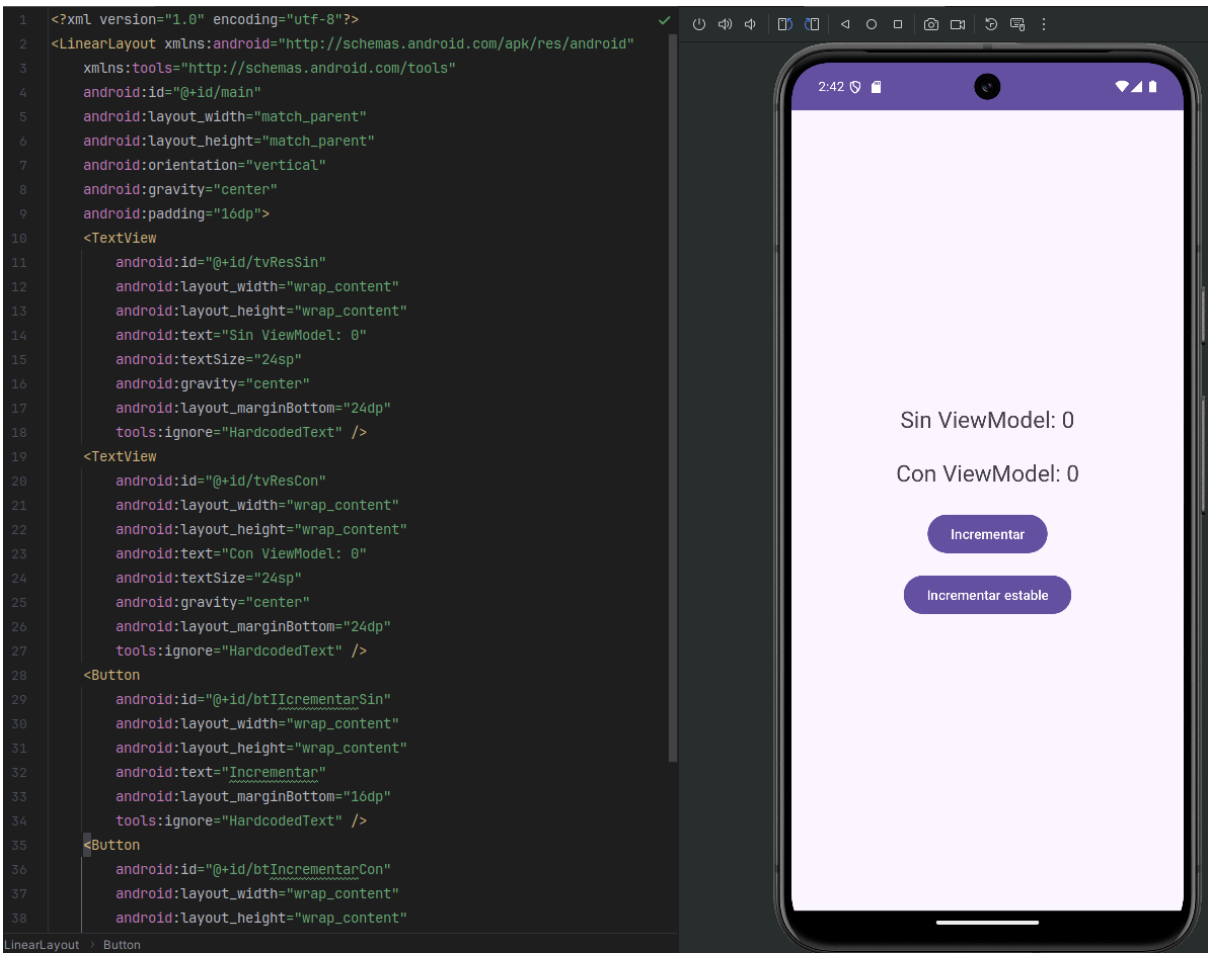
TP Relación entre el ciclo de vida de una activity y un ViewModel

