Android Restful Webservice Tutorial – How to create RESTful webservice in Java – Part 2

Posted By Android Guru on May 11, 2014 | 26 Comments

In the Android RESTFul tutorial series , I am discussing about creating and invoking **RESTful webservice** in Android applications.

In this particular post, I am going to discuss about how to create RESTful webservice using Java.

If you are new to RESTful webservice, I would recommend you to take a look at Introduction to RESTful webservice

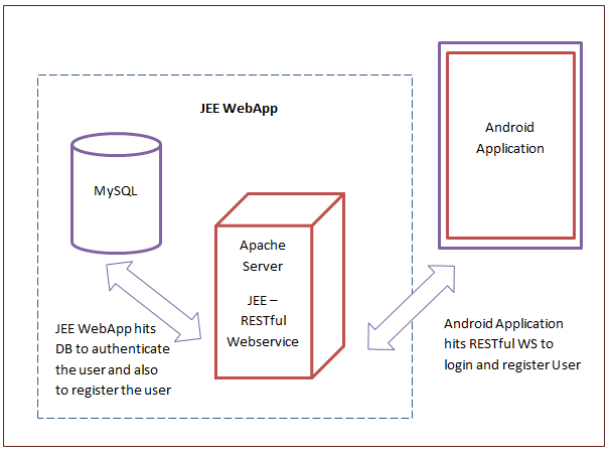
This tutorial series will be published as three parts:

- Part 1 – Introduction to RESTful webservice
- Part 2 – How to create RESTful webservice in Java?
- Part 3 – How to invoke RESTful webservice in Android applications?

How to create RESTFul webservice in Java?

In this post, I will be discussing about creating **RESTful webservice** in Java and in the next post will be discussing how to consume RESTful webservice we are about to create in Android application.

Here is the quick overview of the JEE webapp which we are going to create:



JEE webapp creation involves below steps:

1. Configure MySQL DB and Table
2. Create RESTful Webservice using Jersey

You can download source code from here if you don't want to create Application from scratch, otherwise please proceed with below listings.

MySQL DB and Table

Do follow below steps to create MySQL DB and Table.

Create MySQL DB and Table:

1. Create database called 'users' in phpMyAdmin or create it through command:

```
create database users
```

2. Select database users in phpMyAdmin or select it through command:

```
use users
```

3. Create table called 'user' in phpMyAdmin or through command by pasting below SQL script in SQL Query box:

```
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`)
)
```

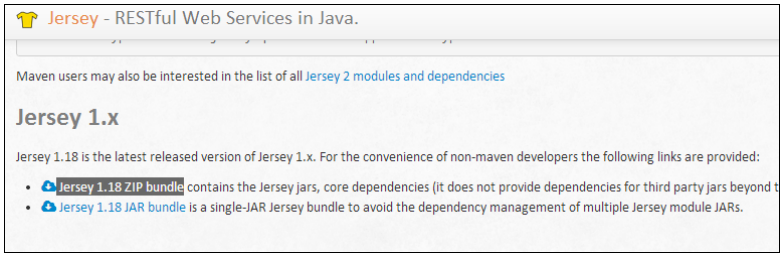

Jersey – RESTful Webservice

Am going to use Jersey framework to design RESTful webservice.

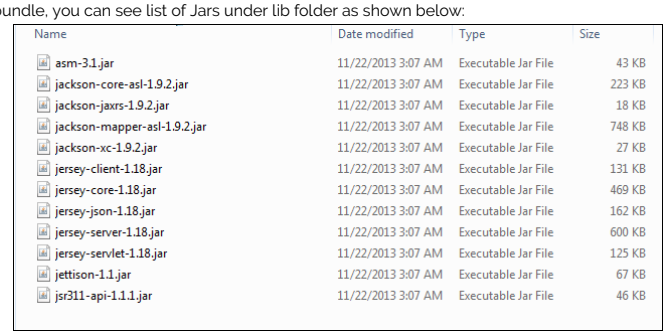
> Reference: http://www.vogella.com/tutorials/REST/article.html#installation_jersey

Do follow the below steps to create RESTful webservice:

