Homework 1 -- MUC Winter 2015/2016

Deadline: October 25th, 23:59.
Note: The assignments are to be done strictly individually.

This is the first of the three homework assignments that will guide you towards developing a fully-functioning Android mobile sensing app. The end goal is to have an app that monitors a user's wireless connectivity, and collects geotagged information about the wireless connection quality. This information is then presented to a user in two different forms: on an interactive map, and via a list of entries. In addition, the app ensure data persistency and sharing via remote cloud services.

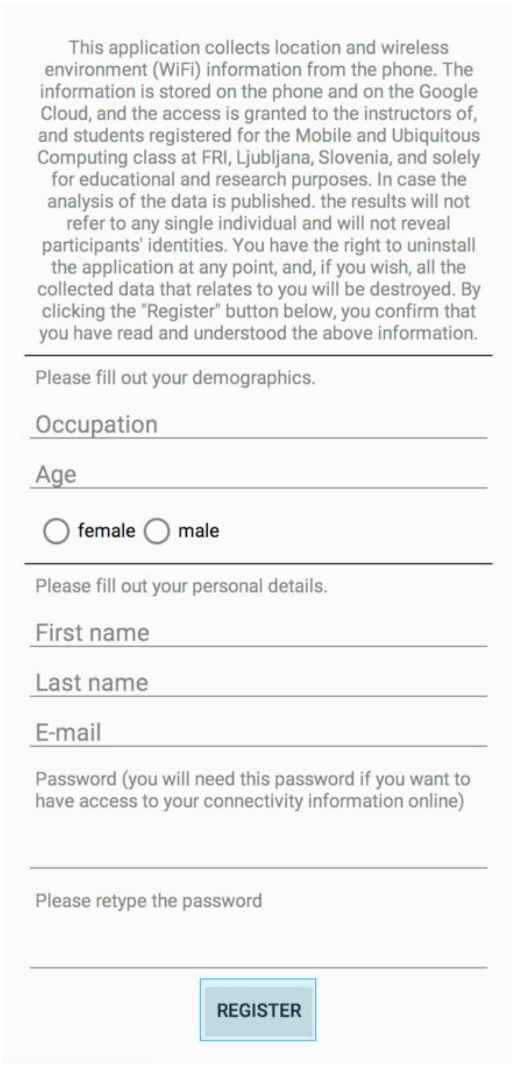
In homework 1 we concentrate on:

- Collecting a user's (manual) input
- Storing and retrieving the data from the application-allocated memory
- Interacting with the user through Tabs and Fragments
- Displaying a Google Map to a user
- Supporting conditional navigation, so that the app behaves differently depending on whether it was launched for the first time or not.

To successfully complete the homework you need to create an Android Studio project, and all the source and resource files that will make the app behave according to the detailed specification below.

Prerequisites

We will use git for tracking changes in your code. In addition, we suggest that you use git for collaborative programming within your class project. Git is a tool for managing your code, and it allows you to track changes to your codebase and collaborate remotely. GitHub and Bitbucket are two most popular online git repository hosting services. You can sign up for any of them, but note that a Bitbucket account allows you to create private repositories by default, while with GitHub you get to create private repositories only if you pay or subscribe to the educational account.
Linking your Github/Bitbucket account to an Android Studio project on your computer will allow you to work on your code on your desktop, and every once a while "push" the code to a Github/Bitbucket server. This will allow us to "pull" your code when the homework deadline comes.


This application collects location and wireless environment (WiFi) information from the phone. The information is stored on the phone and on the Google Cloud, and the access is granted to the instructors of, and students registered for the Mobile and Ubiquitous Computing class at FRI, Ljubljana, Slovenia, and solely for educational and research purposes. In case the analysis of the data is published. the results will not refer to any single individual and will not reveal participants' identities. You have the right to uninstall the application at any point, and, if you wish, all the collected data that relates to you will be destroyed. By clicking the "Register" button below, you confirm that you have read and understood the above information.

Please fill out your demographics.

Occupation

Age

◯ female ◯ male

Please fill out your personal details.

