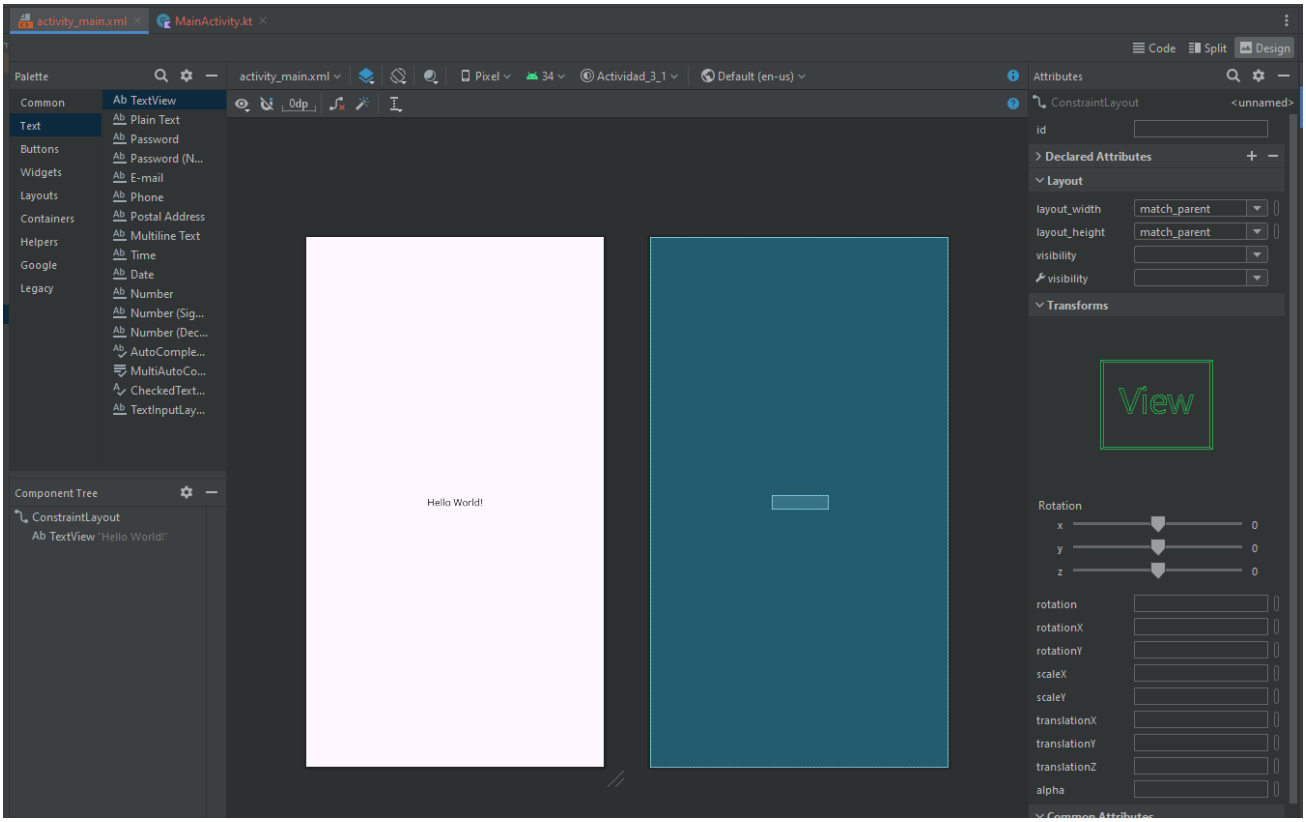
Esta carpeta contendrá todo el código fuente de la aplicación, clases auxiliares, etc. Inicialmente, Android Studio creará por nosotros el código básico de la pantalla (actividad o activity) principal de la aplicación, que recordemos que en nuestro caso era **MainActivity**



2. ***Carpeta /app/src/main/res/***

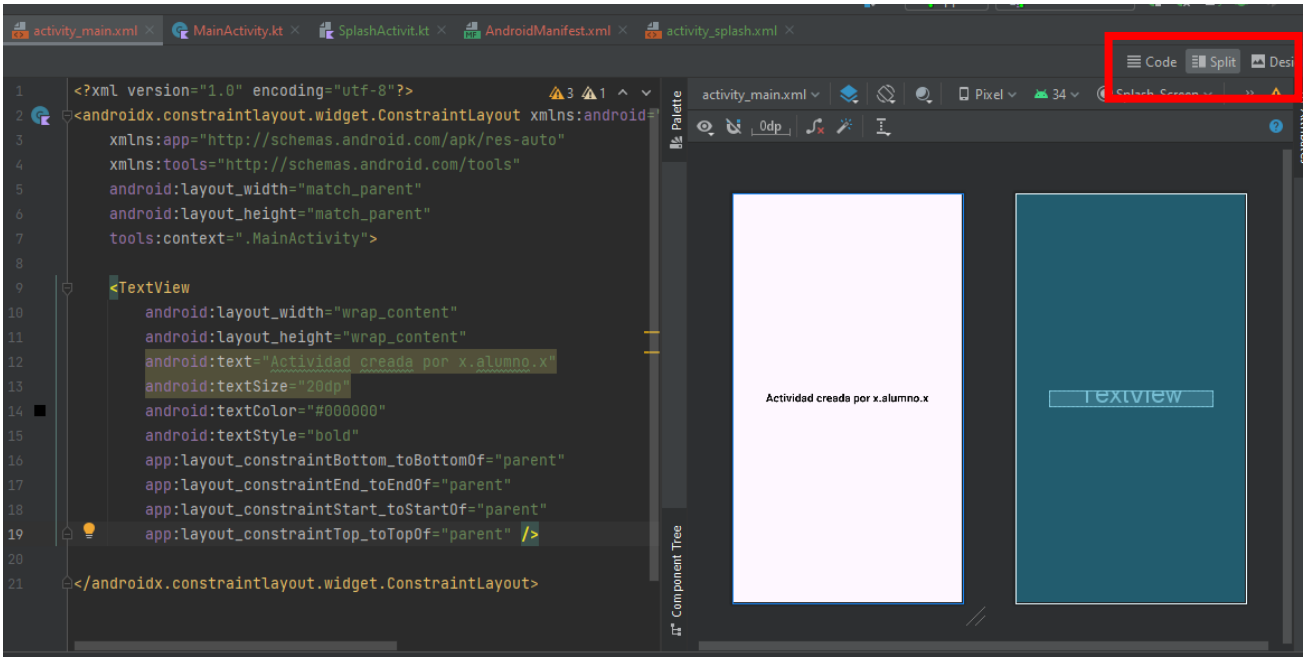
Contiene todos los ficheros de recursos necesarios para el proyecto: imágenes, layouts, cadenas de texto, etc. Los diferentes tipos de recursos se pueden distribuir entre diversas subcarpetas. Entre los recursos creados por defecto cabe destacar los **layouts**, en nuestro caso sólo tendremos por ahora el llamado "**activity_main.xml**", que contienen la definición de la interfaz gráfica de la pantalla principal de la aplicación. Si hacemos doble clic sobre este fichero Android Studio nos mostrará esta interfaz en su editor gráfico, y como podremos comprobar, en principio contiene tan sólo una etiqueta de texto con el mensaje "Hello World!"





