Report

Spring 2018
Alexandre Allani
April 1, 2018

Report of the weather Application

What have I implemented

I have implemented a fully functional weather application as required. Using the wunderground API, all the weather information are fetched and displayed on the screen.
It shows the name of the city, the min/max temperature of the day, and of the next 3 days. It also displays the humidity level, and there is a section for the hourly forecast.

How to use the application

Before fetching the data, the screen should display "NA", and interrogation marks where images are supposed to be. When data are fetched, a toast text gives the information that the application has been refreshed. The data are supposed to be fetched automatically, but if it does not fetch data, press the two arrows in the right-hand corner. This should work

Possible issues

If the application is runned on a real device, there are no issues. However, if it is runned on an Android emulator, some changes must be made. The line 61 in the WeatherController.java file: " String LOCATION_PROVIDER = LocationManager.NETWORK_PROVIDER;", must be replaced by "String LOCATION_PROVIDER = LocationManager.GPS_PROVIDER". Indeed, I used location by the network provider, which is coarser than the GPS, but better in term of refreshment for a physical device. For an emulated device, network location does not work. Hence only the GPS location can be used.
Sometimes due to the wunderground's API, there are no high value in Daejeon, but usually, in other places of the world there are.
If no data are fetched, or if the application crashes, this mean that my Key for the wunderground API has been blocked. It already happened once. This is due to the fact that each refresh consumes 4 requests. And there's a limitation of 10 requests per minute. If this happens, the KEY_API value can be changed on line 40 in the WeatherController.java file.

How is it working ?

In order to use the wunderground API, and make a request, I used the library HTTP Async request, made by John Smith (compile 'com.loopj.android:android-async-http:1.4.9').
I've also used the GPS API and learned how to ask the permission on an Android device. I created all the view variable and In onCreate() I linked them to UI. After that basically, the getWeather() function is the function refreshing the data. The variable mLocationManager is keeping track of GPS coordinates. And with the requestLocationUpdates(), we know when the location has changed. I've chosen to update the location if the time between two requests was greater than 5 seconds, and the distance is greater than 1 km. However, if the user refreshes the app manually, the application oversteps those condition.
The CityValue.java, WeatherCondition.java, WeatherForecast.java and the WeatherHourly.java are the classes where the requests to Wunderground are made. It only parses the JSON Object obtained. They are called by the different apiCall function. The changes of the view are made in the update1(), update2() and update3() function. Indeed, since the requests are supposed to be made asynchronously, only part of the code updates when the information are received explaining why there are 3 update function.
Finally, when the application is going on pause State, the mLocationManager is cleared of all the updates. No more updates are done. And updates start again when onResume() is called.