First name

Last name

E-mail

Password (you will need this password if you want to have access to your connectivity information online)

Please retype the password

REGISTER

So, let's go step by step:

1. Install git (but first check if it's already there). If you're on Ubuntu "**sudo apt-get install git"** should be sufficient. On a Mac you can get a dmg from http://git-scm.com/download/mac , while on Windows you can get it from https://git-scm.com/download/win .
2. Create an Android Studio project for your homework. You can use the same project for all the homeworks. We will just pull the code at different deadlines. Link this project with Bitbucket/Github. For Bitbucket, you need to install the Bitbucket plugin and to create a remote repository (via

Bitbucket webpage) first. Full instructions can be found here:
http://www.goprogramming.space/connecting-android-studio-project-with-bitbucket/
For GitHub, the support is already there, so inside Android Studio, just go to VCS->EnableVersionControlIntegration and select Git. Then just share project on GitHub via VCS->ImportIntoVersionControl->ShareProjectOnGitHub
The remote repository needs to be private!

3. Then, inside VCS->Git you have commit and push that allow you to synchronise your code with the remote server. Don't forget to push your code every once a while, but especially when your code is ready for marking!
4. Open GitHub/Bitbucket webpage and **add a read-only user** for your homework repository. if you're using Bitbucked add **veljkop** and if you're using github add **vpejovic**
5. **Don't forget to push your code before the deadline**, otherwise we can't evaluate your code!

Application Specification (in form of a use case)

Your application should support any device running Android API 11 and above. **The app should come with a name and a start icon of your choice (but not the default one).**

Application Components

RegistrationActivity
When run for the first time, your app should present a user with a **scrollable** registration screen. The screen should look something like the one in the figure. A user should read the consent form (please put the consent text in), fill out the details and click on the button "Register", so that her details are saved on the phone. Before saving the data, and registering the user, you should ensure that the entered details are valid:

- A user's first name, last name, password, occupation, age fields should not be blank
- Age can only be numeric
- A user's sex should be selected
- The password field and the repeat password fields must match
- The password field should not display the actual characters typed, but asterisks (*)
- The password should be between 8 and 16 characters
- The email address field should hold a valid email address string (hint).

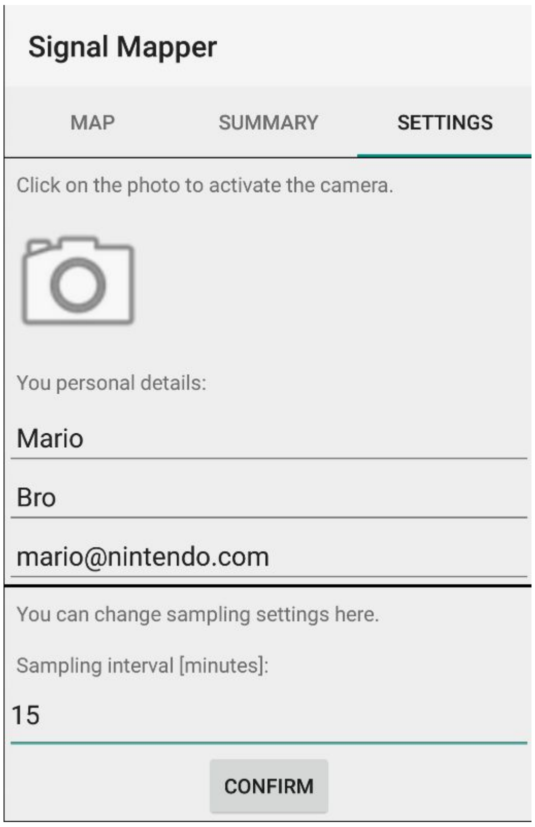
If any of the above are not satisfied, the user must be prompted to fix the exact field that is incorrectly filled. In addition, a universal unique identifier (UUID) must be created at the time of registration. Furthermore, instead of the cleartext password, we generate an MD5 hash value of it. Finally, the data (first name, last name, occupation, age, sex, email, md5 value of the password, UUID, registration time in milliseconds since epoch and the information on the model of the phone/device) are saved to SharedPreferences .

Once the registration is completed, as long as the application is not reinstalled, **the registration screen is not shown to the user again.**

MainActivity
After the registration (or immediately, in case the app is not run for the first time), the user is presented with the MainActivity which hosts a TabLayout with three fragments:

- ConnectionMapFragment
- SummaryFragment
- SettingsFragment

SettingsFragment fetches some of the the data back from SharedPreferences, such as a user's first name, last name and email, and shows each of them in a separate EditText field. In addition, the fragment features an EditText field titled "Sampling interval [minutes]" which allows a numeric input. Values of EditText fields in this fragment can be modified, and after a user clicks "Confirm", the values are saved to SharedPreferences. Just like in the RegistrationActivity above, you must validate that the first and the last name are not empty, and that the email address is valid.
The Settings fragment also features an initially empty, clickable, ImageView. When a user clicks on the view, an Intent is fired with [an action that activates a built-in camera](). The user can then take an image which is displayed in the ImageView.


