Assignment 3: Mobile software

Group 14

Jens Schreiber – 318912

Raúl García - 273785

Contents

[The underlying technology: 3](#_Toc322902005)

[Basic integration: 3](#_Toc322902006)

[Hello World: 4](#_Toc322902007)

[Using The Network + Using UI Elements: 4](#_Toc322902008)

[Parsing Simple Data: 4](#_Toc322902009)

[Deep integration: 4](#_Toc322902010)

[Important sourcecode folders: 5](#_Toc322902011)

[Information Visualizer: 5](#_Toc322902012)

[How to make it work? 5](#_Toc322902013)

## The underlying technology:

**Android** is a Linux-based operating system for mobile devices such as smartphones and tablet computers. It is developed by the Open Handset Alliance, led by Google, and other companies.

Android uses a model view control. So we have on one hand the view, which are represented in xml-files in the res folder. For the datamodel we can use databases, in our case we are just storing the information receiveing from servers on objects such as the Class Post. And controller are more or less every class which handle the workflow.

**Google APIs** a.k.a. Optional APIs: This APIs are optional because them are not supported by every android smartphone. We use the Location-based services.

**Django** is a Web-Framework, programmed in Python. It is following the Model-View-Controller-Schema. This Schema is an architectural design for the structure of software. It divides the software in model (data model), view and controller. A more detailled description can be found in the documentation of the assignment\_2.

**Facebook SDK for Android** provides Facebook Platform support for Android apps. It enables to implement Single Sign On and access the Facebook Platform APIs including the Graph API, FQL, and Dialogs.

**GraphAPI** makes developers easy to write and read data from the network of Facebook. According to a request you receive a response with the according information.

**GeoCoder:** With the GeoCoder it is possible to get latitude and longitude from an given Location via Google. But because of an know bug in the emulator, we haven’t been able to use it. Instead we used a request method which results in a JSON object. The same we have used in assignment\_2. With the different that we used Java to parse it instead of JavaScript and Python.

## Basic integration:

We have choosen Android platform to develop our project. First of all we need to prepare all the environment. We have use eclipse SDK with ADT android plugin, and the Android SDK.

We used the Android API 10 (API 9 was not available) with the target Google API 10 for maps compatibility issues.

To make the basic integration part we had to deal with some Android issues:

- Declare the permissions of our application the android manifest.

- Make the layout of the view to organize the elements.

- Use the “res” folder for all the resources of the app, as pictures.

### Hello World:

The Hello World Activity is not displayed, because of the fact that we have programmed various activities and view. So that it seems not to be necessary to add this as well.

### Using The Network + Using UI Elements:

Can be finded in the BasicInfos tab. In the beginning of the tab you can see the content of the downloadwd file and the modifcation date. Also it is possible to download a picture with a given URL by pressing the Confirm URL.

### Parsing Simple Data:

For the simple data we parsed some data for the Weather from Yahoo. Which you can see in the Weathertab. More or less we can use the Saxparser to get the data from an URI, and then parse the receving data and displaying it in the activity. In general we could reuse some developed functions from the first assignment. But we developed them further and splited them in the XML- and HTTPHelper class. So that we also can handle get- and post-HTTP requests.

## Deep integration:

We have decide to implement a mobile application for our application “SocialGags”, from the previous assignment.

We have adapted our SocialGags REST resources, to serve also to our mobile app. We implemented it checking in each request the user agent and answering xml files with the information that the device needs to the request if the user agent match with the user agent of the app.

resources:

/mobilelogin - This resource is new, receive with POST the necessary info to make the facebook login

/(root) - XML file with all the information of the last post on SocialGags

/location - XML file with the information of the location of all the user friends

/users/<userTarget> - XML file with the details of <userTarget>

/post/create - receive request with POST information to create a new post, the response is a XML file with the information about the new post created

/post/<userTarget>/<postName> - XML file with the information of a specific post of an specific user

/post/<userTarget> - XML file with all the post published by the <userTarget>

The login in our system is based in the login with facebook. In the database we store the Facebook ID of our users as the username.

When a user makes login from the app this login is directly through facebook, and after that the app send the credentials to our server, which verify the information with the database.

To parse the XML files we have adapted the parser we used in the first assignment.

### Important sourcecode folders:

src: contains the controllers and the activities

gen: there you can find the generated R file, which contains all id for layouts, button, imageview and also the other stuff. With this file and the function findViewByID we can find the different view from the res folder in our Java-Classes

res/drawable: pictures and also xml-styles:

res/layout: the different layouts for the activity

res/menu: for the developed menu, in the beginning it was used instead of the tabView to browse the different activities in our SocialGag app

res/values: there are the strings for the default language

src\_01: copied and modified django server from assignment\_2

AndroidManifest: handles all permission, different activities, libraries, camera,...

### Information Visualizer:

This couldn’t be finished in time. But because of the fact we have all the parsers for XML it wouldn’t be a problem to implement it.

### How to make it work?

#### Server:

Connect to the naf server and change directory to

/home/group14/repo/group14/assignment\_3/src\_01/deepint and run “./run.sh”

#### Android:

Basically to get an running application in an emulator you have to do the following steps:

1. Start Eclipse in your own workspace

2. Import Existing Project and choose the assignment\_3 folder from you svn system

3. Copy the facebook-sdk from tools into a folder outside of the assignment\_3 folder, mayber even outside of the workspace. Otherwise it will not work.

4. Create a new AndroidProject from existing source and choose the root folder of the copied facebook-sdk. Don’t forget to choose the Android-API 10

5. assignmet\_3--> properties --> android tab --> add the facebook-sdk as a library to the project

6. RUN the emulator

7. Install the facebook application on your emulator with the adb, located in tools