1. Goto https://jersey.java.net/download.html and download Jersey 1.18 ZIP bundle as shown below:

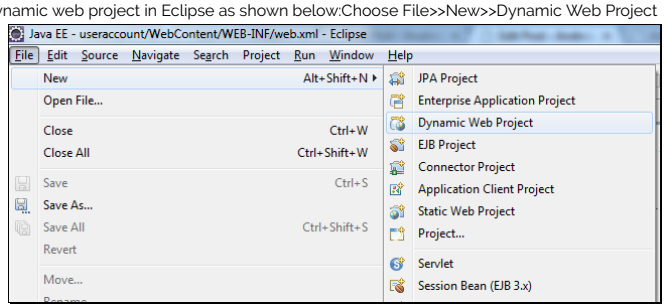


Unzip the bundle, you can see list of Jars under lib folder as shown below:

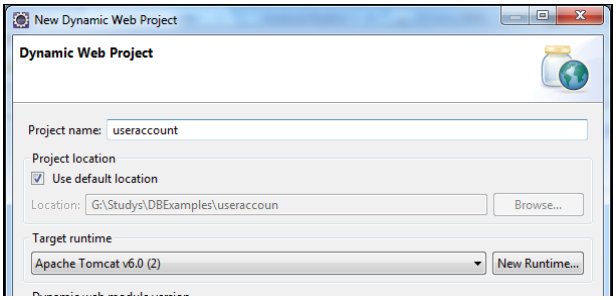


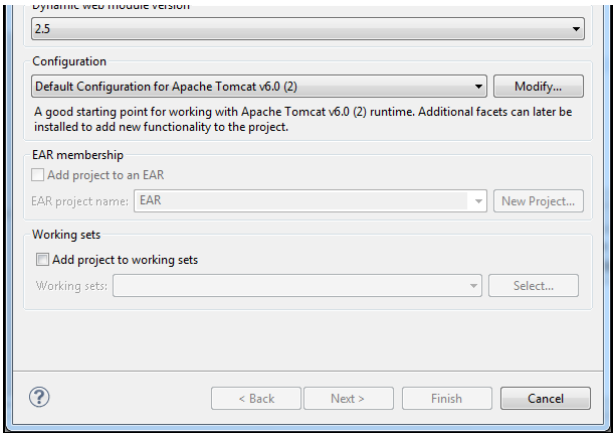
Name	Date modified	Type	Size
asm-3.1.jar	11/22/2013 3:07 AM	Executable Jar File	43 KB
jackson-core-asl-1.9.2.jar	11/22/2013 3:07 AM	Executable Jar File	223 KB
jackson-jaxrs-1.9.2.jar	11/22/2013 3:07 AM	Executable Jar File	18 KB
jackson-mapper-asl-1.9.2.jar	11/22/2013 3:07 AM	Executable Jar File	748 KB
jackson-xc-1.9.2.jar	11/22/2013 3:07 AM	Executable Jar File	27 KB
jersey-client-1.18.jar	11/22/2013 3:07 AM	Executable Jar File	131 KB
jersey-core-1.18.jar	11/22/2013 3:07 AM	Executable Jar File	469 KB
jersey-json-1.18.jar	11/22/2013 3:07 AM	Executable Jar File	162 KB
jersey-server-1.18.jar	11/22/2013 3:07 AM	Executable Jar File	600 KB
jersey-servlet-1.18.jar	11/22/2013 3:07 AM	Executable Jar File	125 KB
jettison-1.1.jar	11/22/2013 3:07 AM	Executable Jar File	67 KB
jsr311-api-1.1.1.jar	11/22/2013 3:07 AM	Executable Jar File	46 KB

2. Create Dynamic web project in Eclipse as shown below:Choose File>>New>>Dynamic Web Project

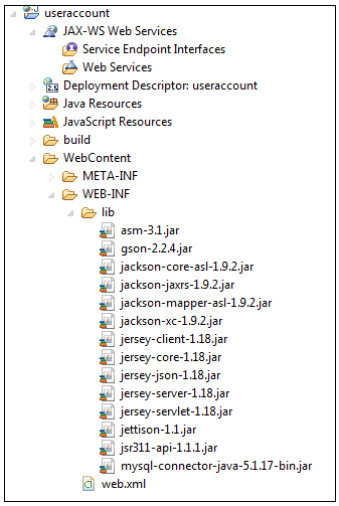


Enter project name as 'useraccount' and Click Finish





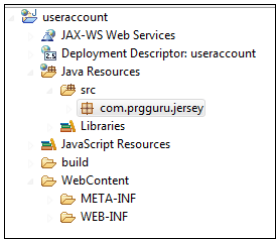
3. Add unzipped Jersey library JARs to WEB-INF/lib folder



