LAB 2 ANDROID – ASYNCHRONOUS TASKS AND SENSORS

Student : Matthieu GEDEON

Teacher : Jean-François LALANDE

Table des matières

Github link to the project : https://github.com/Fraisedesneiges/android-tp2

My take on Lab Work 2

I took on this lab session following all the advices on the instructions and making my best to search on Android Developers and StackOverflow every information that could be useful to me. I took a liking to understand what I was doing, and this Lab was very instructing.

The goal of this Lab was to take advantage of the AsyncTask to execute task in parallel and update the UI with the fetched data from the Flickr API using an HTTPUrlConnection object.

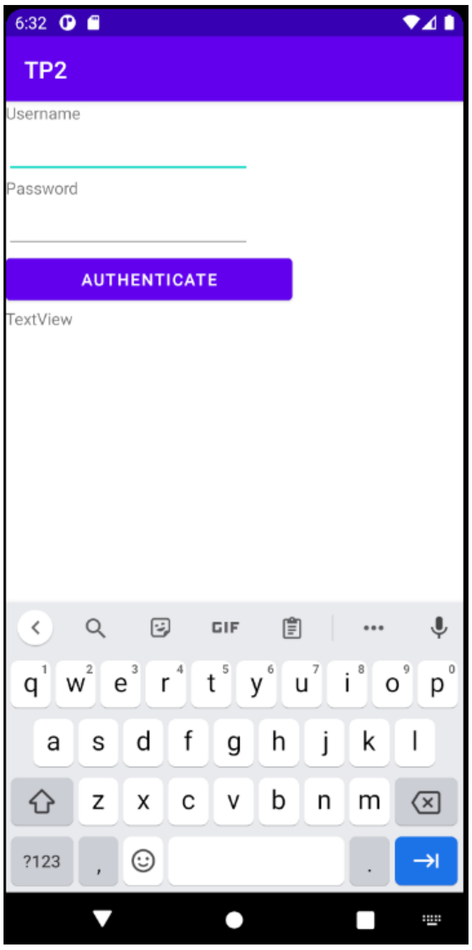
Authentication with an asynchronous task

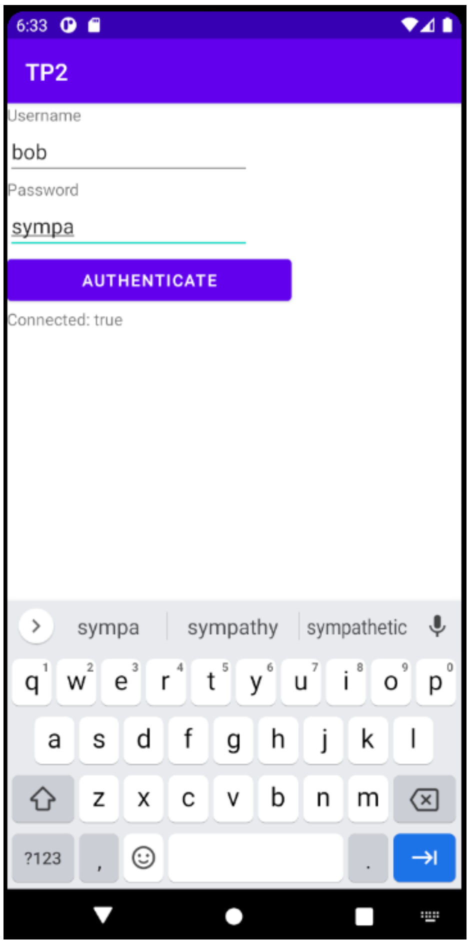
This work is the project **part-one** in the **android-tp2** folder.

The goal of this part is to do a simple authentication by submitting credentials to the "HTTPBin" (https://httpbin.org/basic-auth/bob/sympa) online service using a a Thread object.

Exercise 3: The readStream method must be user implemented. By searching on StackOverflow, you can find several solutions that can be used as the body for this method. I choose one with a StringBuilder and a BufferedReader for better efficiency.

Exercise 4: You can see the errors on the Logcat, the Build or the Run tabs on the bottom of the Android Studio IDE.





