手机后台服务：由于采用Struts2的JSON响应格式，响应详细会自动转变为JSON格式，故直接输出即可。

package com.easyway.json.android;

import java.util.HashMap;

import java.util.Map;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.struts2.interceptor.ServletRequestAware;

import org.apache.struts2.interceptor.ServletResponseAware;

import com.opensymphony.xwork2.ActionSupport;

/\*\*

\* 在android中有时候我们不需要用到本机的SQLite数据库提供数据，更多的时候是从网络上获取数据，

\* 那么Android怎么从服务器端获取数据呢？有很多种，归纳起来有

\*一：基于Http协议获取数据方法。

\*二：基于SAOP协议获取数据方法，

\*那么我们的这篇文章主要是将关于使用Http协议获取服务器端数据，

\*这里我们采取的服务器端技术为java，框架为Struts2,或者可以有Servlet,又或者可直接从JSP页面中获取数据。

\*那么，接下来我们便开始这一路程：

\*首先：编写服务器端方法,我这里采用的MVC框架是Struts2，目的很单纯，就是为了以后做个完整的商业项目，

\*技术配备为：android+SSH。当然，篇幅有限，我这里就直接用Strtus2而已。

\*服务器端：新建WebProject ,选择Java ee 5.0.

\*为了给项目添加Struts2的支持，我们必须导入Struts2的一些类库，如下即可（有些jar包是不必的，但是我们后来扩展可能是要使用到的，就先弄进去）：

\*xwork-core-2.2.1.1.jar struts2-core-2.2.1.1.jar commons-logging-1.0.4.jar freemarker-2.3.16.jar

\*ognl-3.0.jar javassist-3.7.ga.jar commons-ileupload.jar commons-io.jar json-lib-2.1-jdk15.jar

\*处理JSON格式数据要使用到 struts2-json-plugin-2.2.1.1.jar

\* 基于struts2的json插件.

\*

\*

\* 采用接口注入的方式注入HttpServletRequest,HttpServletResponse对象

\*

\* @author longgangbai

\*

\*/

public class LoginAction extends ActionSupport implements ServletRequestAware,

ServletResponseAware {

/\*\* \* \*/

private static final long serialVersionUID = 1L;

HttpServletRequest request;

HttpServletResponse response;

private String userName;

private String password;

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public void setServletRequest(HttpServletRequest request) {

this.request = request;

}

public void setServletResponse(HttpServletResponse response) {

this.response = response;

}

/\*\*

\* 模拟用户登录的业务

\*/

public void login() {

try {

//如果不采用接口注入的方式的获取HttpServletRequest，HttpServletResponse的方式

// HttpServletRequest request =ServletActionContext.getRequest();

// HttpServletResponse response=ServletActionContext.getResponse();

this.response.setContentType("text/json;charset=utf-8");

this.response.setCharacterEncoding("UTF-8");

//JSONObject json=new JSONObject();

Map<String,String> json=new HashMap<String,String>();

if ("admin".equals(userName)&&"123456".equals(password)) {

json.put("message", "欢迎管理员登陆");

} else if ((!"admin".equals(userName))&&"123456".equals(password)) {

json.put("message", "欢迎"+userName+"登陆！");

} else {

json.put("message", "非法登陆信息！");

}

byte[] jsonBytes = json.toString().getBytes("utf-8");

response.setContentLength(jsonBytes.length);

response.getOutputStream().write(jsonBytes);

response.getOutputStream().flush();

response.getOutputStream().close();

} catch (Exception e) {

e.printStackTrace();

}

}

}

struts.xml

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE struts PUBLIC "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN" "http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>

<!-- setting encoding,DynamicMethod,language

<constant name="struts.custom.i18n.resources" value="messageResource"></constant> -->