4. Register Jersey as Servlet dispatcher by adding below code into web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java
  <display-name>useraccount</display-name>
  <servlet>
    <servlet-name>Jersey REST Service</servlet-name>
    <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet
    <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>
```

5. Create a package called 'com.prgguru.jersey' under src folder



6. Create a class called 'Constants.java' under the package 'com.prgguru.jersey' and add below code to it:

Constants class holds application constants like DB class, DB username etc.,

Constants.java

```
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";
```

```
}
```

7. Create a class called 'DBConnection.java' under the package 'com.prgguru.jersey' and add below code to it:

DBConnection class performs DB related operations like Opening DB connection, Inserting records and Selecting records from Table.

DBConnection.java

```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,
        } 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 {
                dbConn = DBConnection.createConnection();
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            Statement stmt = dbConn.createStatement();
            String query = "SELECT * FROM user WHERE username = '" + uname
                    + "' AND password=" + "'" + pwd + "'";
            //System.out.println(query);
            ResultSet rs = stmt.executeQuery(query);
            while (rs.next()) {
                //System.out.println(rs.getString(1) + rs.getString(2) + rs.getS
                isUserAvailable = true;
            }
        } catch (SQLException sqle) {
            throw sqle;
        } catch (Exception e) {
            // TODO Auto-generated catch block
            if (dbConn != null) {
                dbConn.close();
            }
            throw e;
        } finally {
            if (dbConn != null) {
                dbConn.close();
            }
        }
        return isUserAvailable;
    }
    /**
     * Method to insert uname and pwd in DB
     *
     * @param name
     * @param uname
     * @param pwd
     * @return
     * @throws SQLException
     * @throws Exception
     */
    public static boolean insertUser(String name, String uname, String pwd) thro
        boolean insertStatus = false;
        Connection dbConn = null;
        try {
            try {
                dbConn = DBConnection.createConnection();
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            Statement stmt = dbConn.createStatement();
            String query = "INSERT into user(name, username, password) values('
                    + uname + "','" + pwd + "')";
            //System.out.println(query);
            int records = stmt.executeUpdate(query);
            //System.out.println(records);
```

```
            //When record is successfully inserted
            if (records > 0) {
                insertStatus = true;
            }
        } catch (SQLException sqle) {
            //sqle.printStackTrace();
            throw sqle;
        } catch (Exception e) {
            //e.printStackTrace();
            // TODO Auto-generated catch block
            if (dbConn != null) {
                dbConn.close();
            }
            throw e;
        } finally {
            if (dbConn != null) {
                dbConn.close();
            }
        }
        return insertStatus;
    }
}
```

8. Create a class called 'Utility.java' under the package 'com.prgguru.jersey' and add below code to it:

> Utility class has utility methods to perform Null check, contruct JSON etc.,

[pglinkadssmall1]

Utility.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();
    }

}
```

9. Create a class called Register.java under the package 'com.prgguru.jersey' and add below code to it.

> Register class is the REST resource for registering the Users. User details sent from Android application will be inserted into DB after performing necessary checks.

Register.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>/regi
    public String doLogin(@QueryParam("name") String name, @QueryParam("username"
        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
        }else if(retCode == 2){
            response = Utitlity.constructJSON("register",false, "Special Characte
        }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 re
                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;
    }

}
```

10. Create a class called 'Login.java' under the package 'com.prgugur.jersey' and add below code to it.

> Login class is the REST resource which authenticates the Users. It gets the User credentials sent from Android application through HTTP and authenticates whether the credential is valid or not.

Login.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>/logi
    public String doLogin(@QueryParam("username") String uname, @QueryParam("pass
        String response = "";
        if(checkCredentials(uname, pwd)){
            response = Utitlity.constructJSON("login",true);
        }else{
            response = Utitlity.constructJSON("login", false, "Incorrect Email or
        }
    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;
    }

}
```


11. Deploy the web application:Right click on the project 'useraccount' >> Run As >> Run on Server

Code Dissection

Here are the important annotations in Jersey.

