Android project proposals

Luca Bedogni (lbedogni@cs.unibo.it)

April 7, 2016

Introduction

In this document, we describe four possible projects for the exam of the "Laboratorio di applicazioni mobili" course. Each student can choose a project from the set, or suggest something else based on his/her personal interests. In this latter case, project proposals should be submitted via e-mail to Luca Bedogni (lbedogni@cs.unibo.it) , with a brief description of the application goals, contents and requirements.
The following project descriptions should be considered as hints. Students are strongly encouraged to expand the tracks, adding new features to the applications, and/or further customizing the contents.

1 Budget Tracking

This project proposal consists in the deployment of an Android application that enables budget tracking, and provides facilities to keep personal finances in order. In particular, the application should:

- Allow the tracking of everyday expenses
- Manage planned/periodic expenses
- Allow the browsing of expenses by date, and the generation of reports
- Compute and display useful statistics

In the following, each features is further discussed and analysed in detail.

1.1 Allow the tracking of everyday expenses

The application should allow a user adding information about each current expense, such as: current date, amount, description, category, etc. All the information must be stored locally on an internal database. Moreover, it must be given the possibility to save the location where a given expense has been performed (e.g. shop's location). Optional element:

- Save a picture of an item, acquired through the camera.

1.2 Manage and remind planned/periodic expenses

The application should allow a user adding information about periodic expenses (e.g. a loan payment) and about planned expenses in the future (e.g. a phone bill payment). The budget must be updated at the payment date, and periodic reminders should be shown 1 and 2 days before (e.g. through alert dialogs when the application starts, or through notifications in case the application can run in background mode).

1.3 Visualize and browse expenses by date

The application show allow a user visualing and browsing the budget and the list of expenses performed day by day, in the last week or in the last month. It must be given the possibility to save the generated report into a PDF file. In addition, it must be possible to display the locations on the Google Maps.

1.4 Provide weekly and monthly statistics

The application should foresee the possibility to compute and visualize useful statistics about weekly and monthly expenses performed by a user (e.g. total expenses for each category). Charts can be generated to visualize data.

2 Mobile Latex editor

LaTeX is a language to write documents, particularly used for scientific publications and reports. Via a LaTeX compiler, it is possible to generate PDF files from a LaTeX source file. However, the LaTeX compiler requires considerable resources in terms of memory and computation resources, and thus it might not be supported by a mobile device. The project consist in developing a LaTeX editor for the Android platform, that is able to compile the document on a remote server, and then open the corresponding PDF on the device in use. More in detail, the app should:

- Provide a text editor for managing text files.
- Support the LaTeX syntax.
- Foresee the possibility to compile the LaTeX source code on an external server, and then download the resulting PDF.

In the following, each feature is further discussed and analysed in detail.

2.1 Provide a text editor for managing text files

The application should provide classical functionalities of a text editor, i.e. the possibility to open, edit, save, close a file.

2.2 Support the LaTeX syntax

The application should be able to help the user in writing LaTeX code. Syntax highlighting should be provided. Moreover, the applications should provide facilities to insert LaTeX code (e.g. helping the user to insert math symbols).

2.3 Remote compiling

When the user clicks on the Compile button, the file should be automatically transfered to a remote server, where a PDF compiler is working. Once the PDF has been generated, it should be transferred back to the mobile applications. In this case, an Intent should be generated to open the PDF file. Optional elements:

- Handle the compiler logs (e.g transfer them back to the mobile applications)
- Manage .tex files with images. In this case, the app should be able to compress the folder in which the source files are (i.e. tex files and images), and send it to a remote server. At server-side, the compressed archive should be extracted, compiled, and the corresponding PDF should be returned back to the device. After that, the extracted directory on the remote server should be deleted.

3 Crowdsense Application

This project proposal consists in the development of an Android Application that is able to sense data coming from different sensors on the smartphone, and report them to a webservice. The app should be also able to display charts about the sensed data, and also statistics on them.

In the following, each features is further discussed and analysed in detail.

