JUST DO IT
HÖHERE ABTEILUNG FÜR INFORMATIONSTECHNOLOGIE

tgm
Die Schule der Technik

Laborprotokoll
MOBILE ACCESS TO WEB SERVICES

Systemtechnik Labor
5BHIT 2015/16, Gruppe A

Hagen Aad Fock

Version 0.1

Betreuer: Prof. Borko
Note:

Begonnen am 19.02.2016
Beendet am 26.02.2016

Inhaltsverzeichnis

Einführung

Diese Übung gibt einen Einblick in Entwicklungen von mobilen Applikationen.

Ziele

Das Ziel dieser Übung ist eine Anbindung einer mobilen Applikation an ein Webservices.

Die Anbindung soll mit Hilfe eines RESTful Webservice (Gruppe1) umgesetzt werden.

Voraussetzungen

- Grundlagen Java und XML
- Grundlegendes Verständnis über Entwicklungs- und Simulationsumgebungen
- Verständnis von RESTful Webservices

Aufgabenstellung

Es ist eine mobile Anwendung zu implementieren, die sich an das Webservice aus der Übung DezSysLabor-09 "Web Services in Java" anbinden soll. Dabei müssen die entwickelten Schnittstellen entsprechend angesprochen werden.
Es ist freigestellt, welche mobile Implementierungsumgebung dafür gewählt wird. Empfohlen wird aber eine Implementierung auf Android

Bewertung: 16 Punkte
- Anbindung einer mobilen Applikation an die Webservice-Schnittstelle (6 Punkte)
- Registrierung von Benutzern (3 Punkte)
- Login und Anzeige einer Willkommensnachricht (3 Punkte)
- Simulation bzw. Deployment auf mobilem Gerät (2 Punkte)
- Protokoll (2 Punkte)

Ergebnisse

Für die Umsetzung dieser Übung habe ich das Tutorial von Android Guru [1] befolgt.
Nachdem ich den zur Verfügung gestellten Sourcecode importiert hatte mussten noch einige Fehler behoben werden bis die Android App mit meiner REST Webservice-Schnittstelle, aus der Aufgabe DezSys09 [4], kommunizieren konnten.

Anbindung einer mobilen Applikation an die Webservice-Schnittstelle (6 Punkte)

Das erste Problem war, dass ich von dem Emulator auf den Localhost zugreifen musste. Nach einer kurzen Recherche bin ich auf einen Artikel von Stackoverflow [3] gestoßen. Man muss bei der Simulation, statt Localhost die IPv4 Adresse 10.0.2.2 angeben.

Das zweite Problem war, dass meine DezSys 09 [4] Aufgabe mit POST und JSON realisiert wurde und das Tutorial von Android Guru [1] ist auf GET ausgelegt. In den Klassen LoginActivity und RegisterActivity musste das RequestParams Objekt auf ein JSONObject geändert werden.

ALT

```
// Instantiate Http Request Param Object
RequestParams params = new RequestParams();
```

NEU

```
JSONObject params = new JSONObject();
```

Das JSON muss klarerweise mit den angegebenen Daten befüllt werden.

```
StringEntity request = null;
try {
   request = new StringEntity(params.toString());
   request.setContentType(new BasicHeader(HTTP.CONTENT_TYPE,
"application/json"));
} catch (UnsupportedEncodingException e) {
   e.printStackTrace();
}
```

Weil man jetzt die Request-Art von GET und RequestParams auf POST und JSONObject geändert hat muss auch noch die Variante der Verbindung vom AsyncHttpClient abgeändert werden.

ALT

```
AsyncHttpClient client = new AsyncHttpClient();
client.get("http://192.168.2.2:9999/useraccount/login/dologin",params ,new
AsyncHttpResponseHandler()
```

NEU

```
AsyncHttpClient client = new AsyncHttpClient();
client.post(this.getApplicationContext(), "http://10.0.2.2:8080/register",
request, "application/json", new TextHttpResponseHandler() {
```

Die onSuccess Methode musste dann auch noch geändert werden

ALT

```
@Override
            public void onSuccess(String response) {
                // Hide Progress Dialog
                prgDialog.hide();
                try {
                        // JSON Object
                        JSONObject obj = new JSONObject(response);
                        // When the JSON response has status boolean value
assigned with true
                        if(obj.getBoolean("status")){
                            Toast.makeText(getApplicationContext(), "You
are successfully logged in!", Toast.LENGTH_LONG).show();
                            // Navigate to Home screen
                            navigatetoHomeActivity();
                        }
                        // Else display error message
                        else{
                            errorMsg.setText(obj.getString("error_msg"));
                            Toast.makeText(getApplicationContext(),
obj.getString("error_msg"), Toast.LENGTH_LONG).show();
                        }
                } catch (JSONException e) {
                    // TODO Auto-generated catch block
                    Toast.makeText(getApplicationContext(), "Error Occured
[Server's JSON response might be invalid]!", Toast.LENGTH_LONG).show();
                    e.printStackTrace();

                }
            }
```