Annotation	Description
@PATH(your_path)	Sets the path to base URL + /your_path. The base URL is based on your application name, the servlet and the URL pattern from the `web.xml` configuration file.
@POST	Indicates that the following method will answer to an HTTP POST request.
@GET	Indicates that the following method will answer to an HTTP GET request.
@PUT	Indicates that the following method will answer to an HTTP PUT request.
@DELETE	Indicates that the following method will answer to an HTTP DELETE request.
@Produces(MediaType.TEXT_PLAIN[, more-types])	@Produces defines which MIME type is delivered by a method annotated with @GET. In the example text ("text/plain") is produced. Other examples would be "application/xml" or "application/json".
@Consumes(type[, more-types])	@Consumes defines which MIME type is consumed by this method.
@PathParam	Used to inject values from the URL into a method parameter. This way you inject, for example, the ID of a resource into the method to get the correct object.

Login.java

This class has a method called 'doLogin' which is the REST resource, that accepts query parameters as parameters and produce JSON as the response. Query parameters are Username and Password that are used for Authenticating the Users.

URL path to the method 'dologin' is illustrated in the below image:

```
package com.prgguru.jersey;

import javax.ws.rs.GET;  http://localhost/<appln-folder-name>/login/dologin?username=hello@domain.com&password=welcome123
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
     */
```

Register.java

This class has a method called 'doregister' which is the REST resource, that accepts query parameters as parameters and produce JSON as the response.

URL path to the method 'doregister' is illustrated in the below image:

```
package com.prgguru.jersey;
```

```
http://localhost/<appln-folder-name>/register/doregister?name=samuel&username=sam@domain.com&
                                                                    password=welcome123
import java.sql.SQLException;
//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;

    }
```

[pglinkadssmall]

Install Chrome Advanced REST client extension for Testing

Chrome Advanced REST client extension provides an easy way to test the REST API. It provides lot of options like adding request headers, adding request parameters, changing HTTP method by hitting an url. Install Advanced REST client extension in chrome browser and once you installed it you can find it in chrome Apps or an icon at the top right corner.

Registration

URL for registering the User is http://192.168.2.4:9999/useraccount/register/doregister?name=Admin&username=admin@programmerguru.com&password=password. Make sure you changed the IP address to your LAN IP address.

Logging in

URL for logging in the User is http://192.168.2.4:9999/useraccount/register/doregister?username=admin@programmerguru.com&password=password. Make sure you changed the IP address to your LAN IP address.

Here is the video demo of Testing I performed using Chrome Restful client:

RESTful Webservice Testing

Download Source Code

Entire project is zipped and is available for download. Unzip the downloaded project and to import the project into eclipse, launch eclipse >> File >> Import.. >> Choose downloaded project(How to import android project in eclipse).

Download Source Code

*apk in Android is the installation file similar to exe in windows.

If you feel this article is helpful and interesting please spread a word about it to your friends and colleagues by sharing the article in Facebook or Twitter.

