Instituto Tecnológico Superior De Lerdo

Programación móvil

Practica: 3.2 y 3.3

Profesor: Jesús Salas Marín

Alumno: Luis Andres Rodriguez Campos

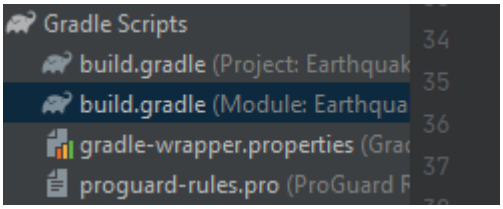
Carrera: Ing. Informática

Sección: A

Grado: 8

Numero de control: 17231573

Nos iremos a build.gradle para gragar el databinding



Aquí usaremos el databinding

```
buildFeatures {
    dataBinding true
}
```

Añadimos estas lineas

```
implementation "com.squareup.retrofit2:converter-moshi:2.5.0"
implementation "androidx.lifecycle:lifecycle-viewmodel:2.2.0"
implementation "androidx.room:room-runtime:2.2.6"
annotationProcessor "androidx.room:room-compiler:2.2.6"
testImplementation 'junit:junit:4.+'
androidTestImplementation 'androidx.test.ext:junit:1.1.2'
androidTestImplementation 'androidx.test.espresso:espresso-core:3.3.0'
```

En nuestro main activity añadiremos el siguiente código y añadimos las librerías necesiaras

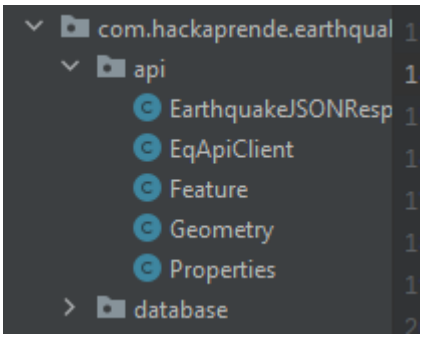
```
package com.hackaprende.earthquakemonitor.main;

import android.os.Bundle;
import android.view.View;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.ViewModelProvider;
import androidx.recyclerview.widget.LinearLayoutManager;

import com.hackaprende.earthquakemonitor.databinding.ActivityMainBinding;

public class MainActivity extends AppCompatActivity {
```

```java
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    ActivityMainBinding binding = ActivityMainBinding.inflate(getLayoutInflater());
    setContentView(binding.getRoot());

    MainViewModel viewModel = new ViewModelProvider( owner: this,
            new MainViewModelFactory(getApplication())).get(MainViewModel.class);

    binding.eqRecycler.setLayoutManager(new LinearLayoutManager( context: this));
```

```java
MainViewModel viewModel = new ViewModelProvider( owner: this,
        new MainViewModelFactory(getApplication())).get(MainViewModel.class);

binding.eqRecycler.setLayoutManager(new LinearLayoutManager( context: this));

EqAdapter adapter = new EqAdapter();
adapter.setOnItemClickListener(earthquake ->
        Toast.makeText( context: MainActivity.this, earthquake.getPlace(),
                Toast.LENGTH_SHORT).show());

binding.eqRecycler.setAdapter(adapter);

viewModel.getEqList().observe( owner: this, eqList -> {
    adapter.submitList(eqList);

    if (eqList.isEmpty()) {
        binding.emptyView.setVisibility(View.VISIBLE);
    } else {
        binding.emptyView.setVisibility(View.GONE);
    }
});

viewModel.downloadEarthquakes();
```

```java
package com.hackaprende.earthquakemonitor.main;

import ...

public class MainViewModel extends AndroidViewModel {

    private final MainRepository repository;

    public MainViewModel(@NonNull Application application) {
        super(application);
        EqDatabase database = EqDatabase.getDatabase(application);
        this.repository = new MainRepository(database);
    }

    public LiveData<List<Earthquake>> getEqList() { return repository.getEqList(); }

    public void downloadEarthquakes() { repository.downloadAndSaveEarthquakes(); }
}
```

Ahora obtendremos la API



Y en EarthQuakeJSON obtendremos un archivo JSON de las APIS necesarias.

```java
package com.hackaprende.earthquakemonitor.api;

import java.util.List;

public class EarthquakeJSONResponse {
    private List<Feature> features;

    public List<Feature> getFeatures() { return features; }
}
```

Un poco mas de código de api

```java
ckage com.hackaprende.earthquakemonitor.api;

port ...

blic class EqApiClient {
  public interface EqService {
      @GET("all_hour.geojson")
      Call<EarthquakeJSONResponse> getEarthquakes();
  }

  private final Retrofit retrofit = new Retrofit.Builder()
          .baseUrl("https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/")
          .addConverterFactory(MoshiConverterFactory.create())
          .build();

  private EqService service;

  private static final EqApiClient ourInstance = new EqApiClient();

  public static EqApiClient getInstance() { return ourInstance; }

  private EqApiClient() {
  }

  public EqService getService() {
      if (service == null) {
          service = retrofit.create(EqService.class);
      }
      return service;
  }
```