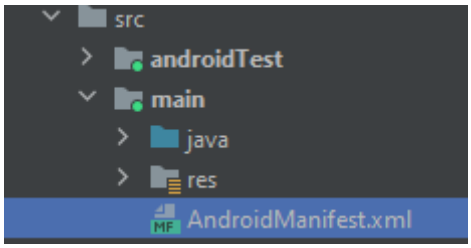
Ventana gráfica de "activity_main.xml

Pulsando sobre las pestañas superior derecha "Design", "Code" y "Split" podremos alternar entre el editor gráfico (tipo arrastrar-y soltar), mostrado en la imagen anterior, y el editor XML que se muestra en la imagen siguiente:



3. ***Fichero /app/src/main/AndroidManifest.xml***

Contiene la definición en XML de muchos de los aspectos principales de la aplicación, como por ejemplo su identificación (nombre, icono, …), sus componentes (pantallas, servicios, …), o los permisos necesarios para su ejecución. Veremos más adelante más detalles de este fichero.



3. Creación de las Activities (ventanas)

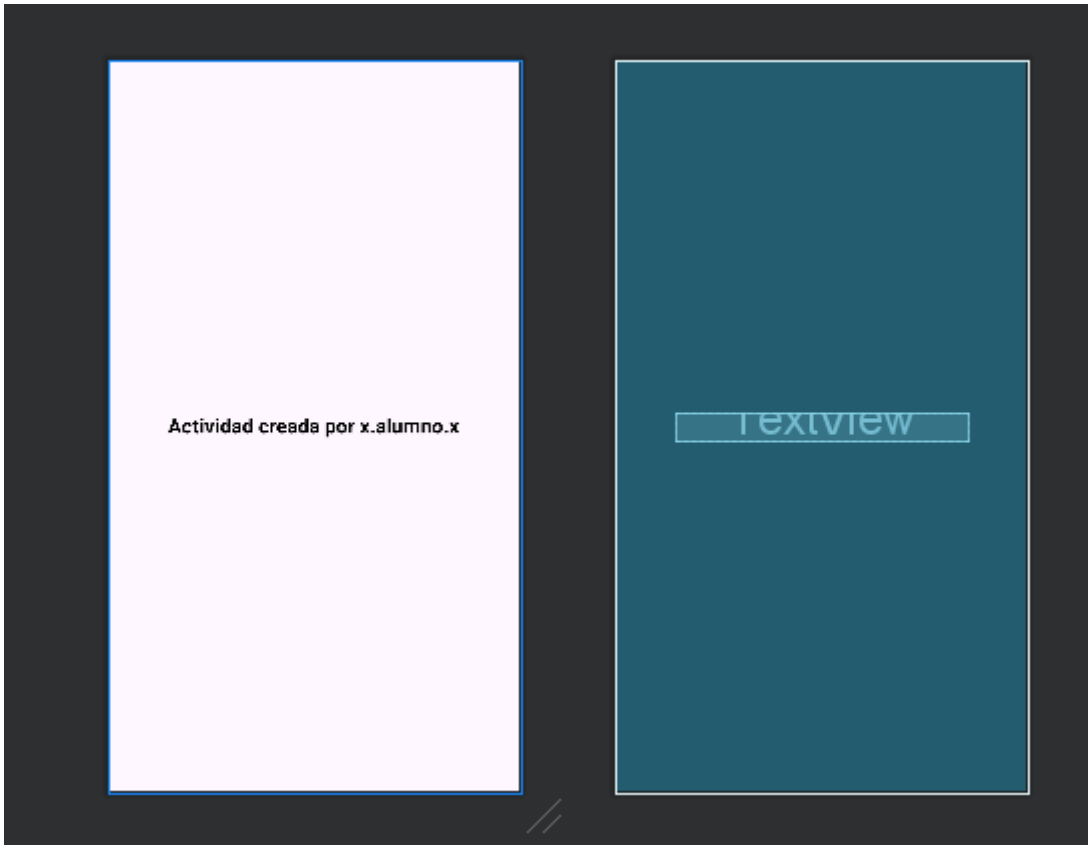
Primero crea la actividad principal (MainActivity) que será la actividad a la que se dirigirá después de que la pantalla de bienvenida desaparezca.