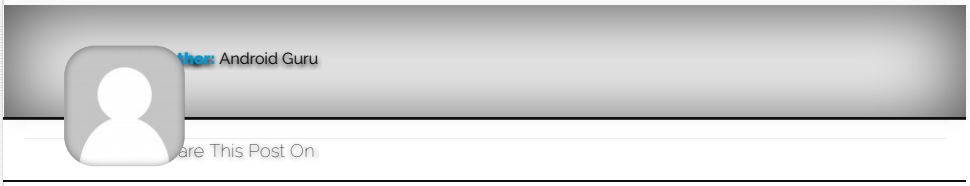
Tweet

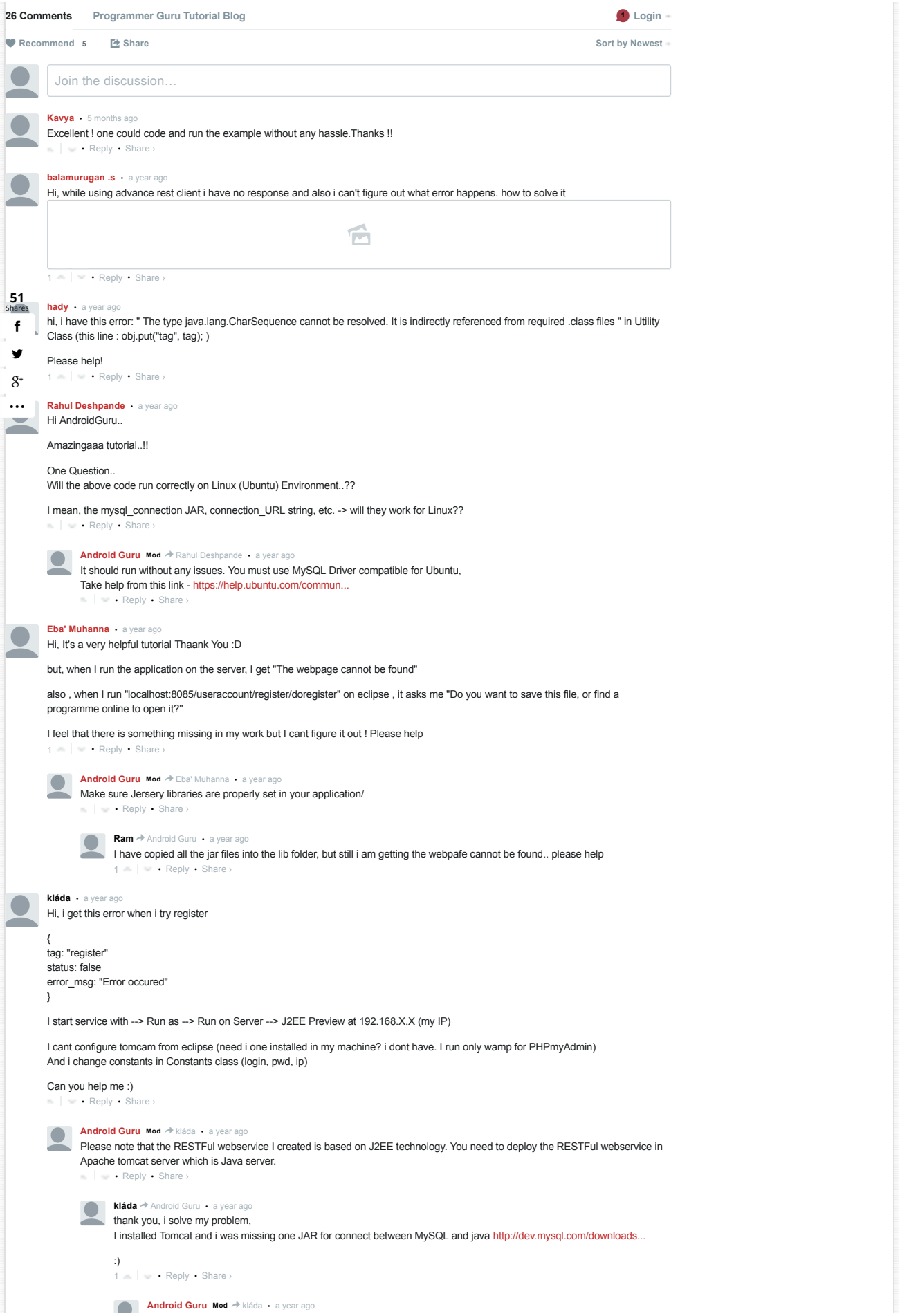
Share 42

You are always welcome to provide your comments and feedback from 26 Comments box.

[pgwriteforus]

[pgfeedback]





26 Comments Programmer Guru Tutorial Blog Login

Recommend 5 Share Sort by Newest

Join the discussion…

Kavya • 5 months ago
Excellent ! one could code and run the example without any hassle.Thanks !!
• Reply • Share ›

balamurugan .s • a year ago
Hi, while using advance rest client i have no response and also i can't figure out what error happens. how to solve it

1 • Reply • Share ›

51 Shares

hady • a year ago
hi, i have this error: " The type java.lang.CharSequence cannot be resolved. It is indirectly referenced from required .class files " in Utility Class (this line : obj.put("tag", tag);)

Please help!

