JUST DO IT
HÖHERE ABTEILUNG FÜR
INFORMATIONSTECHNOLOGIE

tgm
Die Schule der Technik

Protokoll

DezSys09: Web Services in Java

Dezentrale Systeme
5BHITT 2015/16

Melanie Goebel

Version 1.0
Begonnen am 12.02.2016
Beendet am 18. Februar 2016

Note:
Betreuer: M.Borko

Inhaltsverzeichnis

1 Einführung

Diese Übung zeigt die Anwendung von mobilen Diensten in Java.

1.1 Ziele

Das Ziel dieser Übung ist eine Webanbindung zur Benutzeranmeldung in Java umzusetzen. Dabei soll sich ein Benutzer registrieren und am System anmelden können.

Die Kommunikation zwischen Client und Service soll mit Hilfe von JAX-RS (Gruppe1) umgesetzt werden.

1.2 Voraussetzungen

- Grundlagen Java und Java EE
- Verständnis über relationale Datenbanken und dessen Anbindung mittels JDBC oder ORM-Frameworks
- Verständnis von Restful Webservices

1.3 Aufgabenstellung

Es ist ein Webservice mit Java zu implementieren, welches eine einfache Benutzerverwaltung implementiert. Dabei soll die Webapplikation mit den Endpunkten /register und /login erreichbar sein.

Registrierung
Diese soll mit einem Namen, einer eMail-Adresse als BenutzerID und einem Passwort erfolgen. Dabei soll noch auf keine besonderen Sicherheitsmerkmale Wert gelegt werden. Bei einer erfolgreichen Registrierung (alle Elemente entsprechend eingegeben) wird der Benutzer in eine Datebanktabelle abgelegt.

Login
Der Benutzer soll sich mit seiner ID und seinem Passwort entsprechend authentifizieren können. Bei einem erfolgreichen Login soll eine einfache Willkommensnachricht angezeigt werden.

Die erfolgreiche Implementierung soll mit entsprechenden Testfällen dokumentiert werden. Es muss noch keine grafische Oberfläche implementiert werden! Verwenden Sie auf jeden Fall ein gängiges Build-Management-Tool..

Bewertung: 16 Punkte

- Aufsetzen einer Webservice-Schnittstelle (4 Punkte)

- Registrierung von Benutzern mit entsprechender Persistierung (4 Punkte)
- Login und Rückgabe einer Willkommensnachricht (3 Punkte)
- AcceptanceTests (3 Punkte)
- Protokoll (2 Punkte)

2 Arbeitsdurchführung

Für die Aufgabe wurde ein Tutorial von dem Autor Android Guru [1] verwendet.

Folgende Schritte wurden dabei unternommen:

- Anlegen und Befüllen der Datenbank
- Jersey [2] herunterladen und entzippen
- Ein neues Projekt in Eclipse für JaveEE namens useraccount erstellt. (File»New»Dynamic Web Project)
- Jersey Library JARS in das Projekt unter WEB-INF/lib schieben
- JDBC-Connector JARS in das Projekt unter WEB-INF/lib schieben
- Jersey als Server dispatcher registieren (in web.xml Code hinzufügen, siehe Codesnippet 2)
- Package com.prgguru.jersey in src erstellen
- Code in das Package hinzufügen (Constants.java, DBConnection.java, Utility.java, Register.java, Login.java)
- Deploy Projekt: Run As » Run on Server
- Register: http://localhost:8080/javawebservice/register/doregister?name=ersterusername=ersterpasswo
- Login: http://localhost:8080/javawebservice/login/dologin?username=erster1password=1234

```
create database users
use users
CREATE TABLE IF NOT EXISTS 'user' (
  'name' varchar(50) NOT NULL,
  'username' varchar(50) NOT NULL,
  'password' varchar(50) NOT NULL,
  'register_dt' timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY ('username')
)
```

Listing 1: Datenbank erstellen und befüllen

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http
   ://java.sun.com/xml/ns/javaee" xmlns:web="http://java.sun.com/xml/ns/
   javaee/web-app_2_5.xsd" xsi:schemaLocation="http://java.sun.com/xml/ns/
   javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" id="WebApp_ID"
   version="2.5">
  <display-name>useraccount</display-name>
  <servlet>
    <servlet-name>Jersey REST Service</servlet-name>
```

```
    <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</
       servlet-class>
    <init-param>
      <param-name>com.sun.jersey.config.property.packages</param-name>
      <param-value>com.prgguru.jersey</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>Jersey REST Service</servlet-name>
    <url-pattern>/*</url-pattern>
  </servlet-mapping>
</web-app>
```

Listing 2: web.xml