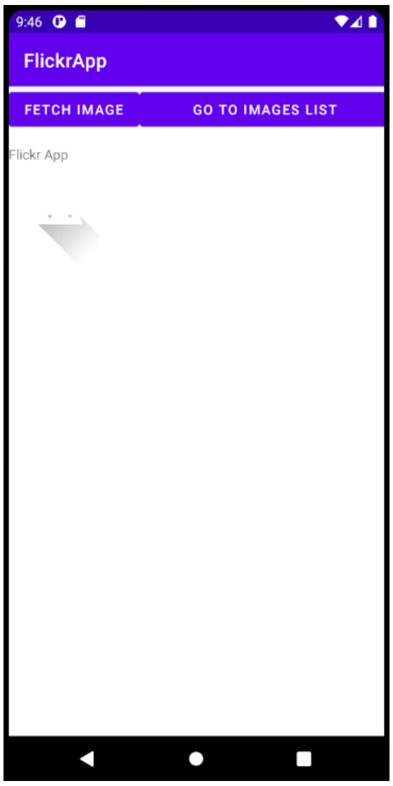
Asynchronous tasks for a real remote service: Flickr

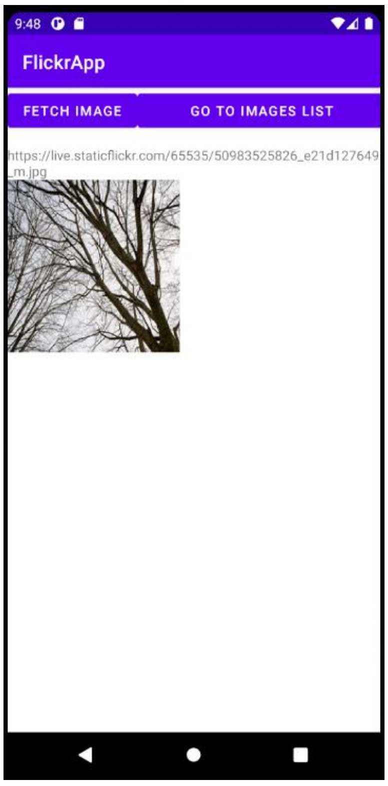
The goal of this part is to get a JSON object from Flickr (https://www.flickr.com/services/feeds/docs/photos_public/) containing URLs that will permit us to fetch bitmap images afterwards and inject them into a ListView. We will use for this AsyncTask and the HTTP library Volley for the API calls as well as build an Adapter to update our view with our fetched data.

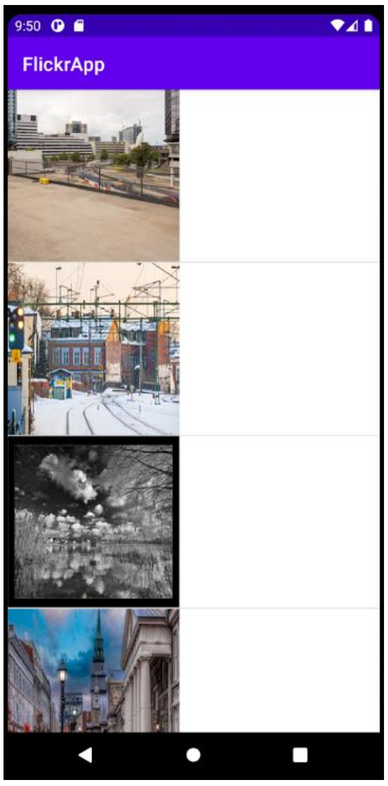
Exercise 12: Data transferred via HTTP is String based, so the received data from the request to Flickr is not really a JSONObject but it can be interpreted back as such afterwards, with of course the jsonFlickrFeed(-) wrapper removed from the string that we will a pass to the JSONObject constructor.

Exercise 13: We can thanks to the AsyncTask object execute in parallel a task with several progression methods that can interact with the UI Thread (and thus not call the runOnUiThread method). In our case we use the doInBackground and onPostExecute methods the most, with first one receiving the first generic type (here a String and to be more precise, a String array) of our AsyncTask and returning the third type (in our case a JSONObject) that will be used in the onPostExecute method.

Exercise 27: Run the block dependencies in the build.gradle to get access to the missing imports.




9:48
FlickrApp
FETCH IMAGE
GO TO IMAGES LIST
https://live.staticflickr.com/65535/50983525826_e21d127649_m.jpg



Conclusion

I failed to make the bonus and the Geolocalization part as I had many works to do on parallel. But I am happy to have succeeded the rest and understood the main concepts behind this Lab work.