En la línea de “android:text”, sustituye “x.alumno.x” por tu nombre. Luego configura el tamaño, negrita, y posición del texto:

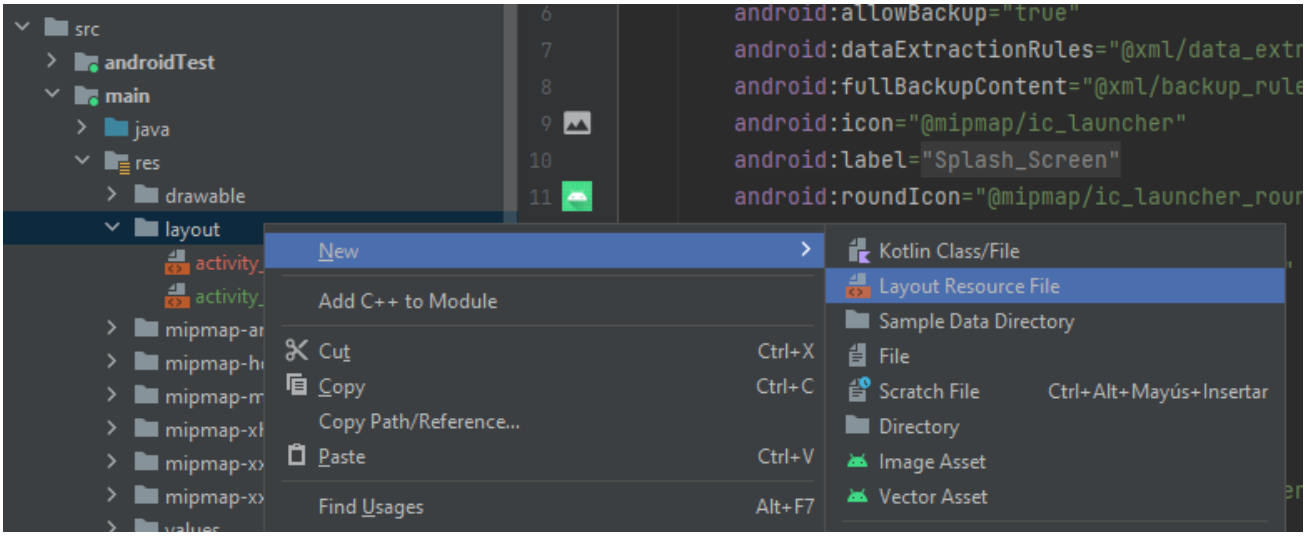
activity_main.xml | MainActivity.kt | SplashActivit.kt | AndroidManifest.xml | activity_splash.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    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"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Actividad creada por x.alumno.x"
        android:textSize="20dp"
        android:textColor="#000000"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

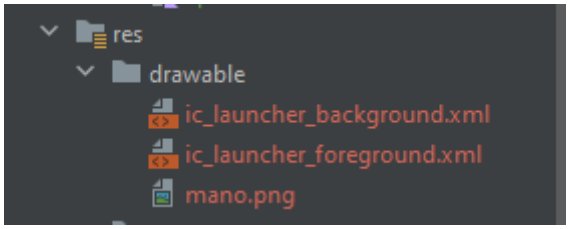
Debería quedarte así:



Ahora tienes que crear la nueva “activity” (ventana) en la carpeta “layout”. En la carpeta “layout” vete a “new” y selecciona “Layout Resource File”, y llámale “activity_splash”



Antes de pasar a configurar esta activity (será la ventana de bienvenida” descarga una imagen/dibujo de una mano saludando y déjalo en la carpeta “res/drawable” (tienes que ir a la carpeta del proyecto en Windows y luego recargar esa carpeta en Android Studio). Y ya deberías ver ese archivo:



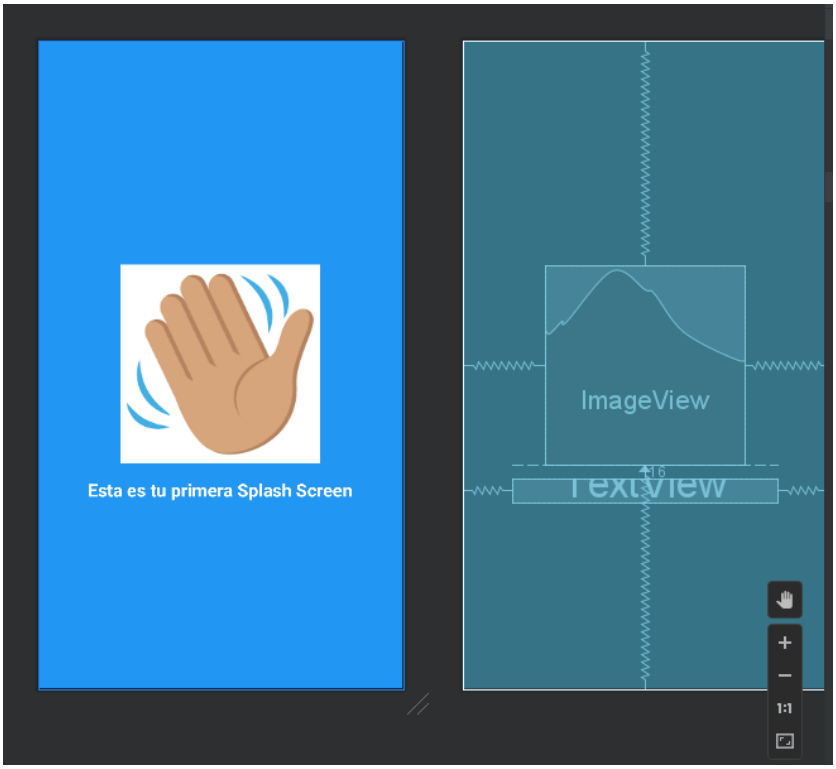
Ahora abre el archivo activity_splash.xml en la carpeta res/layout y define la interfaz de usuario. Incluye un **TextView** para el texto, un **ImageView** para la imagen de la mano y configura el fondo con el color azul. Todo integrado dentro de un mismo "**RelativeLayout"**

En el TextView, incluye el texto "Esta es tu primera Splash Sreen", negrita, centrodo y tamaño **20sp** (se refiere a 20 puntos de escala (scale-independent pixels), son unidades de medida que escalan de acuerdo con la configuración de tamaño de fuente del dispositivo del usuario. Esto ayuda a garantizar que el tamaño del texto se vea adecuado en diferentes dispositivos con diferentes configuraciones de pantalla y tamaños de fuente).

activity_main.xml | MainActivity.kt | SplashActivit.kt | AndroidManifest.xml | activity_splash.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <!-- activity_splash.xml -->
    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#2196F3">