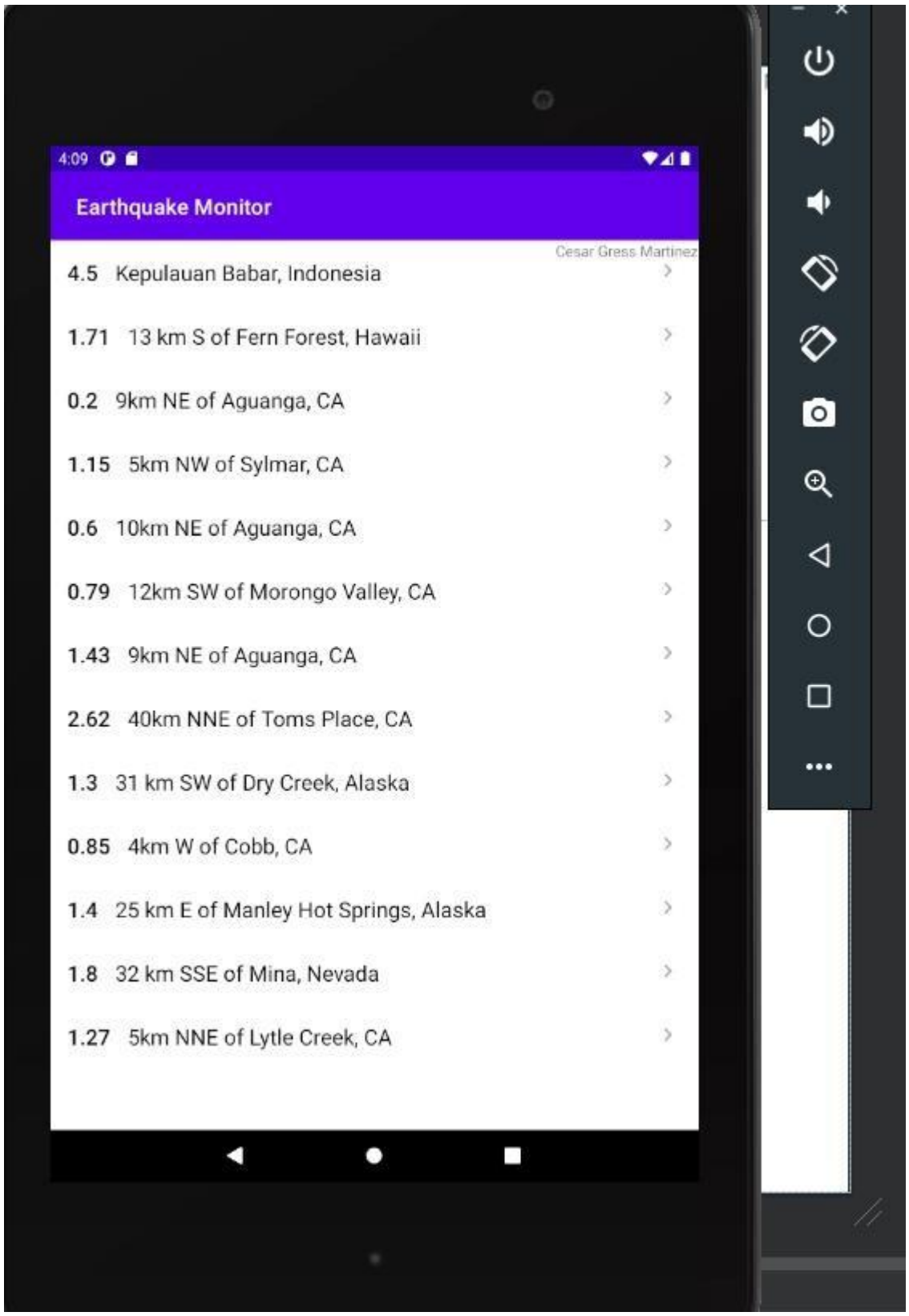
En main activity ponemos el databinding

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

    <data>

    </data>
```

Y quedaría asi

```xml
<FrameLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".main.MainActivity">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/eq_recycler"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        tools:itemCount="5"
        tools:listitem="@layout/eq_list_item" />

    <TextView
        android:id="@+id/empty_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:gravity="center"
        android:textColor="@color/black"
        android:textSize="24sp"
        android:visibility="gone"
        android:text="No earthquakes right now"/>

    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Cesar Gress Martinez"
        android:textAlignment="textEnd"
        android:textDirection="inherit" />

</FrameLayout>
</layout>
```

Prueba


4:09
Earthquake Monitor
Cesar Gress Martinez
4.5 Kepulauan Babar, Indonesia
1.71 13 km S of Fern Forest, Hawaii
0.2 9km NE of Aguanga, CA
1.15 5km NW of Sylmar, CA
0.6 10km NE of Aguanga, CA
0.79 12km SW of Morongo Valley, CA
1.43 9km NE of Aguanga, CA
2.62 40km NNE of Toms Place, CA
1.3 31 km SW of Dry Creek, Alaska
0.85 4km W of Cobb, CA
1.4 25 km E of Manley Hot Springs, Alaska
1.8 32 km SSE of Mina, Nevada
1.27 5km NNE of Lytle Creek, CA

Conclusión

La inclusión de APIS te ayuda bastante a la hora de programar y necesitas datos que son difíciles de conseguir o simplemente tediosos, las APIS nos beneficias mucho cuando las utilizamos igual que los DATABINDING y las pantallas recicladas todo esto mencionado anteriormente nos ayuda a una mejor gestión de uso de memoria en los celulares ya que no crea varias pantallas solo recicle la misma y usa diferentes datos.