Juan Menduiña

Trabajo Práctico N° 1

Ejercicio 1.

Abrir Android Studio y crear un nuevo proyecto con una Actividad vacía (Empty Activity).

Ejercicio 2.

Abrir el Android Virtual Device Manager y crear, como dispositivo virtual, un Galaxy Nexus con el nombre Nexus Seminario Android.

Ejercicio 3.

Probar la aplicación creada en el Ejercicio 1 en el emulador.

Ejercicio 4.

Describir qué representa una Activity.

En *Android Studio*, una *Activity* representa una única pantalla con una interfaz de usuario (UI) en una aplicación Android. Es uno de los componentes fundamentales del ciclo de vida de una app.

Una Activity es una clase que maneja la interacción del usuario con una pantalla. Cada vez que se abre una *app* y se ve una pantalla distinta (por ejemplo, una pantalla de *login*, una pantalla de inicio, un perfil de usuario), eso, generalmente, está representado por una *Activity* diferente.

Se pensar en una Activity como el controlador (*controller*) en el patrón MVC:

- Vista (*View*): Está definida en archivos .xml (*layouts*).
- Modelo (*Model*): Suelen ser las clases de datos o lógica de negocio.
- Controlador (*Activity*): Conecta ambos, recibe eventos (*clicks*, entradas de usuario) y responde mostrando o modificando información.

Ejercicio 5.

Abrir el archivo AndroidManifest.xml. ¿Por qué MainActivity tiene un intent-filter con action MAIN y category LAUNCHER?

```
<activity
    android:name=".MainActivity"
    android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Este bloque sirve para indicarle al sistema que esta *Activity* es el "punto de entrada" principal de la *app*.

- *<action android:name="android.intent.action.MAIN" />* ⟶ Significa que esta *Activity* es la principal, es decir, la primera que se debe ejecutar cuando se inicia la *app*.
- *<category android:name="android.intent.category.LAUNCHER" />* ⟶ Indica que esta *Activity* debe aparecer como un ícono en el *launcher* del dispositivo (el menú de apps del celular).

Este *intent-filter* le dice al sistema operativo Android: "Cuando el usuario toque el ícono de la app, abrí *MainActivity*". Si no se tuviera ese *intent-filter*, la *app* no aparecería en el menú de *apps* del celular y no sabría qué pantalla mostrar al iniciar.

Ejercicio 6.

Crear 2 actividades (NuevaActivity1, NuevaActivity2) y ver qué se modificó en el archivo AndroidManifest.xml.

Cuando se crean nuevas actividades (como *NuevaActivity1* y *NuevaActivity2*), *Android Studio*, automáticamente, las registra en el archivo *AndroidManifest.xml*, porque todas las actividades deben estar declaradas allí para que *Android* pueda reconocerlas y permitir que se ejecuten.

- *android:name=".NuevaActivity1"* ⟶ Define el nombre de la clase de la nueva actividad. El punto al principio (.) indica que está en el mismo paquete que la *app* principal.
- *android:exported="false"* ⟶ Indica que esta actividad no puede ser lanzada desde fuera de la *app*, sólo desde otras partes internas de la propia *app*.

Desde Android 12 (API 31), Google requiere, explícitamente, que todas las actividades tengan declarado el atributo *android:exported*, para temas de seguridad. Este atributo es obligatorio si la *app* tiene *targetSdkVersion 31* o superior.

Ejercicio 7.

Pegar el siguiente código en res/layout/activity_main.xml.

```
<RelativeLayout
        xmlns:android=http://schemas.android.com/apk/res/android
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <Button
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Click"
                android:onClick="onBtnClick"/>
</RelativeLayout>
```

Ejercicio 8.

Añadir el siguiente código en la clase MainActivity.java, probar en el emulador y analizar el resultado:

```
fun onBtnClick(view: View?) {
      val i = Intent(this, NuevaActivity2::class.java)
      startActivity(i)
}
```

Se deben importar estos paquetes: android.view.View, android.content.Intent.

- Se inicia la *app* en el emulador.
- Se muestra su *MainActivity* con un botón que dice *"Click"*.
- Se toca el botón.
- Se abre *NuevaActivity2*.

Eso significa que está funcionando perfecto. El botón, ahora, inicia una nueva pantalla.

Ejercicio 9.

¿Qué significa this en el código del ejercicio anterior?

En *Kotlin* (y también en *Java*), la palabra clave *this* se refiere a la instancia actual de la clase. En este caso, como estás dentro de la clase *MainActivity*, *this* representa la actividad actual, es decir, el objeto de tipo *MainActivity*.

Ejercicio 10.

¿Lo que se utilizó fue un intent implícito o explícito? ¿Cuál es la diferencia entre ambos?

La clase *Intent* necesita dos cosas para funcionar:

Intent(context: Context, destination: Class<>).*

- El primer parámetro es el contexto (*Context*), que le dice desde qué componente del sistema se quiere hacer algo. En este caso, *this* es el contexto, ya que *MainActivity* hereda de *Context*.
- El segundo parámetro es la clase a la que se quiere ir (*NuevaActivity2::class.java*).

Un *intent* explícito es cuando se le dice al sistema, exactamente, a qué clase de actividad se quiere ir, por lo que, en este caso, se utilizó un *intent* explícito, ya que se dice: "Quiero ir a la actividad *NuevaActivity2* que está dentro de la *app*.". Un *intent* implícito es cuando no se especifica la clase exacta, sino que se dice: "Quiero hacer algo y que el sistema busque una *app* o componente que lo pueda hacer."

Ejercicio 11.

A través del AndroidManifest, modificar el nombre que se muestra al usuario para la actividad 2 para que, al hacer click, se muestre con el texto Actividad Nueva.

Ejercicio 12.

Ejercicio 13.

Ejercicio 14.

Ejercicio 15.

Ejercicio 16.

Ejercicio 17.

Ejercicio 18.

Ejercicio 19.

Ejercicio 20.

Ejercicio 21.

Ejercicio 22.

Ejercicio 23.

Ejercicio 24.

Ejercicio 25.