13. August 2015, Horsens

Documentation: Locationbased Alarms

- by Javeed and Zara -

The Preamble

The *Locationbased Alarms* application is build for travellers, who want to take a nap while they are travelling. A person who wants to use the application just have to start the main screen, add an location in the integrated google maps activity and press the start button which will be enabled, after the user chooses a location.

Now it is time to relax, the traveller can get a comfortable position and do not have to care about missing his destination. The Service will continue running in the background and will wake up the user in the right time.

Note: The application is currently build for the context of distances which are far a way, like for travellers who use the train and not going by feed. Please read the “Future View” to have an idea what could be changed soon.

The User Experience (UX) And The Code

Step 1 - `MainActivity`

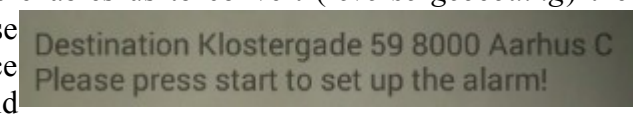
When the user starts the application first he/she will come to the main screen and will find a short description what to do. He also will find two buttons, one will be disabled for now the other one will delegate the user directly to the next screen.

Step 2 - `MapsActivity`

The integrated Google Maps API (v2) enables the user to choose a location on the screen and add a marker. The selected location will be immediately safe in the service class and the user will jump back to the main screen.

Step 3 - `MainActivity`

On top of the screen the user should recheck his selected location, usually he will see the selected address with street and city. The `Geocoder` class enables us to convert (*reverse geocoding*) the latitude and the longitude from a location. In case the user is happy with the selection, he will notice that the other button is already enabled and should press it to come to the next and best step.



Javeed and Zara

Step 4 - `LocationService (+ home screen)`

In case, the user did not started his GPS connection yet, he will be asked to do that.

But the application for it self is finally setted up, that is why the user will be directed to his home screen and just can enjoy his journey. Sleeping or have a nice conversation in the train. As in the preamble quick described, the `LocationService` will run in the background and will be updated at least every 3 minutes or every kilometre. Currently the values are hard coded.

```
locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER,
DEFAULT_TIME, DEFAULT_DISTANCE, locationListener);
```

Step 5 – `AlarmActivity`

When the user arrives in the `DEFAULT_Radius` and the `currentLocation` is already updated, the `LocationService` will call the `AlarmActivity`.

```
locationListener = new LocationListener() {
      @Override
      public void onLocationChanged(Location location) {
            currentLocation = location;

            if(currentLocation.distanceTo(destination) <= DEFAULT_RADIUS) {
                  startActivity(
                  new Intent(getApplicationContext(), AlarmActivity.class)
                  .setFlags(Intent.FLAG_ACTIVITY_NEW_TASK));
            }
      }
}     /* Note: This is just a simplified version of the real code (you can
      find the complete statement in LocationService#callLocationListener() */
```

The `FLAG_ACTIVITY_NEW_TASK` has to be setted, because we are going to call the activity from outside of any activity context (in this case the home screen or any other application).

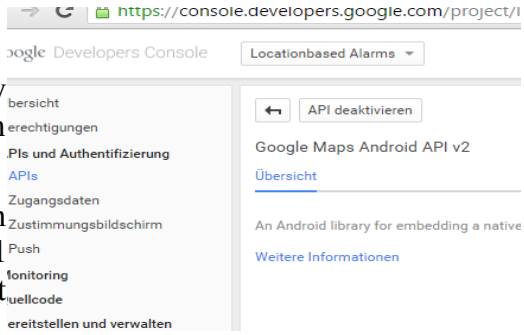
A simple song (saved in the application) will start to wake up the user. We decided to use the MediaPlayer instead of the `AlarmClock` class, because the alarm class is orientated by time, but we need a specific event (`currentLocation.distanceTo(destination) <= DEFAULT_RADIUS`).

13. August 2015, Horsens

Some Issues

So far, we really enjoyed the work on the application, because we feel the need, but of course we had some issues while implementing and testing:

Google Map API v2



To implement the Google Maps API v2 is not really complicate, but there are still some steps to follow, which are easy to make wrong.

In our case we got the **API key** to implement in `google_maps_api.xml`, but we forgot to add the **SSH_1 hash key** to the google account and deploy the project afterwards. That were the moment we got access, to the map.

Still we were not able to set markers on the map nor to catch a location. A big internet research could not help us as well, for some reason, we decided to use the **`SupportMapFragment`** instead of the `MapFragment` to get access with to our map again. And it worked, we still not really understand the reason, because it says, that you just have to use the `SupportMapFragment` if you code in a Android API less then 11, what we do not do.

To run the application in the background and use services

A nice way to really understand how services work. We spend a some time on this task, because we decided to make a own class for the service connection, and got a bit confused by that. The simple way is to have a own service connection instance in every activity, where you need to run some functionality or safe data, after the activity for it self is destroyed. In our case the `MainActivity` and the `MapsActivity`.

Testing

Testing a application like that takes a lot of time. Especially when you decide to use your own device instead of an emulator like Genymotion, because you can not fake all data to test the application. To see if the reason of some `Exception`, you have to be connected with your computer and see the error log. Same for the debugging part.

To be honest, that were one of the most annoying and in same time most fun part of doing this task.

Note - no GPS connection, no alarm!

It might be a problem in fast trains, that you will not have a GPS connection or you will just have it when you already arrived. In this case the alarm will ring very late and the traveller have to rush to get out of the train.

Less Bugs

<Empty> - We could not find any bugs so far which are not yet solved. 

Javeed and Zara

13. August 2015, Horsens

A Future View

As nice this application is, we already figured some disadvantage and some nice to haves for future versions out:

A Real Disadvantage

The application will use a lot of **battery**, especially if you will have a lot of checks of the `currentLocation` so it might be smart, to let the user choose between different battery intensive kinds (e.g. slow train vs. fast train), the different kinds will be a smart combination of the `minTime` and the minDistance of the `LocationManager#requestLocationUpdates()` method. Because, of course if you just make a few checks, the application will not be very exactly anymore or it even could happen, that the traveller will miss his destination, because the checks are to less, in a fast train.

Some Nice To Haves

For the future versions it could be nice, to implement a **vibration** instead of an alarm, or even to play a different songs. We also really would like to focus on the **design** of the application a bit more, because that is a unique selling point which should not be forgotten.

The last but not least nice to have, is of course to implement a database, so the user can be able to safe some more locations, give them self selected names (e.g. home, work, sport ..) or to view a history. We would recommend to use a **SQLite** database, because it will be much more faster then using a web service and to safe some locations on the current device will not take to much storage.

Enjoy ***Locationbased Alarms*** – Javeed and Zara!

https://github.com/zaraG/LocationbasedAlarms

Javeed and Zara