        <ImageView
            android:id="@+id/imageViewHand"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/mano"
            android:layout_centerInParent="true"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Esta es tu primera Splash Screen"
            android:textSize="20sp"
            android:textColor="#FFFFFF"
            android:textStyle="bold"
            android:layout_below="@id/imageViewHand"
            android:layout_marginTop="16dp"
            android:layout_centerHorizontal="true" /><!-- Ajusta esta linea -->

    </RelativeLayout>

</androidx.constraintlayout.widget.ConstraintLayout>
```

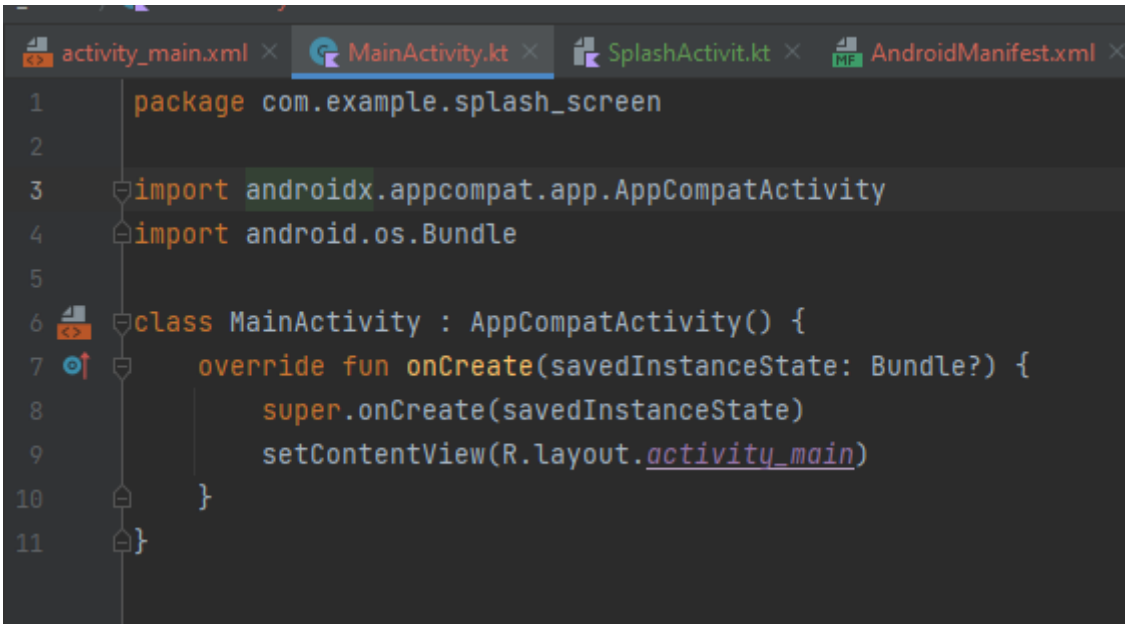
Debería quedar algo así:



4. Ahora crearemos la lógica para que funcione la App

Ahora, crearemos la lógica para que la pantalla de bienvenida aparezca durante 5 segundos antes de pasar a la pantalla principal.

- Primero abre el archivo MainActivity.kt, para revisarlo:



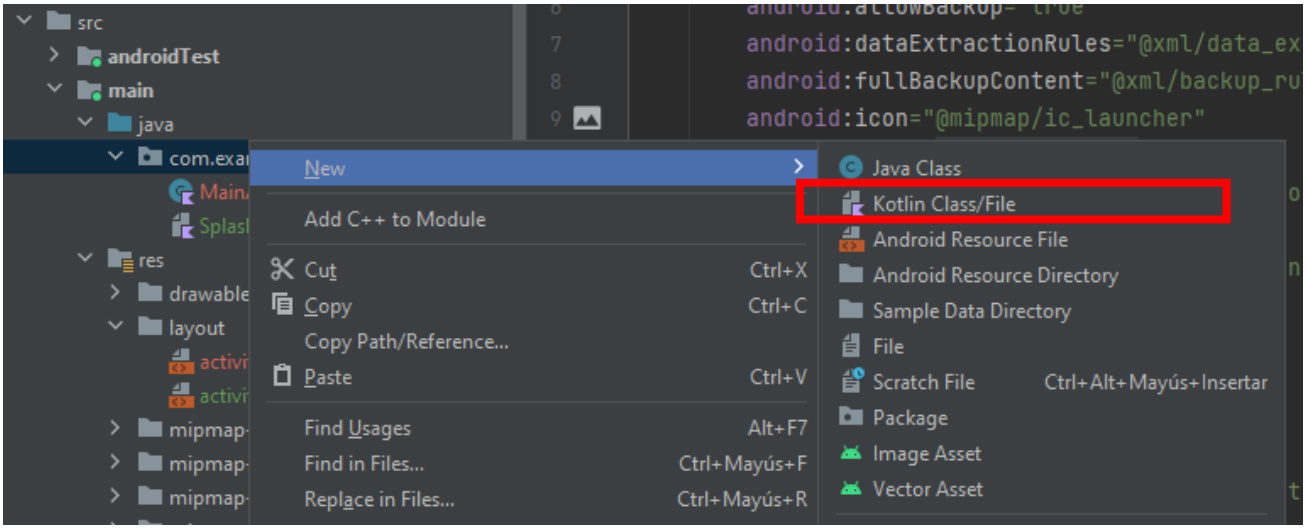
El método "**onCreate**" implementa la lógica de inicio de una actividad (su ciclo de vida) y se llama cuando la actividad se crea por primera vez. Con "override" se está reemplazando o sobrescribiendo el método "oncreate" y guardando los datos del estado anterior de la actividad.

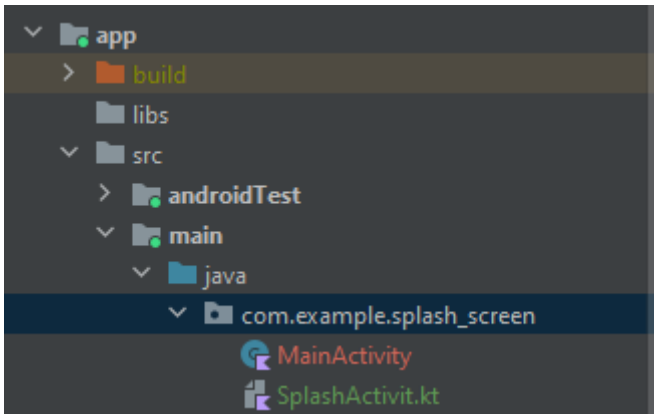
Con "**setContentView(R.layout.activity_main)**" se establece la interfaz de usuario de la actividad actual, utilizando el diseño definido en el archivo XML activity_main.xml (ubicado en la ruta: **/res/layout**). Este archivo XML contiene la estructura y los elementos visuales (botones, textos, imágenes, etc.) que formarán la interfaz gráfica de la aplicación cuando se ejecute.

Este archivo no lo vamos a modificar de momento.

- Ahora, tienes que crear un nuevo archivo Kotlin para establecer la lógica de la activity_splash.

  Para ello vete a las carpeta main/java/caom.example.splash_screen y crea el nuevo archivo, llámale "SplashActivity":





Ahora, crea la lógica para que la pantalla de bienvenida aparezca durante 5 segundos (5000 milisegundo) antes de pasar a la pantalla principal. Abre el archivo SplashActivity.kt y agrega el siguiente código:

activity_main.xml | MainActivity.kt | SplashActivit.kt | AndroidManifest.xml | activity_splash.xml

