Tutorial. API

Objectiu

Fer una connexió a una api i recollir les dades.

API

https://sunrise-sunset.org/api

Demo

Consulta la documentació oficial aquí

Pas a pas. HTTPCONNECTION

1. Crearem el Thread, tal i com hem fet altres vegades

```
public class ApiThread extends AsyncTask<Void, Void, Void> {
```

2. Crearem el constructor i els mètodes doInBackground i onPostExecute
3. En el constructor li passarem les dades que vulguem consultar, per exemple la latitud i la longitud.
4. En el mètode doInBackground:
   Crearem l'objecte URL que apuntarà a la URL de l'API

```
URL url = new URL("https://api.sunrise-sunset.org/json?lat=36.7201600&lng=-4.4203400");
```

*Assegura't que aquesta URL funciona correctament obrint-la amb un navegador

Realitzarem la connexió mitjançant la classe HttpURLConnection

```
HttpURLConnection httpURLConnection = (HttpURLConnection) url.openConnection();
```

Llegirem els resultats i els passarem a String

```
// Read API results
InputStream inputStream = httpURLConnection.getInputStream();
BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(inputStream));

String data = bufferedReader.readLine();
return data;
```

5. Al mètode onPostExecute analizarem i extraurem les dades del string data (json de l'api)
   El primer que fem és convertir tot l'String data en un json de nou per poder-lo analitzar.

```
jObject = new JSONObject(data);
```

Ara hem d'agafar les dades del "fill" results

```
jObject = jObject.getJSONObject("results");
```

Finalment agafem el "fill" sunrise de results

```
String sunrise = jObject.getString("sunrise");
Log.i("logtest", "------>" + sunrise);
```

Pas a pas. Retrofit

Documentació oficial: https://square.github.io/retrofit/

1. Afegim les llibreries i fem click a Sync now

```
//Retrofit
implementation 'com.squareup.retrofit2:retrofit:2.9.0'

//Gson
implementation 'com.squareup.retrofit2:converter-gson:2.3.0'
```

2. Creem el model. Crearem el model de dades que volem agafar de l'Api. Per a fer-ho crearem un nou package que es dira model i dins hi crearem una nova classe.
   Els noms dels atributs de la classe han de ser els mateixos que els de l'api per a que Retrofit ho pugui enllaçar.

```
public class ModelResults {
   public String sunrise;
   public String sunset;
}

public class ModelApi {
   public String status;
   public ModelResults results;
}
```

   Fixeu-vos que creem un model de dades per a cada objecte del json. Una recomanació és que creeu els models de baix a dalt.

   Crearem els getters dels atributs.

3. Crearem la interfície. Ens servirà per declarar les crides a l'api.

```
public interface ApiCall {
   @GET("json?lat=36.7201600&lng=-4.4203400")
   Call<ModelApi> getData();
}
```

Ara per ara no li passarem els paràmetres, ho fem de manera estàtica

4. En el MainActivity farem la declaració de la crida al retrofit.

```
Retrofit retrofit = new Retrofit.Builder()
        .baseUrl("https://api.sunrise-sunset.org/")
        .addConverterFactory(GsonConverterFactory.create())
        .build();
```

Fixeu-vos que li passem la base de la URL i a la interfície només els paràmetres. També li passem el conversor del JSON que hem importat a l'inici

```
ApiCall apiCall = retrofit.create(ApiCall.class);
Call<ModelApi> call = apiCall.getData();
```

Aquestes dues línies el que fan és crear un objecte de la classe ApiCall i finalment executar la crida a l'Api mitjançant el mètode getData.

El retorn d'informació el treballarem a través del Callback

```
call.enqueue(new Callback<ModelApi>(){
   @Override
   public void onResponse(Call<ModelApi> call, Response<ModelApi> response) {
       if(response.code()!=200){
           Log.i("testApi", "checkConnection");
           return;
       }

      Log.i("testApi",     response.body().getStatus()     +     "     -     "     +
response.body().getResults().getSunrise());
   }

   @Override
   public void onFailure(Call<ModelApi> call, Throwable t) {

   }
});
```

5. Passar variables a l'interfície. Per passar variables a la interfície hem de modificar la crida a l'api.

```
@GET("json?")
Call<ModelApi> getData(@Query("lat") String lat, @Query("lng") String lng);
```

Exemples:

```
@GET("json?lat=3.4&long=3.5")
Call<ModelApi> getData(@Query("lat") String lat, @Query("lng") String lng);

@GET("json/{lat}/{lng}")
Call<ModelApi> getData(@Path("lat") String lat, @Path("lng") String lng);
```

Fixeu-vos que li passem dos paràmetres lat i lng que afegirà a la url per convertir-la en json?lat=X&lng=Y

Si vulguessim generar una url tipus: pokeapi.com/v2/pokemon/nom els paràmetres els hauríem de passar per Path, perquè formen part del path de la URL. A la documentació de l'Api hi trobareu tota la informació

6. Finalment cal modificar la crida a l'Api i passar-li les dades.

```
Call<ModelApi> call = apiCall.getData(lat, lng);
```

Exercici

Utilitzarem l'API de Flickr per mostrar fotos d'una localització concreta. El primer que hem de fer és entendre l'estructura de l'API, posteriorment creareu tot el model i la crida a retrofit.
Volem extreure 5 imatges de cada punt i mostrar-los en un slider en una nova finestra.

1. Entendre l'API de Flickr
   Informació de l'API. Funcionalitat Search
   https://www.flickr.com/services/api/flickr.photos.search.html

   Tester de Search
   https://www.flickr.com/services/api/explore/flickr.photos.search

   Aconseguir la URL de les fotos
   https://www.flickr.com/services/api/misc.urls.html

   Claus:
   Clave: 79d466885188b99d6762980d64029892
   Secreto: 2c609b0101cb50dc

   URL exemple:
   https://www.flickr.com/services/rest/?method=flickr.photos.search&api_key=79d466885188b99d6762980d64029892&lat=41.3879&lon=2.16992&format=json&nojsoncallback=1

2. Creareu el model fixeu-vos en les aniuacions del json
   Haureu de crear el model Photo on hi haurà la informació de cada fotografia (id, owner, secret, server…), el model Photos (page, pages, perpage, total i una array de tipus Photo) i finalment el model pare que tindrà una instància de Photos i stat.

3. Crea la crida a l'API amb retrofit
4. Fixa't en com es genera la URL d'una imatge
   https://live.staticflickr.com/{server-id}/{id}_{secret}.jpg

   server-id = 65535
   id = 51879563573

secret = 3c58e9bcde

https://live.staticflickr.com/65535/51879563573_3c58e9bcde.jpg

5. Crea una array de tipus String amb les 5 primeres imatges del lloc que seleccionis
6. Fes un log per poder visualitzar-les