```
package com.prgguru.jersey;
//Change these parameters according to your DB
public class Constants {
    public static String dbClass = "com.mysql.jdbc.Driver";
    private static String dbName= "users";
    public static String dbUrl = "jdbc:mysql://localhost:3306/"+dbName;
    public static String dbUser = "root";
    public static String dbPwd = "password";
})
```

Listing 3: Constants.java

```
package com.prgguru.jersey;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class DBConnection {
    /**
     * Method to create DB Connection
     *
     * @return
     * @throws Exception
     */
    @SuppressWarnings("finally")
    public static Connection createConnection() throws Exception {
        Connection con = null;
        try {
            Class.forName(Constants.dbClass);
            con = DriverManager.getConnection(Constants.dbUrl, Constants.dbUser, Constants.dbPwd);
        } catch (Exception e) {
            throw e;
        } finally {
            return con;
        }
    }
    /**
     * Method to check whether uname and pwd combination are correct
     *
     * @param uname
     * @param pwd
     * @return
     * @throws Exception
     */
    public static boolean checkLogin(String uname, String pwd) throws Exception {
```

```
        boolean isUserAvailable = false;
        Connection dbConn = null;
        try {
            try {
41              dbConn = DBConnection.createConnection();
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
46          Statement stmt = dbConn.createStatement();
            String query = "SELECT * FROM user WHERE username = '" + uname
                    + "' AND password=" + "'" + pwd + "'";
            //System.out.println(query);
            ResultSet rs = stmt.executeQuery(query);
51          while (rs.next()) {
                //System.out.println(rs.getString(1) + rs.getString(2) + rs.getString(3));
                isUserAvailable = true;
            }
        } catch (SQLException sqle) {
56          throw sqle;
        } catch (Exception e) {
            // TODO Auto-generated catch block
            if (dbConn != null) {
                dbConn.close();
61          }
            throw e;
        } finally {
            if (dbConn != null) {
                dbConn.close();
66          }
        }
        return isUserAvailable;
    }
    /**
71   * Method to insert uname and pwd in DB
     *
     * @param name
     * @param uname
     * @param pwd
76   * @return
     * @throws SQLException
     * @throws Exception
     */
    public static boolean insertUser(String name, String uname, String pwd) throws SQLException,
         Exception {
81      boolean insertStatus = false;
        Connection dbConn = null;
        try {
            try {
                dbConn = DBConnection.createConnection();
86          } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            Statement stmt = dbConn.createStatement();
91          String query = "INSERT into user(name, username, password) values('"+name+ "','"+"'"
                    + uname + "','" + pwd + "')";
            //System.out.println(query);
            int records = stmt.executeUpdate(query);
            //System.out.println(records);
96          //When record is successfully inserted
            if (records > 0) {
                insertStatus = true;
            }
        } catch (SQLException sqle) {
101         //sqle.printStackTrace();
            throw sqle;
        } catch (Exception e) {
            //e.printStackTrace();
            // TODO Auto-generated catch block
106         if (dbConn != null) {
                dbConn.close();
```

```
            }
            throw e;
        } finally {
            if (dbConn != null) {
                dbConn.close();
            }
        }
        return insertStatus;
    }
}
```

Listing 4: DBConnection.java

```
package com.prgguru.jersey;

import org.codehaus.jettison.json.JSONException;
import org.codehaus.jettison.json.JSONObject;

public class Utitlity {
    /**
     * Null check Method
     *
     * @param txt
     * @return
     */
    public static boolean isNotNull(String txt) {
        // System.out.println("Inside isNotNull");
        return txt != null && txt.trim().length() >= 0 ? true : false;
    }

    /**
     * Method to construct JSON
     *
     * @param tag
     * @param status
     * @return
     */
    public static String constructJSON(String tag, boolean status) {
        JSONObject obj = new JSONObject();
        try {
            obj.put("tag", tag);
            obj.put("status", new Boolean(status));
        } catch (JSONException e) {
            // TODO Auto-generated catch block
        }
        return obj.toString();
    }

    /**
     * Method to construct JSON with Error Msg
     *
     * @param tag
     * @param status
     * @param err_msg
     * @return
     */
    public static String constructJSON(String tag, boolean status,String err_msg) {
        JSONObject obj = new JSONObject();
        try {
            obj.put("tag", tag);
            obj.put("status", new Boolean(status));
            obj.put("error_msg", err_msg);
        } catch (JSONException e) {
            // TODO Auto-generated catch block
        }
        return obj.toString();
    }