```kotlin
package com.example.splash_screen

import android.content.Intent
import android.os.Bundle
import android.os.Handler
import android.os.Looper
import androidx.appcompat.app.AppCompatActivity

new *
class SplashActivity : AppCompatActivity() {

    new *
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_splash)

        // Usar el constructor de Handler que acepta un Looper
        Handler(Looper.getMainLooper()).postDelayed({
            // Crea un Intent para iniciar la actividad principal
            val intent = Intent( packageContext: this@SplashActivity, MainActivity::class.java)
            startActivity(intent)
            finish()
        }, delayMillis: 5000) // 5000 milisegundos = 5 segundos
    }
}
```

Para ello, se ha utilizado **Handler**, que es una clase en Android que se utiliza para programar tareas (o mensajes) para ser ejecutadas en un hilo (thread) específico. En este caso lo hemos utilizado para crear un looper (estructura de control que repite un conjunto de instrucciones hasta que se cumple una condición específica (que pasen los 5 segundos).

Y para ello hemos creado también un **intent** (recuerda que un "Intent" es un objeto que proporciona una descripción de una operación a realizar, como iniciar una actividad (Activity), enviar o recibir datos entre componentes de una aplicación, iniciar servicios, etc)

- El próximo paso va a ser hacer que la actividad “SplashActivity” esté registrada correctamente en el archivo **AndroidManifest.xml** y que esta sea la actividad de inicio.

  Para ello tienes que modificar ese archivo cambiando los datos de la “activity_main” que es la que está configurada por defecto para inicializarse por los datos de la “activity_splash”.

  También tiene que estar registrada la “actividad” de la “MainActivity”

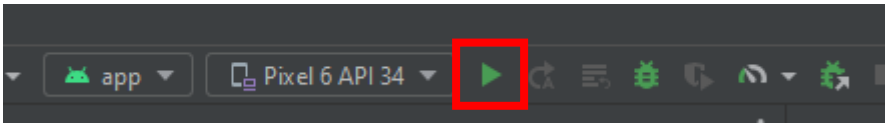
  Tiene que quedar así:

activity_main.xml | MainActivity.kt | SplashActivit.kt | AndroidManifest.xml | activity_splash.xml

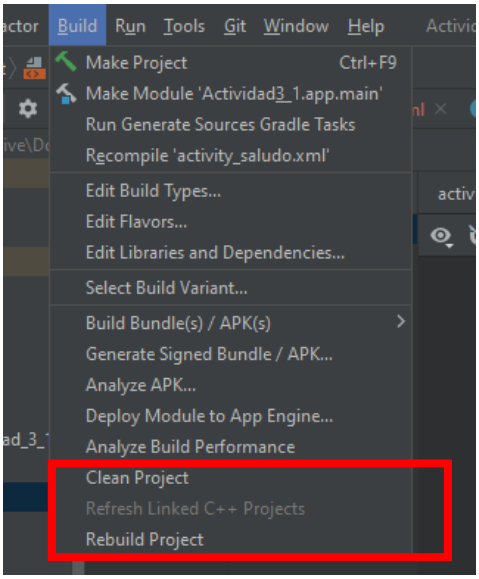
```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="Splash_Screen"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Splash_Screen"
        tools:targetApi="31">
        <activity
            android:name=".SplashActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

        <activity
            android:name=".MainActivity"
            android:label="@string/app_name">
        </activity>