<constant name="struts.i18n.encoding" value="UTF-8"/>

<constant name="struts.enable.DynamicMethodInvocation" value="true"/>

<!-- add package here extends="struts-default"-->

<package name="default" extends="json-default"><!--需要将struts-default改为-->

<action name="login" class="com.easyway.json.android.LoginAction"

method="login">

<result type="json"/>

<!--返回值类型设置为json,不设置返回页面-->

</action>

</package>

</struts>

 web.xml

<?xml version="1.0" encoding="UTF-8"?>

<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd">

<!-- 定义Struts2的核心控制器：FilterDispatcher -->

<filter>

<!-- 定义核心Filter的名称 -->

<filter-name>struts2</filter-name>

<!-- 定义Filter的实现类 -->

<filter-class>

org.apache.struts2.dispatcher.FilterDispatcher

</filter-class>

</filter>

<filter-mapping>

<filter-name>struts2</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

<welcome-file-list>

<welcome-file>index.jsp</welcome-file>

</welcome-file-list>

</web-app>

jsp测试页面：

<%@ page language="java" import="java.util.\*" pageEncoding="UTF-8"%>

<%@ taglib uri="/struts-tags" prefix="s"%>

<%

String path = request.getContextPath();

String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";

%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Ajax调用web服务</title>

<script type="text/javascript">

var xmlHttpReq; //用于保存XMLHttpRequest对象的全局变量

//用于创建XMLHttpRequest对象

function createXmlHttp() {

//根据window.XMLHttpRequest对象是否存在使用不同的创建方式

// if (window.XMLHttpRequest) {

// xmlHttp = new XMLHttpRequest(); //FireFox、Opera等浏览器支持的创建方式

// } else {

// xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");//IE浏览器支持的创建方式

// }

if (window.ActiveXObject) {

xmlHttpReq = new ActiveXObject("Microsoft.XMLHTTP");

} else if (window.XMLHttpRequest) {

xmlHttpReq = new XMLHttpRequest();

}

}

function loadAjax() {

alert("-------1----------");

createXmlHttp(); //创建XmlHttpRequest对象

alert("-------2---------");

var url="http://localhost:8080/AndroidStruts2JSON/login.action?userName=admin&password=123456&date="+new Date();

xmlHttpReq.open("get", encodeURI(encodeURI(url+param,"UTF-8"),"UTF-8"), true);

//xmlHttpReq.open("get", encodeURI(encodeURI(url,"UTF-8"),"UTF-8"), true);//上传图片

xmlHttpReq.setrequestheader("content-type","application/x-www-form-urlencoded");//post提交设置项

xmlHttpReq.onreadystatechange = loadCallback; //IE这里设置回调函数

xmlHttpReq.send(null);

}

function loadCallback() {

alert("-------3---------");

//alert(xmlHttpReq.readyState);

if (xmlHttpReq.readyState == 4) {

alert("aa");

//if(xmlHttpReq.status==200){

document.getElementById("contentDiv").innerHTML=xmlHttpReq.responseText;

//}

}

}

</script>

<body>

<div id="contentTypeDiv">

</div>

<br/><br/>

<div id="contentDiv">

</div>

<input type="button" value="调用" onclick="loadAjax()">

</body>

</head>

</html>

手机前台服务：

package com.easyway.android.json;

import java.io.IOException;

import org.apache.http.HttpEntity;

import org.apache.http.HttpResponse;

import org.apache.http.client.ClientProtocolException;

import org.apache.http.client.HttpClient;

import org.apache.http.client.methods.HttpGet;

import org.apache.http.impl.client.DefaultHttpClient;

import org.apache.http.util.EntityUtils;

import org.json.JSONException;

import org.json.JSONObject;

import android.app.Activity;

import android.app.AlertDialog;

import android.app.AlertDialog.Builder;

import android.content.DialogInterface;

import android.os.Bundle;

import android.os.StrictMode;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

/\*\*

\*

\*

\* 在android中有时候我们不需要用到本机的SQLite数据库提供数据，更多的时候是从网络上获取数据，

\* 那么Android怎么从服务器端获取数据呢？有很多种，归纳起来有

\* 一：基于Http协议获取数据方法。

\* 二：基于SAOP协议获取数据方法

\*

\*备注：在网络有关的问题最好添加以下两项：

\* 1.线程和虚拟机策略

\* ///在Android2.2以后必须添加以下代码

\* //本应用采用的Android4.0

\* //设置线程的策略

\* StrictMode.setThreadPolicy(new StrictMode.ThreadPolicy.Builder()

