

# Bloque 3. Ejercicio Resuelto Dos Activitys

Descargar este ejercicio [pdf](#) o [html](#)

Vamos a crear un nuevo proyecto que se llamará EjercicioResuelto2Activitys. Como requisitos del mismo: queremos que se ejecute en modo apaisado y que no le afecte el cambio de orientación del dispositivo. La versión mínima soportada será la 19. La aplicación constará de dos actividades, en la primera el usuario introducirá su nombre en un EditText y cuando el usuario pulse el botón Saludo, se abrirá otra ventana donde saludaremos de forma personalizada al usuario.

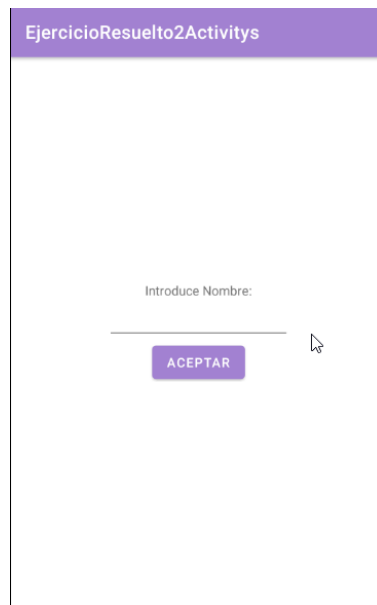
## PASO1. Crearemos el proyecto siguiendo los pasos ya realizados en los ejercicios anteriores

## PASO2. Pasemos a definir el layout de la activity principal

Para ello abre el archivo **activity\_main.xml** y copias el siguiente código, en temas posteriores se describirán los elementos que se usan:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <TextView
        android:layout_gravity="center"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Introduce Nombre:"/>
    <EditText
        android:layout_gravity="center"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:id="@+id/datos" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/button"
        android:text="Saludo"/>
</LinearLayout>
```

El resultado visual de la primera ventana de nuestra aplicación sería algo parecido al siguiente:



### PASO3. Pasemos a definir la interfaz gráfica de la segunda ventana.

Para ello creamos un nuevo fichero llamado activity2.xml, y en él añadimos únicamente una etiqueta TextView donde mostraremos el mensaje personalizado al usuario.

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

    <TextView
        android:id="@+id/textoActivity2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="132dp"
        android:layout_marginTop="320dp"
        android:text="textoActivity2"
        android:textSize="20dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

### PASO4. Crear una activity para la pantalla secundaria.

Crearemos una clase nueva derivada de AppCompatActivity. En ella implementaremos el método onCreate, al igual que se hizo para la anterior.

```

class Activity2 : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
4       setContentView(R.layout.activity2)
5       val textView=findViewById<TextView>(R.id.textoActivity2);
6       val intento= this.intent
7       textView.setText("Hola "+intento.getStringExtra("DATO")+ ", ¿como estas?")
    }
}

```

📌 la **línea4** lo que hace es cargar el layout que queremos asociar con la actividad principal y asignarlo al contexto de la aplicación, **Línea5** el método `findViewById` busca la vista en el layout del contexto y la hinch (crear el objeto del tipo correspondiente asociado a esa vista, en este caso `TextView`). **Línea6** recuperamos la instancia de intent que llega a la actividad. **Línea7** extraemos la información del intent mediante el get correspondiente, en este caso `getStringExtra` pasando como parámetro la clave que lo identifica, y se añade al `textView`.

👉 es importante declarar en el Manifest todas las actividades que se añadan a nuestra aplicación, en nuestro caso:

```

<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:screenOrientation="landscape"
    android:configChanges="orientation|screenSize"
    android:theme="@style/Theme.EjercicioResuelto2Activitys">
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
16    <activity android:name=".Activity2"/>
</application>

```


## PASO5. Vamos a implementar la lógica de funcionamiento de nuestra aplicación.

El código es el siguiente, solo se explican los elementos nuevos del código.

```

class MainActivity: AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
5       findViewById<Button>(R.id.button).setOnClickListener {
6           val intento=Intent(applicationContext, Activity2::class.java)
7           val datos=findViewById<EditText>(R.id.datos).text.toString()
8           intento.putExtra("DATO", datos)
9           startActivity(intento)
        }
    }
}

```

 En la **línea5** se infla el botón y con su método escuchador de Click añadimos el código que queremos que ocurra al pulsar el botón. En la **Línea6** se crea un intent redirigido a la Activity2. **Línea7** se extrae la información, introducida por el usuario, del EditText. **Línea8** se añade el extra al intent usando la clave identificativa "DATO", para posteriormente extraer esa información. **Línea9** se inicia la actividad con el intent creado.