    </application>

</manifest>
```

5. Compilar y probar el funcionamiento de la App en un dispositivo virtual

Para ver el aspecto y el comportamiento de tu app en un dispositivo, debes compilarla y ejecutarla. Android Studio configura proyectos para que puedas implementar tu app en un dispositivo virtual o físico. Si tuviéramos un dispositivo configurado podríamos ejecutar “Run App” para probar la App:



También es importante utilizar y conocer las opciones “Clean Project” y rebuild Project que nos ayudan a compilar el proyecto sin problemas (o ayudarnos a solucionarlos)

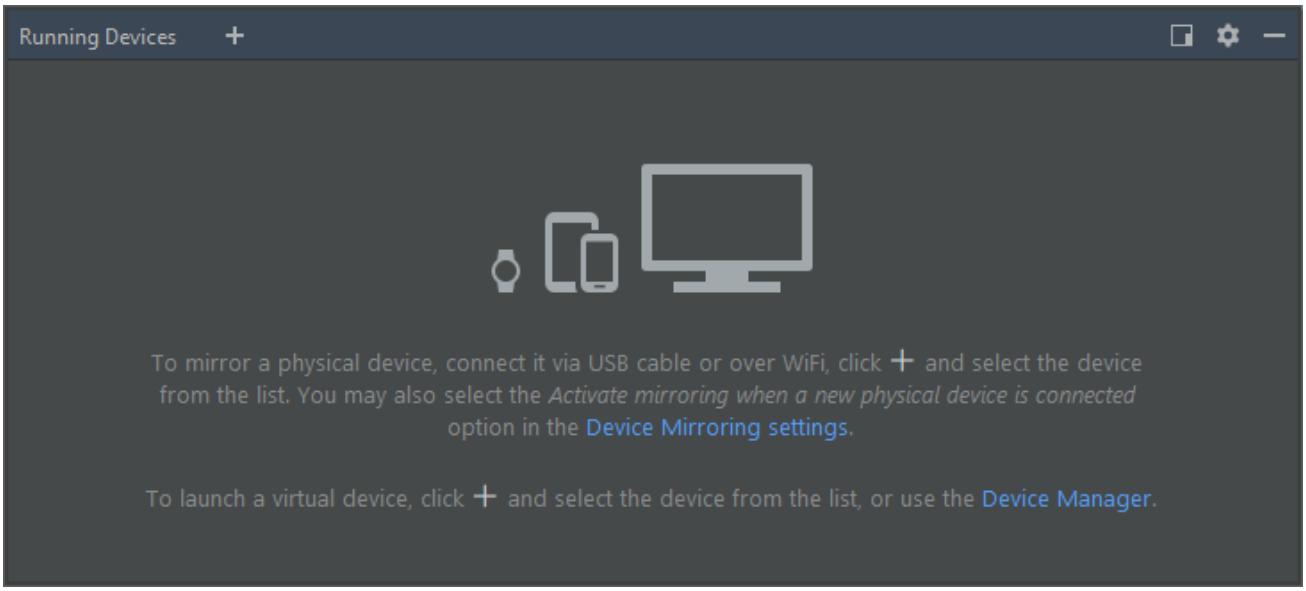


Clean y Rebuild Project

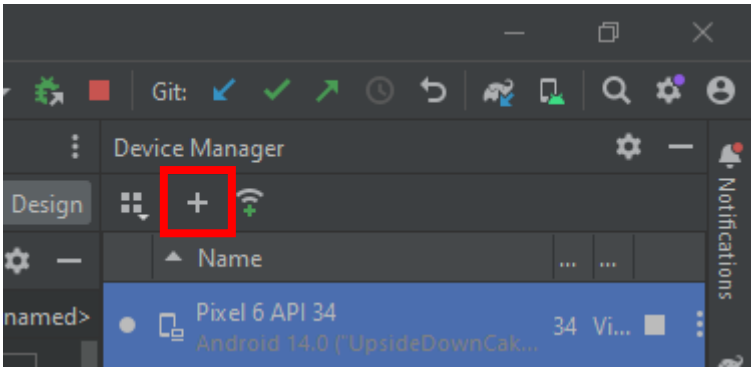
Clean Project (Limpiar Proyecto), limpia el proyecto eliminando todos los archivos compilados y los artefactos generados en la compilación anterior.Se utiliza cuando hay problemas en la compilación o cuando se quieren eliminar los archivos generados para comenzar desde cero. Puede ser útil después de realizar cambios significativos en el proyecto.

Rebuild Project (Reconstruir Proyecto), realiza la acción de "Clean Project" y luego realiza una compilación completa desde cero. Elimina todos los archivos compilados y regenera todo, incluyendo los archivos de clase, recursos y otros artefactos de construcción.Se utiliza cuando los problemas persisten incluso después de realizar la "Clean Project". Puede ser beneficioso cuando se sospecha que hay problemas profundos en la configuración del proyecto.

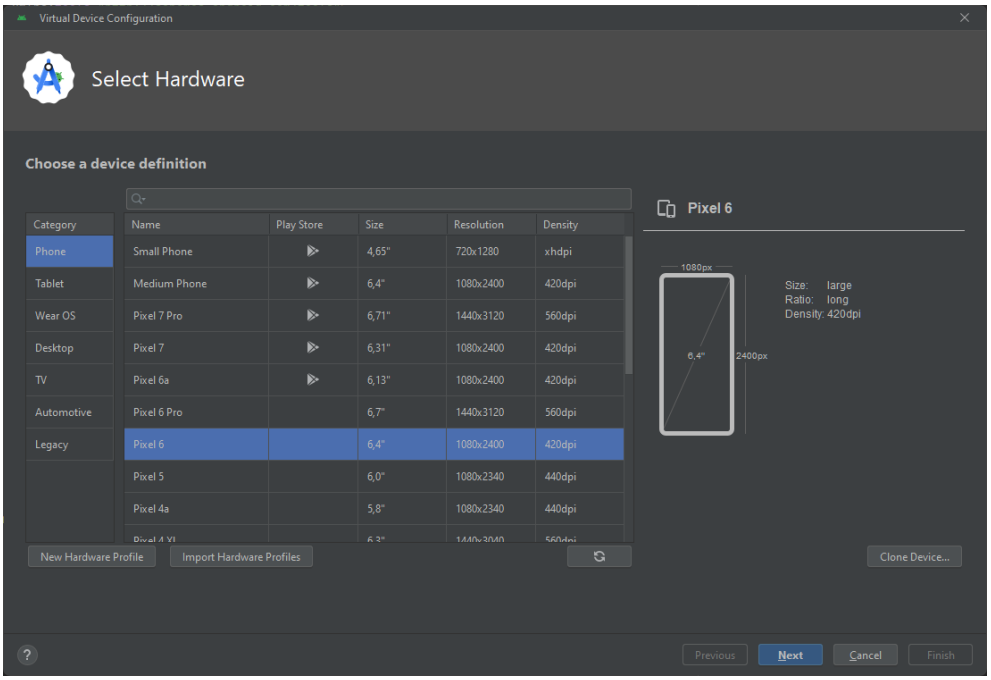