NEU Registrieren

```
@Override
public void onSuccess(int statusCode, Header[] headers, String
responseBody) {
    prgDialog.hide();
    // When Http response code is '201'
    if (statusCode == 201) {
        Toast.makeText(getApplicationContext(), responseBody,
Toast.LENGTH_LONG).show();
        // Navigate to Home screen
        navigatetoLoginActivity(findViewById(android.R.id.content));
    }
}
```

NEU Login

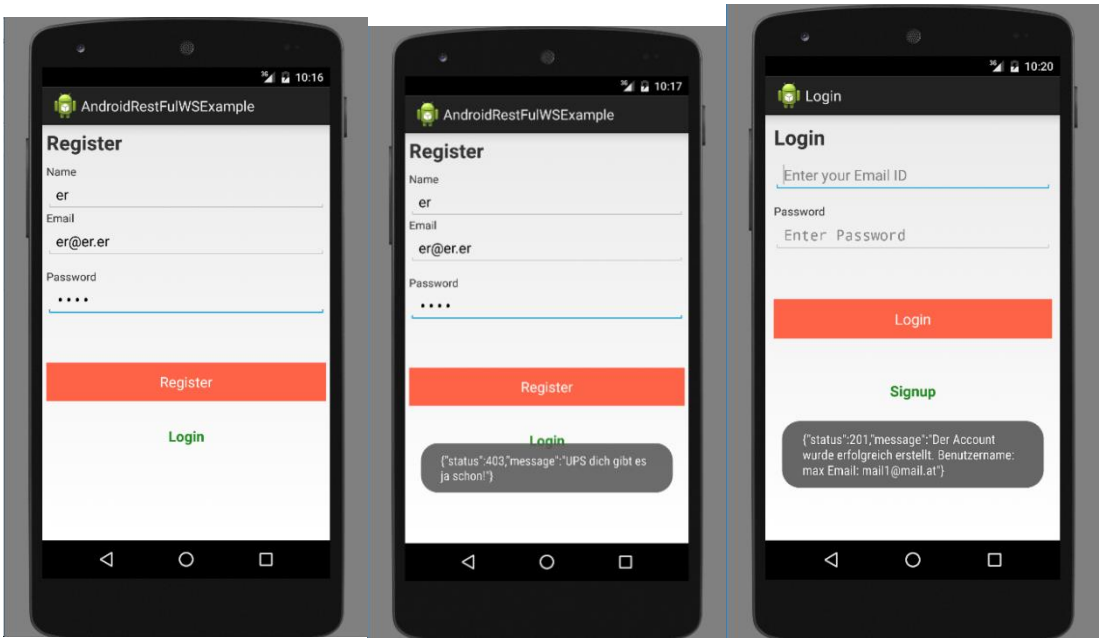
```
@Override
public void onSuccess(int statusCode, Header[] headers, String
responseBody) {
    prgDialog.hide();
    // When Http response code is '200'
    if(statusCode == 200) {
        Toast.makeText(getApplicationContext(), responseBody,
Toast.LENGTH_LONG).show();
        // Navigate to Home screen
        navigatetoHomeActivity();
    }

}
```

Nach diesen Schritten hat es schlussendlich funktioniert.

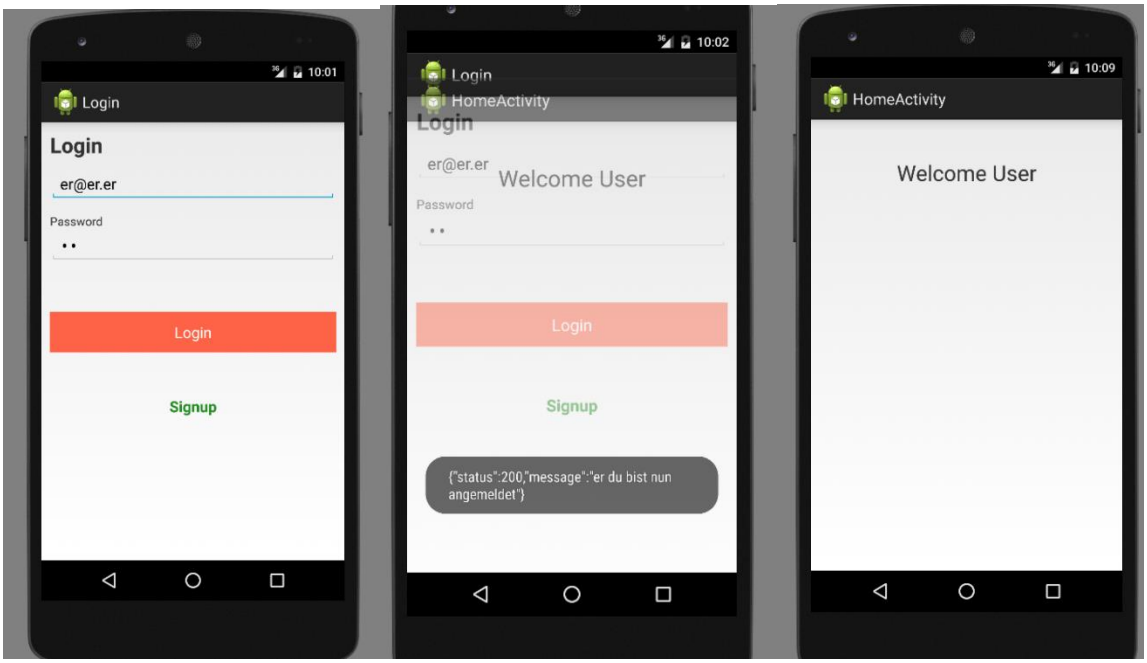
Registrierung von Benutzern (3 Punkte)

Nun da die Schnittstelle konfiguriert ist, lege ich einen User an.



