Programmer Guru

Home Basics » Write For Us Advertise With Us Demo Videos Other Tutorials » About Us Contact Us

Home Web Service Android Restful Webservice Tutorial – How to call RESTful webservice in Android – Part 3

Android Restful Webservice Tutorial – How to call RESTful webservice in Android – Part 3

Posted By Android Guru on May 27, 2014 | 32 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 invoke RESTful webservice (that has been created and deployed in local application server) from inside Android applications.

I already discussed about What is RESTful webservice and how to create it in couple of posts earlier. Have listed those posts links below, it would be helpful for you if you quickly take a look at those articles.

> 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?

In this post, I will be discussing about invoking **RESTful webservice** we already created in the previous post (Read: How to create RESTful webservice in Java?).

If you haven't created the RESTful webservice yet, please go ahead and create it before proceeding with below steps.

Video Demo

Android RESTful Webservice Demo

Steps involved in invoking RESTful webservice in Android application:

By looking at the video demo, you got to know what are we going to develop.

About application

- It is a simple application to register and login the users.
- During registration and logging-in, user credentials (Username and Password) are taken to Server using RESTful webservice to register or authenticate the User.

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

Step 1: Create Android Application Project

Create new android project [File >> New >> Android Application Project] with project name AndroidRestFulWSExample

Enter package name as 'com.prgguru.example'

Choose Minimum Required SDK, Target SDK and Compile with. Confused on choosing these options? Take a look at Minimum Required SDK – Target SDK – Compile With post.

Click Next button and finally click 'Finish' to create project

Step 2: Add library to project

Add below third party library into project's 'lib' folder. You can also download it if you don't have it with you.

Android Asynchronous Http Client – An asynchronous callback-based Http client for Android built on top of Apache's HttpClient libraries which is used by Pinterest, Instagram etc.,. Download

Step 3: Design Screens

We are going to have three activities:

HomeActivity – Home Screen of the Application. Launched once after successful login.

LoginActivity – Login screen of the application

RegisterActivity – Register screen of the application

Before creating screens, make sure below String resources are added. Add string resources to strings.xml present under /res/values folder.

strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">AndroidRestFulWSExample</string>
    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>
    <string name="title_activity_login">Login</string>
    <string name="title_activity_home">HomeActivity</string>
    <string name="register_title">Register</string>
    <string name="login_title">Login</string>
    <string name="name">Name</string>
    <string name="email">Email</string>
    <string name="pwd">Password</string>
<string name="btnRegister">Register</string>
</resources>
```

Create three layout XMLs under /res/layout folder:

home.xml

Home Screen of the application that will be launched once after successful login by User.

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.prgguru.example.HomeActivity"
    tools:ignore="MergeRootFrame" >

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_gravity="fill_vertical"
        android:layout_marginTop="20pt"
        android:gravity="center_horizontal"
        android:text="Welcome User"
        android:textSize="25dip" />

</FrameLayout>
```

login.xml

Login screen of the application.

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
```

```
    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:padding="10dip" >

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginBottom="10dip"
            android:text="@string/login_title"
            android:textSize="25sp"
            android:textStyle="bold" />

        <EditText
            android:id="@+id/loginEmail"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:hint="Enter your Email ID"
            android:inputType="textEmailAddress" />

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="15dip"
            android:text="@string/pwd" />

        <EditText
            android:id="@+id/loginPassword"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:hint="Enter Password"
            android:inputType="textPassword" />

        <TextView
            android:id="@+id/login_error"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:padding="10dip"
            android:textColor="#e30000"
            android:textStyle="bold" />

        <Button
            android:id="@+id/btnLogin"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="20dip"
            android:background="#ff6347"
            android:onClick="loginUser"
            android:text="Login"
            android:textColor="#fff" />

        <Button
            android:id="@+id/btnLinkToRegisterScreen"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="40dip"
            android:background="@null"
            android:onClick="navigatetoRegisterActivity"
            android:text="Signup"
            android:textColor="#228b22"
            android:textStyle="bold" />
    </LinearLayout>

</ScrollView>
```

register.xml

Registration screen of the application.

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:padding="5dip" >

        <!-- View Title Label -->

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginBottom="10dip"
            android:text="@string/register_title"
            android:textSize="25sp"
            android:textStyle="bold" />
        <!-- Name Label -->

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="@string/name" />
        <!-- Name TextField -->

        <EditText
            android:id="@+id/registerName"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content" android:hint="Enter your Name",

        <!-- Email Label -->