Si todos estos pasos se han ejecutado sin errores, guarda el proyecto y ahora vamos a **probar la aplicación Android** en nuestro PC sin tener que recurrir a un dispositivo físico, por lo que tenemos que definir lo que se denominan **AVD (Android Virtual Device).** Para crear un AVD seleccionaremos el menú Tools / Android / AVD Manager. Si es la primera vez que accedemos a esta herramienta veremos la pantalla siguiente:



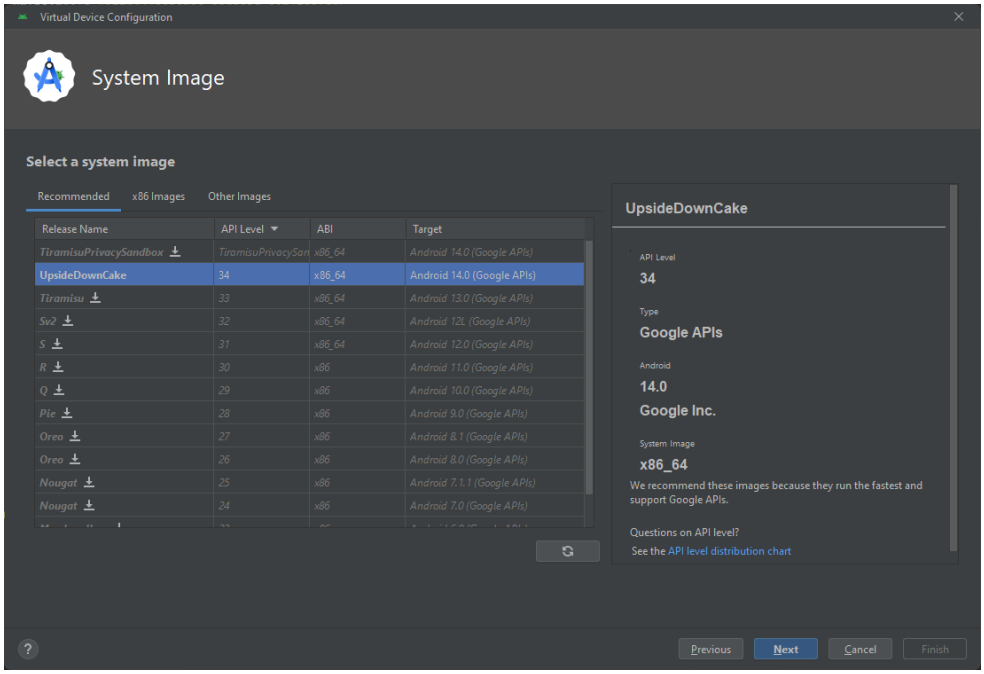
Si no te ha aparecido esta pantalla, puedes hacerlo desde aquí:



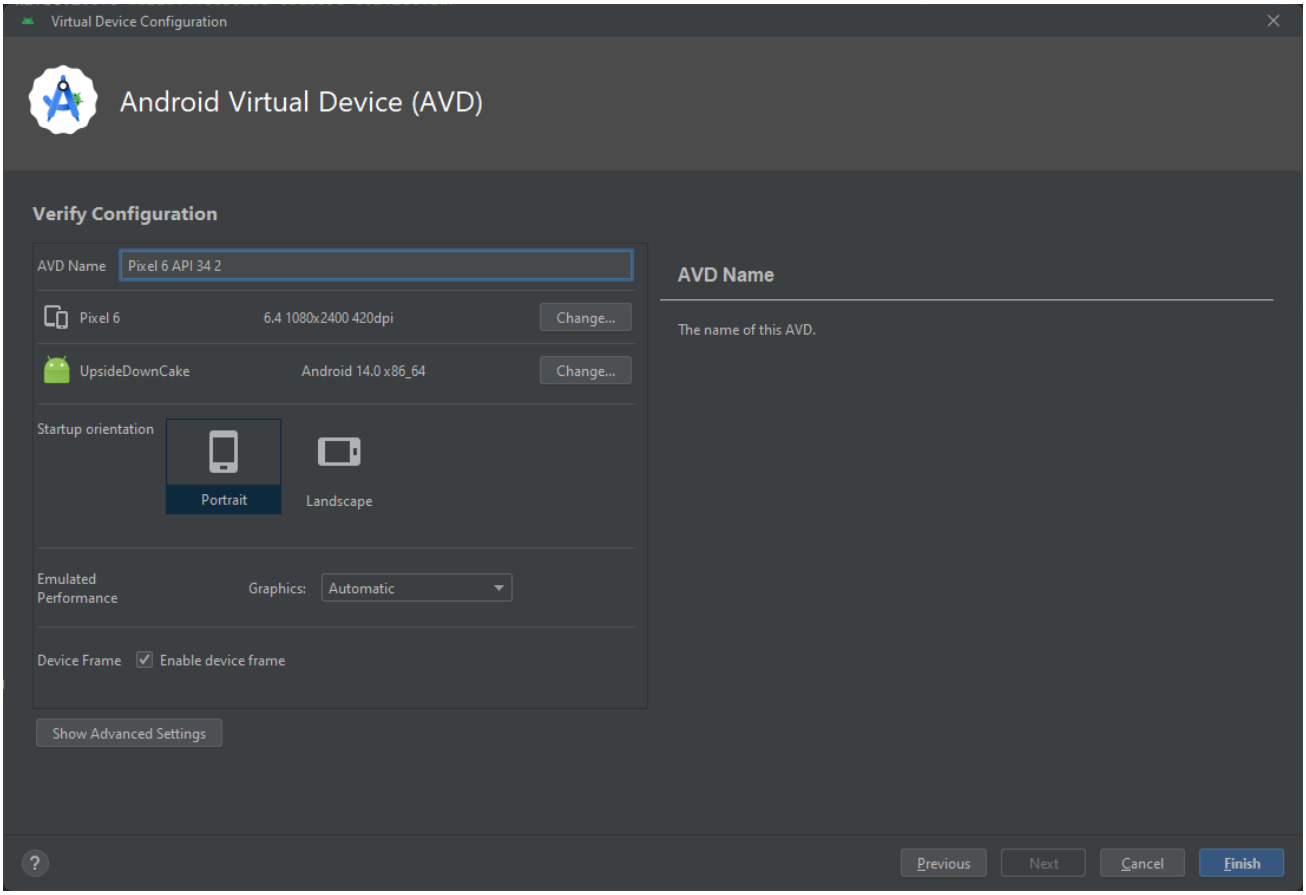
Pulsando el botón central "**Create a virtual device**" de la primera pantalla o dándole al "+" en el "Device Manager" accederemos al asistente para crear un AVD. En el primer paso tendremos que seleccionar a la izquierda qué tipo de dispositivo queremos que "simule" nuestro AVD (teléfono, tablet, reloj, ...) y el tamaño, resolución, y densidad de píxeles de su pantalla. En mi caso seleccionaré por ejemplo las características de un Pixel 6 y pasaremos al siguiente paso pulsando "Next".



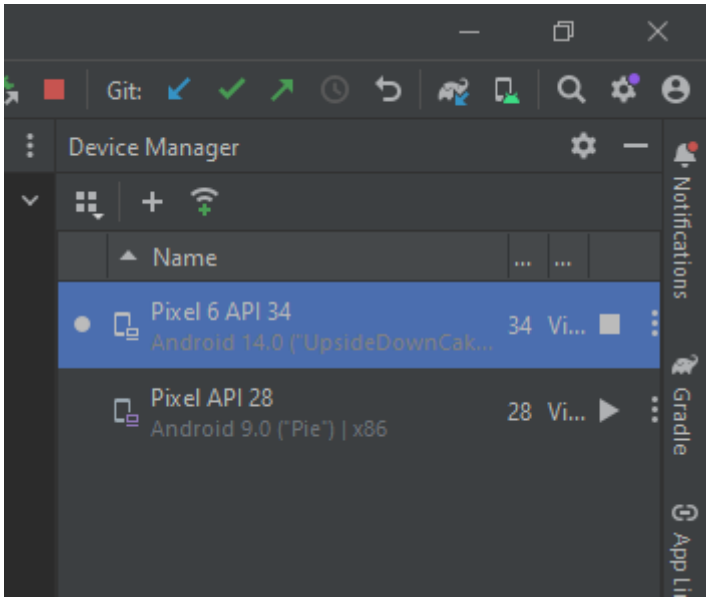
En la siguiente pantalla seleccionaremos la versión de Android que utilizará el AVD. Aparecerán directamente disponibles las que instalamos desde el SDK Manager al instalar el entorno, aunque tenemos la posibilidad de descargar e instalar nuevas versiones desde esta misma pantalla. En mi caso utilizaré la última versión de Android (UpsideDownCake) (API Level 34) para este primer AVD (podemos crear tantos como queramos para probar nuestras aplicaciones sobre distintas condiciones).