1. Crear un proyecto y en la MainActivity crear dos textView llamado tvResSin y tvResCon y dos botones una para incrementar sin usar un ViewModel y otro botón para incrementar usando un ViewModel.



```xml
<?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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    android:padding="16dp">
    <TextView
        android:id="@+id/tvResSin"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Sin ViewModel: 0"
        android:textSize="24sp"
        android:gravity="center"
        android:layout_marginBottom="24dp"
        tools:ignore="HardcodedText" />
    <TextView
        android:id="@+id/tvResCon"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Con ViewModel: 0"
        android:textSize="24sp"
        android:gravity="center"
        android:layout_marginBottom="24dp"
        tools:ignore="HardcodedText" />
    <Button
        android:id="@+id/btIIcrementarSin"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Incrementar"
        android:layout_marginBottom="16dp"
        tools:ignore="HardcodedText" />
    <Button
        android:id="@+id/btIncrementarCon"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

LinearLayout › Button

2. Creemos una clase Incrementar con una función muy sencilla:

```java
package com.dev.appincrementar;

public class Incrementar {  no usages
    public static final int incrementar(int num){  no usages
        return ++num;
    }
}
```

3. Creamos un ViewModel que es el que tendrán en cuenta el listener del botón btIncrementarCon y el tvResCon

```java
package com.dev.appincrementar;

import androidx.lifecycle.ViewModel;

public class IncrementarViewModel extends ViewModel {  2 usages
    private Integer resultado;  4 usages
    public int getResultado() {  3 usages
        if (resultado == null) {
            resultado = 0;
        }
        return resultado;
    }
    public void setResultado(int resultado) {  1 usage

        this.resultado = resultado;
    }
}
```

4. Probamos agregar la siguiente funcionalidad en nuestra clase MainActivity:

```java
public class MainActivity extends AppCompatActivity {
    private ActivityMainBinding binding;  8 usages
    private IncrementarViewModel incrementarViewModel;  5 usages
    private int res = 0;  4 usages

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        binding = ActivityMainBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());
        // Inicializamos nuestro ViewModel
        incrementarViewModel = new ViewModelProvider( owner: this).get(IncrementarViewModel.class);
        Log.d( tag: "TAG1",  msg: "onCreate()");
        // Mostrar el valor de los  editText ya que este método se ejecutará varias veces
        binding.tvResCon.setText("Con ViewModel: "+incrementarViewModel.getResultado());
        binding.tvResSin.setText("Sin ViewModel: "+res );
        tarea();
    }
    public void tarea() {  1 usage
        binding.btIIcrementarSin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Actualizar solo la variable local (no se mantiene tras rotación)
                res = Incrementar.incrementar(res);
                binding.tvResSin.setText("Sin ViewModel: "+res );
            }
        });
        binding.btIncrementarCon.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Incrementar el valor almacenado en el ViewModel (se mantiene tras rotación)
                incrementarViewModel.setResultado(Incrementar.incrementar(incrementarViewModel.getResultado()));
                binding.tvResCon.setText("Con ViewModel: "+incrementarViewModel.getResultado());
            }
        });
    }
```

5. Probamos el botón incrementar varias veces y giramos la pantalla. Observar cómo se pierden los datos ante el cambio de pantalla. Esto se debe a los cambios en el ciclo de vida de una Activity.

6. Sobreescribir los siguientes métodos, agregando la etiqueta en un Lod.d “TAG1”:

```
@Override
protected void onStart(){
  super.onStart();
        Log.d( tag: "TAG1", msg: "onStart()");
 }
    @Override
protected void onResume(){
  super.onResume();
  Log.d( tag: "TAG1", msg: "onResume()");
}
@Override
protected void onPause(){
    super.onPause();
    Log.d( tag: "TAG1", msg: "onPause()");
}
@Override
protected void onStop(){
  super.onStop();
  Log.d( tag: "TAG1", msg: "onStop()");
}
@Override
protected void onDestroy(){
  super.onDestroy();
  Log.d( tag: "TAG1", msg: "onDestroy()");
  }
}
```

Filtrar la etiqueta TAG1 en LogCat y observar como cambia de estado la activity solo con rotar la pantalla.