Signal Mapper

MAP | SUMMARY | SETTINGS

Click on the photo to activate the camera.

You personal details:

Mario

Bro

mario@nintendo.com

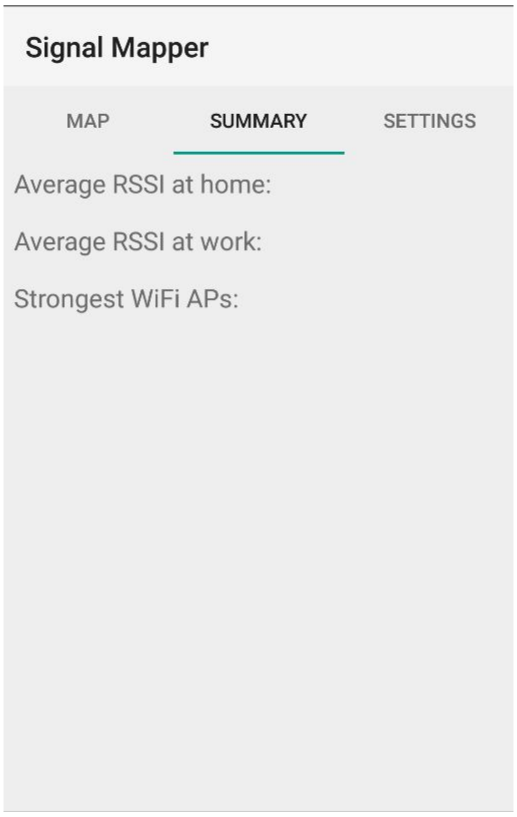
You can change sampling settings here.

Sampling interval [minutes]:

15

CONFIRM

SummaryFragment for now contains only the three TextView labels shown in the figure. In addition, it has a ListView that you can't see at the moment (as it is empty), and that we will later populate with the strongest WiFi access points.

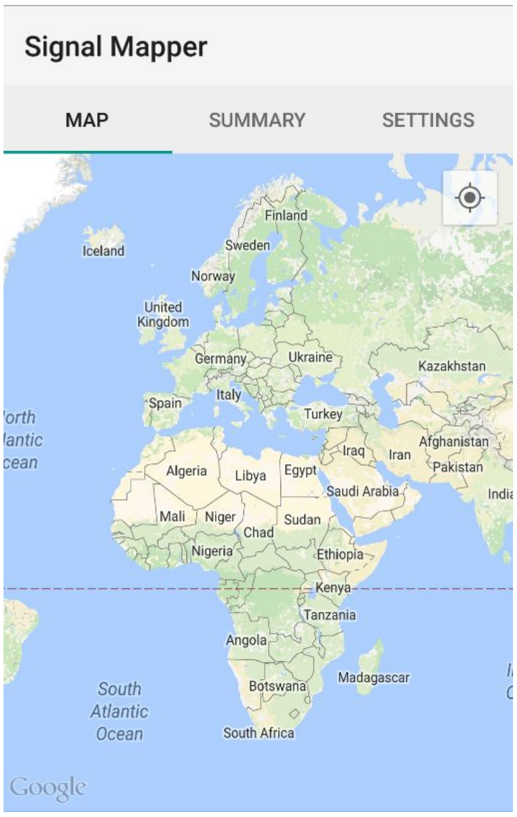

Signal Mapper

MAP | SUMMARY | SETTINGS

Average RSSI at home:

Average RSSI at work:

Strongest WiFi APs:



ConnectionMapFragment extends [SupportMapFragment]() and lets the user explore the map.

Things to keep in mind

- Spend some time thinking about the structure of your app. If you embrace the Model-View-Controller (MVC) paradigm early on, it will be much easier for you to manage your app once it gets larger.
- Use packages and classes wiseley. Perhaps you can look arrange the package names loosely around the MVC paradigm?
- As much as it is possible avoid hardcoding values. For example, translating your app to Slovene should involve merely creating a resource file with translated string values, nothing else.

The deliverables of the first homework

- The source code of your application will be checked out from the git account you provided at 00:00 on October 26th.
- An APK of your app, to be uploaded via ucilnica.