1 • Reply • Share ›

Rahul Deshpande • a year ago
Hi AndroidGuru..

Amazingaaa tutorial..!!

One Question..
Will the above code run correctly on Linux (Ubuntu) Environment..??

I mean, the mysql_connection JAR, connection_URL string, etc. -> will they work for Linux??
• Reply • Share ›

Android Guru Mod ➚ Rahul Deshpande • a year ago
It should run without any issues. You must use MySQL Driver compatible for Ubuntu,
Take help from this link - https://help.ubuntu.com/commun...
• Reply • Share ›

Eba' Muhanna • a year ago
Hi, It's a very helpful tutorial Thaank You :D

but, when I run the application on the server, I get "The webpage cannot be found"

also , when I run "localhost:8085/useraccount/register/doregister" on eclipse , it asks me "Do you want to save this file, or find a programme online to open it?"

I feel that there is something missing in my work but I cant figure it out ! Please help
1 • Reply • Share ›

Android Guru Mod ➚ Eba' Muhanna • a year ago
Make sure Jersery libraries are properly set in your application/
• Reply • Share ›

Ram ➚ Android Guru • a year ago
I have copied all the jar files into the lib folder, but still i am getting the webpafe cannot be found.. please help
1 • Reply • Share ›

kláda • a year ago
Hi, i get this error when i try register

{
tag: "register"
status: false
error_msg: "Error occured"
}

I start service with --> Run as --> Run on Server --> J2EE Preview at 192.168.X.X (my IP)

I cant configure tomcam from eclipse (need i one installed in my machine? i dont have. I run only wamp for PHPmyAdmin)
And i change constants in Constants class (login, pwd, ip)

Can you help me :)
• Reply • Share ›

Android Guru Mod ➚ kláda • a year ago
Please note that the RESTFul webservice I created is based on J2EE technology. You need to deploy the RESTFul webservice in Apache tomcat server which is Java server.
• Reply • Share ›

kláda ➚ Android Guru • a year ago
thank you, i solve my problem,
I installed Tomcat and i was missing one JAR for connect between MySQL and java http://dev.mysql.com/downloads...

:)
1 • Reply • Share ›

Android Guru Mod ➚ kláda • a year ago

Search this website

powered by Google

Categories

Categories

Select Category

51 Shares

Follow Us

43.1k Follows

Facebook	Google+
18.1k Followers	1.2k Followers
Twitter	**Pinterest**
339 Followers	44 Followers
RSS	**YouTube**
23k Followers	344 Followers

Recent Popular Random

6 Most Fascinating Emulators for Android Devices
Jun 5, 2015

Android GUI Tools and Kits you can't afford to miss as an app developer
Jan 15, 2015

How to Position Toast in Android Application?
Jan 13, 2015

How to Receive Simple Data from Other Apps?
Jan 4, 2015

Tags

Action Bar Activity AJAX android apps development Android Basics android development company androidversions Android Wear Application AQuery AsyncTask Authentication Broadcast receivers Button Call Camera Dot Net Web Service Eclipse Email Facebook Gallery GCM Image Image_Upload Intent Java Web Service jQuery jQuery Mobile JSON Layout Library Localization Media Notifications php Restful webservice SD Card Seek bar SQLite Text to speech Vibrate Video web Web service Webview

Recent Posts

- 6 Most Fascinating Emulators for Android Devices June 5, 2015
- Android GUI Tools and Kits you can't afford to miss as an app developer January 15, 2015
- How to Position Toast in Android Application? January 13, 2015

Android Color Generator

About Me

Welcome to ProgrammerGuruTutorial Blog. Let me introduce myself more formally.
I'm a normal guy, engineer by education who is passionate about Programming and Internet.
Here I am writing lot about jQuery, AJAX, Android and JSON. Want to join with me in the journey of learning aforementioned technology? You are in the right place then :)
Keep me posted with your valuable feedback and comments.
Happy learning

Applications AQuery Basics Facebook GCM JSON Layout Localization Media SQLite Toast Web Service