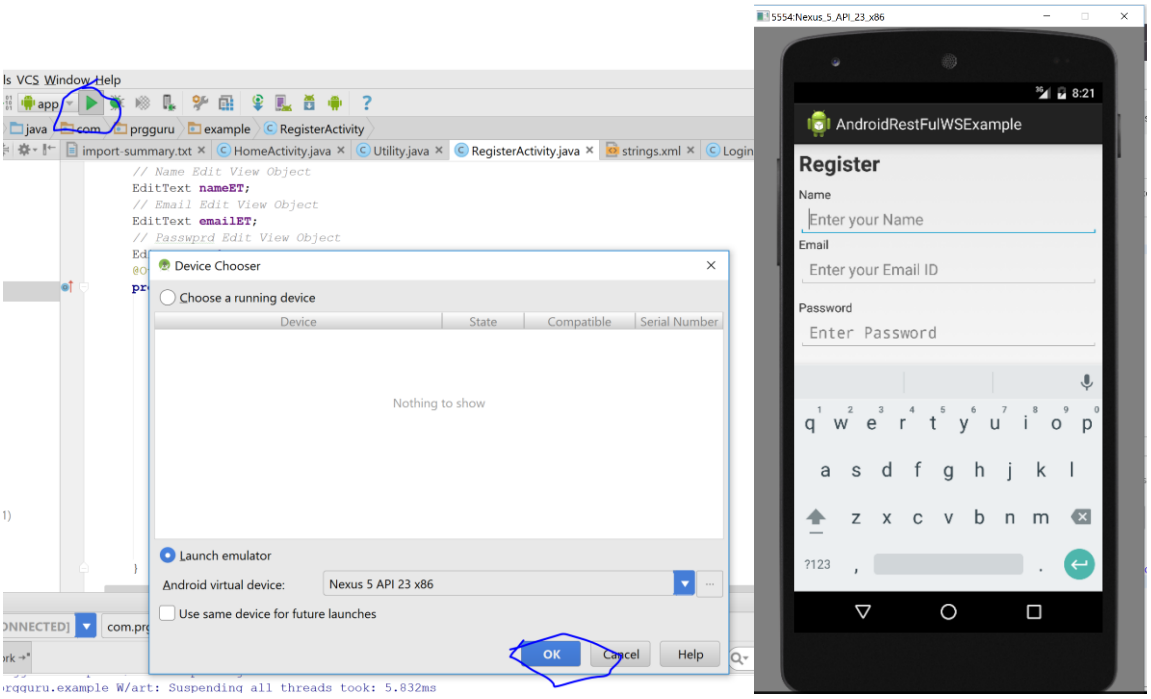
Login und Anzeige einer Willkommensnachricht (3 Punkte)

Nachdem ich einen User in meiner Datenbank abgespeichert habe kann man sich auch als dieser anmelden.



Simulation bzw. Deployment auf mobilem Gerät (2 Punkte)

Nachdem ich den Sourcecode vom Android Guru Tutorial hatte, konnte ich schon den Code auf eine Simulation deployen.


Device Chooser
Choose a running device
Device
State
Compatible
Serial Number
Nothing to show
Launch emulator
Android virtual device:
Nexus 5 API 23 x86
Use same device for future launches
OK
Cancel
Help
5554:Nexus_5_API_23_x86
AndroidRestFulWSExample
Register
Name
Enter your Name
Email
Enter your Email ID
Password
Enter Password

Githublink

Github hfock DezSys11
https://github.com/hfock-tgm/DezSys11.git

Quellen

[1] "Android Restful Webservice Tutorial – How to call RESTful webservice in Android – Part 3"; Posted By Android Guru on May 27, 2014; online: http://programmerguru.com/android-tutorial/android-restful-webservice-tutorial-how-to-call-restful-webservice-in-android-part-3/
[2] "Referenzimplementierung von DezSys09"; Paul Kalauner; online: https://github.com/pkalauner-tgm/dezsys09-java-webservices
[3] „How to connect to my http://localhost web server from Android Emulator in Eclipse “; stackoverflow; online: http://stackoverflow.com/questions/5806220/how-to-connect-to-my-http-localhost-web-server-from-android-emulator-in-eclips
[4] “DezSys09”; hfock-tgm; online: https://github.com/hfock-tgm/DezSys09.git