3.1 Sensor Configuration

The application should provide a screen in which are displayed all the sensors installed on the smartphone, including also cellular connections and WiFi. Each user using the app should be able to select which sensors she/he wants to sense. These data should be sensed periodically, and reported to an external webservice.

3.2 Send Data

The data sensed should be periodically sent to a remote webservice. You can implement your own, or use services like ThingSpeak. The app should run at boot, and report the data in the background, according to a time interval the user can choose. Users should also be able to decide whether to report the data only when using or WiFi, or also when using cellular.

So when the user defines a time interval T between readings, each T seconds the application should sense all the sensors selected by the user. If the actual connection matches the selection of the user (just wifi or also cellular), the data should be uploaded immediately. Otherwise, it should be kept locally (csv, database, you can choose), and sent when the connection is available.

3.3 Statistics

The application should be able to display statistics and charts directly on the phone. The data should be downloaded from the webservice. Daily/Weekly/Monthly statistics should be provided, and the user should be able to switch between them and visualize them.

4 IFTTT Engine

This project proposal consists in the deployment of an Android application that performs automatic actions and triggers when specific conditions are met, on the basis of the popular If-this-then-that programming (IFTTT) paradigm. In particular, the application should:

- Recognize a set of pre-defined contexts (e.g. attending a meeting)
- Capture a set of pre-defined events (e.g. a phone call is incoming)
- Define a list of possible actions (e.g. turn off the ring tones)
- Build an engine that allows a user to specify the default action to perform when a context is recognized and/or an event is generated (e.g. turn off the ring tones when there is a phone call incoming and I'm attending a meeting).

In the following, each features is further discussed and analysed in detail.

4.1 Context Recognition

The application must be able to recognize a predefined set of contexts that characterize the current user's operations. Basic context can be defined in the basis of: (i) temporal information (e.g. any time between 8-9am) and/or (ii) spatial information (e.g. I'm close to the DISI department) and/or (iii) mobility information (e.g. I'm moving at speed higher than 10Km/h). The user must be given the possibility to provide a context's name, and the information that characterize the context. Optionally:

- More fine-grained contexts can be recognized considering the information provided by embedded sensors (e.g. accelerometers), the radio interfaces (e.g. Wifi) or the photocamera/microphone.

4.2 Event Recognition

The application must be able to capture and recognize a list of external events that might occurr on the smartphone. Examples of events: phone call, SMS reception, open WiFi network detected, Bluetooth connection detected, etc

4.3 Action Definition

The application must provide a list of pre-defined actions and notifications that can be executed. Three major categories of actions must be considered:

- Setting manipulation: these actions modify the smartphone setting (e.g. turn on the vibrations).
- Notifications: these actions recall the user's attention through status-bar notifications.
- Social network: these actions involve operations on social media (e.g. publish a state update on Facebook) Additional and more specialized categories of actions can be defined.

4.4 IFTTT Rules Definition

The application must allow a user specifiying IFTTT rules. An IFTTT rule is composed by an action, a context (optional) and an event (optional). When the context/event is recognized, the corresponding action must be performed. The application must be able to run in background in order to continously monitor the current context and capture incoming events. Optional elements:

- Context/event can be combined through boolean operators AND, OR, NOT.
- Multiple actions can be defined for the same context/event.

5 Bologna OpenData

The Bologna Municipality has made available several data in an open format, through the website `http://dati.comune.bologna.it/`. These include more than 800 different dataset concerning the mobility, municipal buildings, and many more.

The goal of the project is to leverage on the data provided to offer different kind of services to the user. The application shoukd be able to update the data by using the permalink API offered by the municipality, the details of which are published on the specific dataset to be used.

This project does not have specific guidelines, as everything depends on the chosen dataset. Thus, students are advised to check with Dr. Luca Bedogni before starting to develop their own project.

6 Project bonus

While for the exam it is sufficient to implement the requirements presented above, we strongly encourage to enrich the application by adding new original features to the applications.

7 Projects submission

Projects must be submitted through email to lam-projects@cs.unibo.it, including all the code, a technical report, and a presentation (10-15 slides).