}
```

Listing 5: Utility.java

```java
package com.prgguru.jersey;

import java.sql.SQLException;

import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.QueryParam;
import javax.ws.rs.core.MediaType;
//Path: http://localhost/<appln-folder-name>/register
@Path("/register")
public class Register {
    // HTTP Get Method
    @GET
    // Path: http://localhost/<appln-folder-name>/register/doregister
    @Path("/doregister")
    // Produces JSON as response
    @Produces(MediaType.APPLICATION_JSON)
    // Query parameters are parameters: http://localhost/<appln-folder-name>/register/doregister?name=pqrs&username=abc&password=xyz
    public String doLogin(@QueryParam("name") String name, @QueryParam("username") String uname,
        @QueryParam("password") String pwd){
        String response = "";
        //System.out.println("Inside doLogin "+uname+"  "+pwd);
        int retCode = registerUser(name, uname, pwd);
        if(retCode == 0){
            response = Utitlity.constructJSON("register",true);
        }else if(retCode == 1){
            response = Utitlity.constructJSON("register",false, "You are already registered");
        }else if(retCode == 2){
            response = Utitlity.constructJSON("register",false, "Special Characters are not allowed in Username and Password");
        }else if(retCode == 3){
            response = Utitlity.constructJSON("register",false, "Error occured");
        }
        return response;

    }

    private int registerUser(String name, String uname, String pwd){
        System.out.println("Inside checkCredentials");
        int result = 3;
        if(Utitlity.isNotNull(uname) && Utitlity.isNotNull(pwd)){
            try {
                if(DBConnection.insertUser(name, uname, pwd)){
                    System.out.println("RegisterUSer if");
                    result = 0;
                }
            } catch(SQLException sqle){
                System.out.println("RegisterUSer catch sqle");
                //When Primary key violation occurs that means user is already registered
                if(sqle.getErrorCode() == 1062){
                    result = 1;
                }
                //When special characters are used in name,username or password
                else if(sqle.getErrorCode() == 1064){
                    System.out.println(sqle.getErrorCode());
                    result = 2;
                }
            }
            catch (Exception e) {
                // TODO Auto-generated catch block
                System.out.println("Inside checkCredentials catch e ");
                result = 3;
            }
        }else{
            System.out.println("Inside checkCredentials else");
            result = 3;
        }

        return result;
    }
```

```
}
```

Listing 6: Register.java

```
package com.prgguru.jersey;

import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;
import javax.ws.rs.QueryParam;
import javax.ws.rs.core.MediaType;
//Path: http://localhost/<appln-folder-name>/login
@Path("/login")
public class Login {
    // HTTP Get Method
    @GET
    // Path: http://localhost/<appln-folder-name>/login/dologin
    @Path("/dologin")
    // Produces JSON as response
    @Produces(MediaType.APPLICATION_JSON)
    // Query parameters are parameters: http://localhost/<appln-folder-name>/login/dologin?username=abc&
        password=xyz
    public String doLogin(@QueryParam("username") String uname, @QueryParam("password") String pwd){
        String response = "";
        if(checkCredentials(uname, pwd)){
            response = Utitlity.constructJSON("login",true);
        }else{
            response = Utitlity.constructJSON("login", false, "Incorrect Email or Password");
        }
    return response;
    }

    /**
     * Method to check whether the entered credential is valid
     *
     * @param uname
     * @param pwd
     * @return
     */
    private boolean checkCredentials(String uname, String pwd){
        System.out.println("Inside checkCredentials");
        boolean result = false;
        if(Utitlity.isNotNull(uname) && Utitlity.isNotNull(pwd)){
            try {
                result = DBConnection.checkLogin(uname, pwd);
                //System.out.println("Inside checkCredentials try "+result);
            } catch (Exception e) {
                // TODO Auto-generated catch block
                //System.out.println("Inside checkCredentials catch");
                result = false;
            }
        }else{
            //System.out.println("Inside checkCredentials else");
            result = false;
        }

        return result;
    }

}
```

Listing 7: Login.java

Literatur

[1] Android Guru. Android restful webservice tutorial how to create restful webservice in java part 2. `http://programmerguru.com/android-tutorial/android-restful-webservice-tutorial-how-to-create-restful-webservice-in-java-part-2/`. zuletzt besucht: 12.02.2016.

[2] Oracle Cooperation. Jersey download. `https://jersey.java.net/download.html`. zuletzt besucht: 12.02.2016.

Tabellenverzeichnis

Listings

Abbildungsverzeichnis