```

```
        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="@string/email" />
        <!-- Email TextField -->

        <EditText
            android:id="@+id/registerEmail"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:hint="Enter your Email ID"
            android:inputType="textEmailAddress" />

        <!-- Password Label -->

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="15dip"
            android:text="@string/pwd" />
        <!-- Password TextField -->

        <EditText
            android:id="@+id/registerPassword"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:hint="Enter Password"
            android:inputType="textPassword" />

        <!-- Error message -->

        <TextView
            android:id="@+id/register_error"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:padding="10dip"
            android:textColor="#e30000"
            android:textStyle="bold" />

        <Button
            android:id="@+id/btnRegister"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="20dip"
            android:background="#ff6347"
            android:onClick="registerUser"
            android:text="@string/btnRegister"
            android:textColor="#fff" />

        <!-- Link to Login Screen -->

        <Button
            android:id="@+id/btnLinkToLoginScreen"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="20dip"
            android:background="@null"
            android:onClick="navigatetoLoginActivity"
            android:text="Login"
            android:textColor="#228b22"
            android:textStyle="bold" />
    </LinearLayout>

</ScrollView>
```

Step 4: Utility.java – Utitlity Class

We are done with Layout designing, let us jump into Coding.

Create Utility.Java under the package com.prgguru.example and fill it with below code. It has Utility methods to be used in other Activity classes.

```
package com.prgguru.example;

import java.util.regex.Matcher;
import java.util.regex.Pattern;
/**
 * Class which has Utility methods
 *
 */
public class Utility {
    private static Pattern pattern;
    private static Matcher matcher;
    //Email Pattern
    private static final String EMAIL_PATTERN =
            "^[_A-Za-z0-9-\+]+(\.[_A-Za-z0-9-]+)*@"
            + "[A-Za-z0-9-]+(\.[A-Za-z0-9]+)*(\.[A-Za-z]{2,})$";

    /**
     * Validate Email with regular expression
     *
     * @param email
     * @return true for Valid Email and false for Invalid Email
     */
    public static boolean validate(String email) {
        pattern = Pattern.compile(EMAIL_PATTERN);
        matcher = pattern.matcher(email);
        return matcher.matches();

    }
    /**
     * Checks for Null String object
     *
     * @param txt
```

```
     * @return true for not null and false for null String object
     */
    public static boolean isNotNull(String txt){
        return txt!=null && txt.trim().length()>0 ? true: false;
    }
}
```

Step 5: Home.java – Home Activity Class

Home screen Activity which will be launched once after successful login by User.

```
package com.prgguru.example;

import android.app.Activity;
import android.os.Bundle;
/**
 *
 * Home Screen Activity
 */
public class HomeActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //Displays Home Screen
        setContentView(R.layout.home);
    }

}
```

Step 6: LoginActivity.java – Login Screen Activity Class

Login screen which will allow User to login to Application. Make sure the IP address has been changed in the AsyncHttp Get method with localhost or IP address of your machine.