\* .detectDiskReads()

\* .detectDiskWrites()

\* .detectNetwork() // or .detectAll() for all detectable problems

\* .penaltyLog()

\* .build());

\* //设置虚拟机的策略

\* StrictMode.setVmPolicy(new StrictMode.VmPolicy.Builder()

\* .detectLeakedSqlLiteObjects()

\* //.detectLeakedClosableObjects()

\* .penaltyLog()

\* .penaltyDeath()

\* .build());

\* 2.可以访问网络的权限：

\* 即AndroidManifest.xml中配置：

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

\*

\*

\* @author longgangbai

\*

\*

\*/

public class AndroidHttpJSONActivity extends Activity {

private static String processURL="http://192.168.134.1:8080/AndroidStruts2JSON/login.action?";

private EditText txUserName;

private EditText txPassword;

private Button btnLogin;

/\*\*

\* Called when the activity is first created.

\*/

@Override

public void onCreate(Bundle savedInstanceState) {

///在Android2.2以后必须添加以下代码

//本应用采用的Android4.0

//设置线程的策略

StrictMode.setThreadPolicy(new StrictMode.ThreadPolicy.Builder()

.detectDiskReads()

.detectDiskWrites()

.detectNetwork() // or .detectAll() for all detectable problems

.penaltyLog()

.build());

//设置虚拟机的策略

StrictMode.setVmPolicy(new StrictMode.VmPolicy.Builder()

.detectLeakedSqlLiteObjects()

//.detectLeakedClosableObjects()

.penaltyLog()

.penaltyDeath()

.build());

super.onCreate(savedInstanceState);

//设置页面布局

setContentView(R.layout.main);

//设置初始化视图

initView();

//设置事件监听器方法

setListener();

}

/\*\*

\* 创建初始化视图的方法

\*/

private void initView() {

btnLogin=(Button)findViewById(R.id.btnLogin);

txPassword=(EditText)findViewById(R.id.txtPassword);

txUserName=(EditText)findViewById(R.id.txtUserName);

}

/\*\*

\* 设置事件的监听器的方法

\*/

private void setListener() {

btnLogin.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

String userName=txUserName.getText().toString();

String password=txPassword.getText().toString();

loginRemoteService(userName,password);

}

});

}

/\*\*

\* 获取Struts2 Http 登录的请求信息

\* @param userName

\* @param password

\*/

public void loginRemoteService(String userName,String password){

String result=null;

try {

//创建一个HttpClient对象

HttpClient httpclient = new DefaultHttpClient();

//远程登录URL

processURL=processURL+"userName="+userName+"&password="+password;

Log.d("远程URL", processURL);

//创建HttpGet对象

HttpGet request=new HttpGet(processURL);

//请求信息类型MIME每种响应类型的输出（普通文本、html 和 XML，json）。允许的响应类型应当匹配资源类中生成的 MIME 类型

//资源类生成的 MIME 类型应当匹配一种可接受的 MIME 类型。如果生成的 MIME 类型和可接受的 MIME 类型不 匹配，那么将

//生成 com.sun.jersey.api.client.UniformInterfaceException。例如，将可接受的 MIME 类型设置为 text/xml，而将

//生成的 MIME 类型设置为 application/xml。将生成 UniformInterfaceException。

request.addHeader("Accept","text/json");

//获取响应的结果

HttpResponse response =httpclient.execute(request);

//获取HttpEntity

HttpEntity entity=response.getEntity();

//获取响应的结果信息

String json =EntityUtils.toString(entity,"UTF-8");

//JSON的解析过程

if(json!=null){

JSONObject jsonObject=new JSONObject(json);

result=jsonObject.get("message").toString();

}

if(result==null){

json="登录失败请重新登录";

}

//创建提示框提醒是否登录成功

AlertDialog.Builder builder=new Builder(AndroidHttpJSONActivity.this);

builder.setTitle("提示")

.setMessage(result)

.setPositiveButton("确定", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

dialog.dismiss();

}

}).create().show();

} catch (ClientProtocolException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (JSONException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}