Instituto Superior Tecnológico del Azuay

Tema2

Consumir API con Retrofit

Carrera

Teología en Desarrollo de Software

Asignatura

Desarrollo de Aplicaciones Móviles

Docente

Mgtr. Patricio Pacheco

Estudiante

Marlon Vélez

Ciclo

Cuarto

Grupo

“B”

Fecha de Entrega

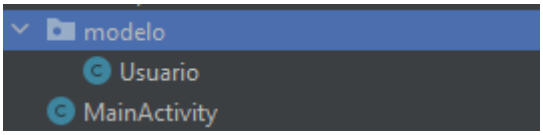
18-03-2022

Consumiendo Servicios de un API a través de la librería de Retrofit

1. Creamos un nuevo proyecto dentro del IDE IntelliJ y entramos al build.gradle para agregar la siguientes dependencias.

   implementation 'com.squareup.retrofit2:retrofit:2.5.0'
   implementation 'com.squareup.retrofit2:converter-gson:2.5.0'

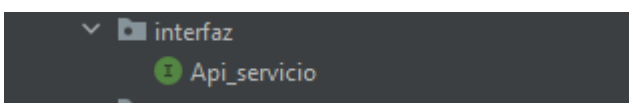
2. Una vez hicimos agregadas las dependencias, creamos un nuevo paquete llamado “modelo” y dentro de ese creamos una clase modelo llamada “Usuarios”.



3. Dentro de la clase Usuario creamos las variables necesarias, estas deben ser nombradas exactamente como el JSON del API que estemos consumiendo.

```
public class Usuario {

    private int id;
    private String name;
    private String username;

    //GETTERS

    public int getId() { return id; }

    public String getName() { return name; }

    public String getUsername() { return username; }

    //SETTERS

    public void setId(int id) { this.id = id; }

    public void setName(String name) { this.name = name; }

    public void setUsername(String username) { this.username = username; }
}
```

4. Una vez estemos con el modelo, creamos otro paquete llamado interfaz y luego una interfaz llamada Api_servicio.



5. Dentro de esta interfaz vamos a crear los métodos para poder consumir los servicios de la API. En este caso, creamos un método get para conseguir el listado de usuarios.

```java
package com.example.mi_retrofit_velez.interfaz;

import com.example.mi_retrofit_velez.modelo.Usuario;
import retrofit2.Call;
import retrofit2.http.GET;

import java.util.List;

public interface Api_servicio {

    @GET("users")
    Call<List<Usuario>> getAllUsuers();
}
```

6. Después en la vista (layout) creamos un NestedScrollView y dentro de este insertamos un TextView y a este le asignamos id.

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

    <androidx.core.widget.NestedScrollView android:layout_width="match_parent" android:layout_height="match_parent">

        <TextView
                android:id="@+id/usuarioItem"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintLeft_toLeftOf="parent"
                app:layout_constraintRight_toRightOf="parent"
                app:layout_constraintTop_toTopOf="parent"
        />
    </androidx.core.widget.NestedScrollView>

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

7. Una vez hecho esto en la clase MainActivity crearemos el método para poder finalmente consumir el servicio del API.

```
30
31      private void getUser(){
32
33          Retrofit retrofit = new Retrofit.Builder()
34                  .baseUrl("https://jsonplaceholder.typicode.com/")
35                  .addConverterFactory(GsonConverterFactory.create())
36                  .build();
37
38          Api_servicio api= retrofit.create(Api_servicio.class);
39
40          Call<List<Usuario>> user= api.getAllUsuers();
41
42          user.enqueue(new Callback<List<Usuario>>() {
43              @Override
44              public void onResponse(Call<List<Usuario>> call, Response<List<Usuario>> response) {
45                  if (!response.isSuccessful()) {
46                      usuarioItem.setText("Codigo: "+response.code());
47                      return;
48                  }
49
50                  List<Usuario> listaUsuario= response.body();
51
52                  for(Usuario user: listaUsuario){
53
54                      String content= "";
55                      content+= "nombre:"+ user.getName()+ "\n";
56                      content+= "usuario:"+ user.getUsername()+ "\n\n";
57                      usuarioItem.append(content);
58                  }
59              }
60
61              @Override
62              public void onFailure(Call<List<Usuario>> call, Throwable t) { usuarioItem.setText(t.getMessage()); }
65          });
66      }
67
68  }
```

Para enviar una solicitud de red a la API, tenemos que utilizar la clase Retrofit Builder y especificar la URL base del servicio, en esta, ponemos la URL del API solo tomando en cuenta parte antes de hacer referencia al método que queremos consumir (esa parte va en la interfaz Api_servicio).

Los servicios pueden ser creados usando Api_servicio para enlazar con end points específicos. Retrofit lee y analiza los datos de la API a nivel de base y luego devuelve los resultados al hilo de la interfaz de usuario a través del método onResponseo.

Luego simplemente mandamos estos datos a nuestra vista haciendo referencia mediante el id a nuestro TextView.

Al final este será el resultado será un listado de usuarios que me arroja el API.