LoginActivity.java is well commented, so by reading the comments you will understand the meaning of it. In case, if you still have question 32 Comments it right away.

```
package com.prgguru.example;

import org.json.JSONException;
import org.json.JSONObject;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.loopj.android.http.AsyncHttpClient;
import com.loopj.android.http.AsyncHttpResponseHandler;
import com.loopj.android.http.RequestParams;
/**
 *
 * Login Activity Class
 *
 */
public class LoginActivity extends Activity {
    // Progress Dialog Object
    ProgressDialog prgDialog;
    // Error Msg TextView Object
    TextView errorMsg;
    // Email Edit View Object
    EditText emailET;
    // Passwprd Edit View Object
    EditText pwdET;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.login);
        // Find Error Msg Text View control by ID
        errorMsg = (TextView)findViewById(R.id.login_error);
        // Find Email Edit View control by ID
        emailET = (EditText)findViewById(R.id.loginEmail);
        // Find Password Edit View control by ID
        pwdET = (EditText)findViewById(R.id.loginPassword);
        // Instantiate Progress Dialog object
        prgDialog = new ProgressDialog(this);
        // Set Progress Dialog Text
        prgDialog.setMessage("Please wait...");
        // Set Cancelable as False
        prgDialog.setCancelable(false);
    }

    /**
     * Method gets triggered when Login button is clicked
     *
     * @param view
     */
    public void loginUser(View view){
        // Get Email Edit View Value
        String email = emailET.getText().toString();
        // Get Password Edit View Value
        String password = pwdET.getText().toString();
        // Instantiate Http Request Param Object
        RequestParams params = new RequestParams();
        // When Email Edit View and Password Edit View have values other than Nu
        if(Utility.isNotNull(email) && Utility.isNotNull(password)){
            // When Email entered is Valid
            if(Utility.validate(email)){
                // Put Http parameter username with value of Email Edit View cor
```

```
                params.put("username", email);
                // Put Http parameter password with value of Password Edit Value
                params.put("password", password);
                // Invoke RESTful Web Service with Http parameters
                invokeWS(params);
            }
            // When Email is invalid
            else{
                Toast.makeText(getApplicationContext(), "Please enter valid emai
            }
        } else{
            Toast.makeText(getApplicationContext(), "Please fill the form, don't
        }

    }

    /**
     * Method that performs RESTful webservice invocations
     *
     * @param params
     */
    public void invokeWS(RequestParams params){
        // Show Progress Dialog
        prgDialog.show();
        // Make RESTful webservice call using AsyncHttpClient object
        AsyncHttpClient client = new AsyncHttpClient();
        client.get("http://192.168.2.2:9999/useraccount/login/dologin",params ,
            // When the response returned by REST has Http response code '200'
            @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 assi
                        if(obj.getBoolean("status")){
                            Toast.makeText(getApplicationContext(), "You are su
                            // Navigate to Home screen
                            navigatetoHomeActivity();
                        }
                        // Else display error message
                        else{
                            errorMsg.setText(obj.getString("error_msg"));
                            Toast.makeText(getApplicationContext(), obj.getStri
                        }
                } catch (JSONException e) {
                    // TODO Auto-generated catch block
                    Toast.makeText(getApplicationContext(), "Error Occured [Ser
                    e.printStackTrace();

                }
            }
            // When the response returned by REST has Http response code other
            @Override
            public void onFailure(int statusCode, Throwable error,
                String content) {
                // Hide Progress Dialog
                prgDialog.hide();
                // When Http response code is '404'
                if(statusCode == 404){
                    Toast.makeText(getApplicationContext(), "Requested resource
                }
                // When Http response code is '500'
                else if(statusCode == 500){
                    Toast.makeText(getApplicationContext(), "Something went wrc
                }
                // When Http response code other than 404, 500
                else{
                    Toast.makeText(getApplicationContext(), "Unexpected Error c
                }
            }
         });
    }

    /**
     * Method which navigates from Login Activity to Home Activity
     */
    public void navigatetoHomeActivity(){
        Intent homeIntent = new Intent(getApplicationContext(),HomeActivity.clas
        homeIntent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(homeIntent);
    }

    /**
     * Method gets triggered when Register button is clicked
     *
     * @param view
     */
    public void navigatetoRegisterActivity(View view){
        Intent loginIntent = new Intent(getApplicationContext(),RegisterActivity
        loginIntent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(loginIntent);
    }

}
```

Step 7: RegisterActivity.java – RegisterScreen Activity Class

Register screen which will allow User to register him/herself to the Application. Make sure the IP address has been changed in the AsyncHttp Get method with localhost or IP address of your machine:

RegisterActivity.java is well commented, so by reading the comments you will understand the meaning of it. In case, if you still have question 32 Comments it right away.

```
package com.prgguru.example;

import org.json.JSONException;
import org.json.JSONObject;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.loopj.android.http.AsyncHttpClient;
import com.loopj.android.http.AsyncHttpResponseHandler;
import com.loopj.android.http.RequestParams;
/**
 *
 * Register Activity Class
 */
public class RegisterActivity extends Activity {
    // Progress Dialog Object
    ProgressDialog prgDialog;
    // Error Msg TextView Object
    TextView errorMsg;
    // Name Edit View Object
    EditText nameET;
    // Email Edit View Object
    EditText emailET;
    // Passwprd Edit View Object
    EditText pwdET;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.register);
        // Find Error Msg Text View control by ID
        errorMsg = (TextView)findViewById(R.id.register_error);
        // Find Name Edit View control by ID
        nameET = (EditText)findViewById(R.id.registerName);
        // Find Email Edit View control by ID
        emailET = (EditText)findViewById(R.id.registerEmail);
        // Find Password Edit View control by ID
        pwdET = (EditText)findViewById(R.id.registerPassword);
        // Instantiate Progress Dialog object
        prgDialog = new ProgressDialog(this);
        // Set Progress Dialog Text
        prgDialog.setMessage("Please wait...");
        // Set Cancelable as False
        prgDialog.setCancelable(false);
    }

    /**
     * Method gets triggered when Register button is clicked
     *
     * @param view
     */
    public void registerUser(View view){
        // Get NAme ET control value
        String name = nameET.getText().toString();
        // Get Email ET control value
        String email = emailET.getText().toString();
        // Get Password ET control value
        String password = pwdET.getText().toString();
        // Instantiate Http Request Param Object
        RequestParams params = new RequestParams();
        // When Name Edit View, Email Edit View and Password Edit View have valu
        if(Utility.isNotNull(name) && Utility.isNotNull(email) && Utility.isNotN
            // When Email entered is Valid
            if(Utility.validate(email)){
                // Put Http parameter name with value of Name Edit View control
                params.put("name", name);
                // Put Http parameter username with value of Email Edit View cor
                params.put("username", email);
                // Put Http parameter password with value of Password Edit View
                params.put("password", password);
                // Invoke RESTful Web Service with Http parameters
                invokeWS(params);
            }
            // When Email is invalid
            else{
                Toast.makeText(getApplicationContext(), "Please enter valid emai
            }
        }
        // When any of the Edit View control left blank
        else{
            Toast.makeText(getApplicationContext(), "Please fill the form, don't
        }

    }

    /**
     * Method that performs RESTful webservice invocations
     *
     * @param params
     */
    public void invokeWS(RequestParams params){
        // Show Progress Dialog
        prgDialog.show();
        // Make RESTful webservice call using AsyncHttpClient object
        AsyncHttpClient client = new AsyncHttpClient();
        client.get("http://192.168.2.2:9999/useraccount/register/doregister",par
            // When the response returned by REST has Http response code '200'
             @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 assi
                            if(obj.getBoolean("status")){
                                // Set Default Values for Edit View controls
                                setDefaultValues();
                                // Display successfully registered message using To
                                Toast.makeText(getApplicationContext(), "You are su
                            }
                            // Else display error message
                            else{
                                errorMsg.setText(obj.getString("error_msg"));
                                Toast.makeText(getApplicationContext(), obj.getStri
                            }
                    } catch (JSONException e) {
                        // TODO Auto-generated catch block
                        Toast.makeText(getApplicationContext(), "Error Occured [Ser
                        e.printStackTrace();

                    }
                }
                // When the response returned by REST has Http response code other
                @Override
                public void onFailure(int statusCode, Throwable error,
                    String content) {
                    // Hide Progress Dialog
                    prgDialog.hide();
                    // When Http response code is '404'
                    if(statusCode == 404){
                        Toast.makeText(getApplicationContext(), "Requested resource
                    }
                    // When Http response code is '500'
                    else if(statusCode == 500){
                        Toast.makeText(getApplicationContext(), "Something went wro
                    }
                    // When Http response code other than 404, 500
                    else{
                        Toast.makeText(getApplicationContext(), "Unexpected Error o
                    }
                }
            });
    }

    /**
     * Method which navigates from Register Activity to Login Activity
     */
    public void navigatetoLoginActivity(View view){
        Intent loginIntent = new Intent(getApplicationContext(),LoginActivity.cl
        // Clears History of Activity
        loginIntent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(loginIntent);
    }

    /**
     * Set degault values for Edit View controls
     */
    public void setDefaultValues(){
        nameET.setText("");
        emailET.setText("");
        pwdET.setText("");
    }

}
```

Step 8: Add Activities in AndroidManifest.xml

Make sure you added activities we created in AndroidManifest.xml.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.prgguru.example"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="19" />

    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.prgguru.example.RegisterActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name="com.prgguru.example.LoginActivity"
            android:label="@string/title_activity_login" >
        </activity>
        <activity
            android:name="com.prgguru.example.HomeActivity"
            android:label="@string/title_activity_home" >
        </activity>
    </application>