En el siguiente paso del asistente podremos configurar algunas características más del AVD, si simulará tener cámara frontal y/o trasera, teclado físico, ... Recomiendo pulsar el botón "Show Advanced Settings" para ver todas las opciones disponibles. Si quieres puedes ajustar cualquiera de estos parámetros, pero por el momento os recomiendo dejar todas las opciones por defecto.

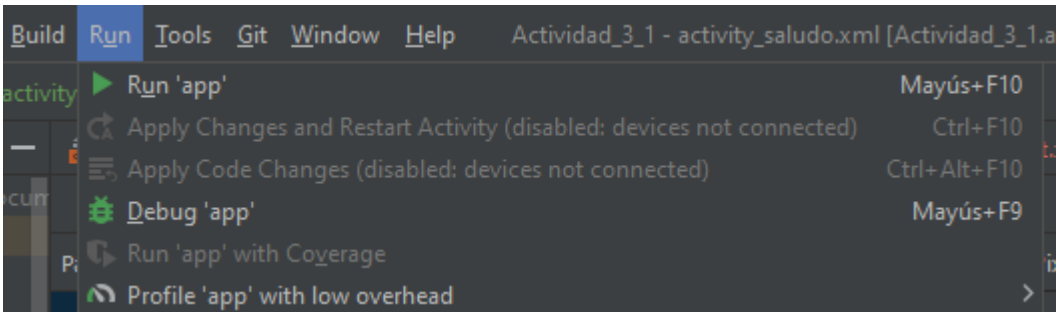


Tras pulsar el botón Finish tendremos ya configurado nuestro AVD, por lo que podremos comenzar a probar nuestras aplicaciones sobre él. En el Device Manager veremos los dispositivos que tenemos:

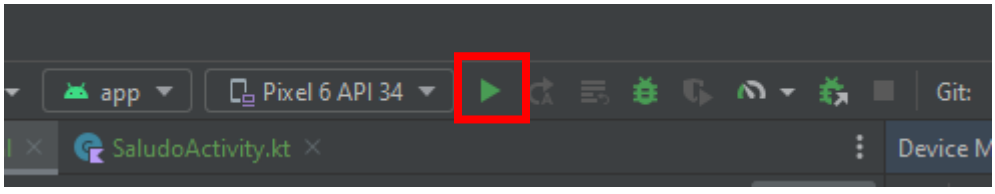


Para ejecutar la App en el dispositivo que hemos creado podemos hacerlo de varias opciones:

1. Pulsaremos simplemente el menú **Run / ejecutar Run** 'app' (o la tecla rápida Mayús+F10). Android Studio.



2. Ejecutarlo desde la ventana de navegación de la parte superior izquierda:



Donde en caso de tener más dispositivos, podemos seleccionar el que queramos en ese momento. Le damos al botón de **"Run App"**

Si todo va bien, tras una pequeña (o no tan pequeña) espera aparecerá el emulador de Android y se iniciará automáticamente nuestra aplicación (si se inicia el emulador pero no se ejecuta

automáticamente la aplicación podemos volver a ejecutarla desde Android Studio, mediante el menú Run, sin cerrar el emulador ya abierto).

Prueba a escribir un nombre y pulsar el botón "Aceptar" para comprobar si el funcionamiento es el correcto.