</manifest>
```

Step 9: Add Permission in AndroidManifest.xml

Don't forget to add internet permission in AndroidManifest.xml:

```
<!-- Permission: Allow application to connect to Internet -->
 <uses-permission android:name="android.permission.INTERNET" />
```

Error Handling

When the Http response code returned by REST web service is other 200, which means an error might have occurred at server end. Error can be easily handled inside onFailure() method of AsyncHttpClient's post method as shown below:


```
@Override
public void onFailure(int statusCode, Throwable error,
        String content) {
        // Hide Progress Dialog
        prgDialog.hide();
        // When Http response code is '404'
        if(statusCode == 404){
            Toast.makeText(getApplicationContext(), "Requested resource not four
        }
        // When Http response code is '500'
        else if(statusCode == 500){
            Toast.makeText(getApplicationContext(), "Something went wrong at ser
        }
        // When Http response code other than 404, 500
        else{
            Toast.makeText(getApplicationContext(), "Unexpected Error occcured!
        }
 }
```

RESTful Webservice Response – JSON

Login

When login is successful, the JSON returned by REST will look like:

```
{
tag: "login"
status: true
}
```

When login is unsuccessful (User entered incorrect Username or Password), the JSON returned by REST will look like:

```
{
tag: "login"
status: false
error_msg: "Incorrect Email or Password"
}
```

Registration

When registration is successful, the JSON returned by REST will look like:

```
{
tag: "register"
status: true
}
```

When registration is unsuccessful (User already registered), the JSON returned by REST will look like:

```
{
tag: "register"
status: false
error_msg: "You are already registered"
}
```

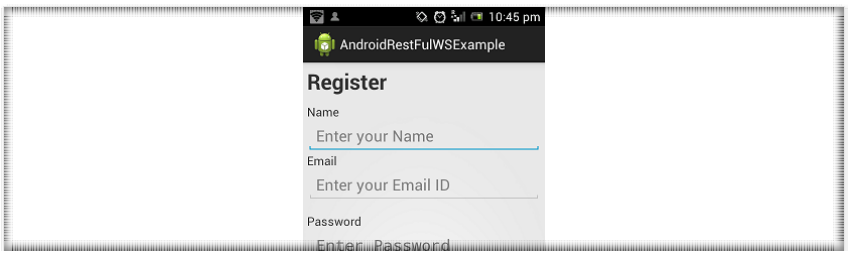
[pglinkadssmall1]

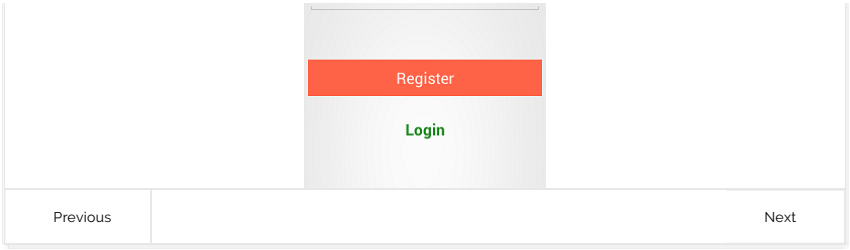
Demo

> Make sure the RESTful webservice you created is up and running.

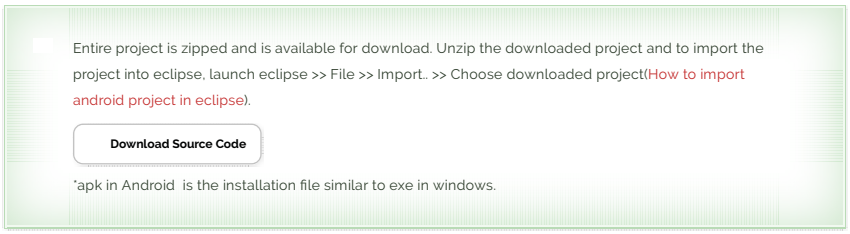
That's all. It's time to test our code.

Run the application using emulator or device by right clicking on the project >> Run as >> Android applicaiton >> Choose emulator or device.





Download Source Code



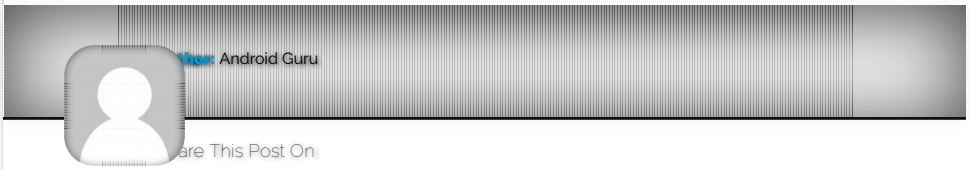
[pglinkadssmall]

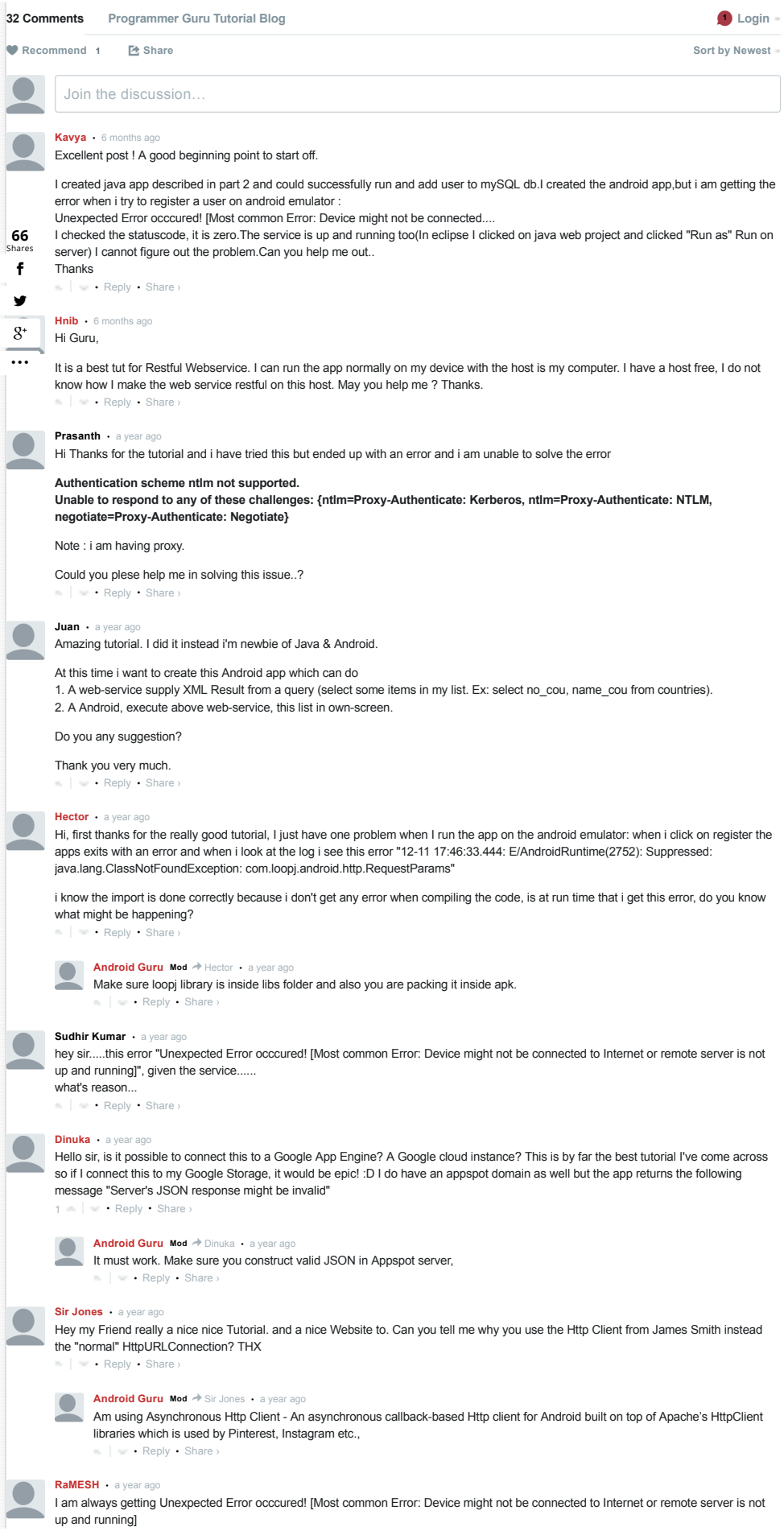
If you feel this article is helpful and interesting please spread a word about it to your friends and colleagues by sharing the article in Facebook or Twitter.

You are always welcome to provide your comments and feedback from 32 Comments box.

[pgwriteforus]

[pgfeedback]





32 Comments Programmer Guru Tutorial Blog Login

Recommend 1 Share Sort by Newest

Join the discussion…

Kavya • 6 months ago

Excellent post ! A good beginning point to start off.

I created java app described in part 2 and could successfully run and add user to mySQL db.I created the android app,but i am getting the error when i try to register a user on android emulator :
Unexpected Error occcured! [Most common Error: Device might not be connected....
I checked the statuscode, it is zero.The service is up and running too(In eclipse I clicked on java web project and clicked "Run as" Run on server) I cannot figure out the problem.Can you help me out..
Thanks

1 • Reply • Share ›

66 Shares

Hnib • 6 months ago

Hi Guru,

It is a best tut for Restful Webservice. I can run the app normally on my device with the host is my computer. I have a host free, I do not know how I make the web service restful on this host. May you help me ? Thanks.

• Reply • Share ›

Prasanth • a year ago

Hi Thanks for the tutorial and i have tried this but ended up with an error and i am unable to solve the error

Authentication scheme ntlm not supported.
Unable to respond to any of these challenges: {ntlm=Proxy-Authenticate: Kerberos, ntlm=Proxy-Authenticate: NTLM, negotiate=Proxy-Authenticate: Negotiate}

Note : i am having proxy.

Could you plese help me in solving this issue..?

• Reply • Share ›

Juan • a year ago

Amazing tutorial. I did it instead i'm newbie of Java & Android.

At this time i want to create this Android app which can do
1. A web-service supply XML Result from a query (select some items in my list. Ex: select no_cou, name_cou from countries).
2. A Android, execute above web-service, this list in own-screen.

Do you any suggestion?

Thank you very much.

• Reply • Share ›

Hector • a year ago

Hi, first thanks for the really good tutorial, I just have one problem when I run the app on the android emulator: when i click on register the apps exits with an error and when i look at the log i see this error "12-11 17:46:33.444: E/AndroidRuntime(2752): Suppressed: java.lang.ClassNotFoundException: com.loopj.android.http.RequestParams"

i know the import is done correctly because i don't get any error when compiling the code, is at run time that i get this error, do you know what might be happening?

• Reply • Share ›

Android Guru Mod ➚ Hector • a year ago

Make sure loopj library is inside libs folder and also you are packing it inside apk.

• Reply • Share ›

Sudhir Kumar • a year ago

hey sir.....this error "Unexpected Error occcured! [Most common Error: Device might not be connected to Internet or remote server is not up and running]", given the service.....
what's reason...

• Reply • Share ›

Dinuka • a year ago

Hello sir, is it possible to connect this to a Google App Engine? A Google cloud instance? This is by far the best tutorial I've come across so if I connect this to my Google Storage, it would be epic! :D I do have an appspot domain as well but the app returns the following message "Server's JSON response might be invalid"

1 • Reply • Share ›

Android Guru Mod ➚ Dinuka • a year ago

It must work. Make sure you construct valid JSON in Appspot server,

• Reply • Share ›

Sir Jones • a year ago

Hey my Friend really a nice nice Tutorial. and a nice Website to. Can you tell me why you use the Http Client from James Smith instead the "normal" HttpURLConnection? THX

• Reply • Share ›

Android Guru Mod ➚ Sir Jones • a year ago

Am using Asynchronous Http Client - An asynchronous callback-based Http client for Android built on top of Apache's HttpClient libraries which is used by Pinterest, Instagram etc.,

• Reply • Share ›

RaMESH • a year ago

I am always getting Unexpected Error occcured! [Most common Error: Device might not be connected to Internet or remote server is not up and running]

Search this website

powered by Google

Categories

66 Shares

Categories

Select Category

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

SnackBar Vs Toast
Feb 26, 2016

Introduction to Android Fragments
Feb 25, 2016

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

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

- SnackBar Vs Toast February 26, 2016
- Introduction to Android Fragments February 25, 2016
- 6 Most Fascinating Emulators for Android Devices June 5, 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

Designed by **Elegant Themes** | Powered by **WordPress**