**java使用struts2框架开发微信服务号**

2015年05月22日 12:59:47

阅读数：3946

一、前言

此文章仅代表个人的见解

二、微信接入

1、登录微信公众平台官网后，在公众平台后台管理页面*-*开发者中心页，点击“修改配置”按钮，填写服务器地址（*URL*）、*Token*和*EncodingAESKey*，其中*URL*是开发者用来接收微信消息和事件的接口*URL*。*Token*可由开发者可以任意填写，用作生成签名（该*Token*会和接口*URL*中包含的*Token*进行比对，从而验证安全性）。*EncodingAESKey*由开发者手动填写或随机生成，将用作消息体加解密密钥。

同时，开发者可选择消息加解密方式：明文模式、兼容模式和安全模式。模式的选择与服务器配置在提交后都会立即生效，请开发者谨慎填写及选择。加解密方式的默认状态为明文模式，选择兼容模式和安全模式需要提前配置好相关加解密代码，详情请参考消息体签名及加解密部分的文档

*2*、开发者提交信息后，微信服务器将发送*GET*请求到填写的服务器地址*URL*上，*GET*请求携带四个参数：*signature(*微信加密签名，*signature*结合了开发者填写的*token*参数和请求中的*timestamp*参数、*nonce*参数*),timestamp(*时间戳*),nonce*（随机数），*echostr*（随机字符串）。

开发者通过检验*signature*对请求进行校验（下面有校验方式）。若确认此次*GET*请求来自微信服务器，请原样返回*echostr*参数内容，则接入生效，成为开发者成功，否则接入失败。

**加密/校验流程如下：**

    1. 将token、timestamp、nonce三个参数进行**字典序**排序

    2. 将三个参数字符串拼接成一个字符串进行sha1加密

    3. 开发者获得加密后的字符串可与signature对比，标识该请求来源于微信

参考*api*地址[*http://mp.weixin.qq.com/wiki/17/2d4265491f12608cd170a95559800f2d.html*](http://mp.weixin.qq.com/wiki/17/2d4265491f12608cd170a95559800f2d.html)

*3*、根据*api*第三步：依据接口文档实现业务逻辑

验证*URL*有效性成功后即接入生效，成为开发者。如果公众号类型为服务号（订阅号只能使用普通消息接口），可以在公众平台网站中申请认证，认证成功的服务号将获得众多接口权限，以满足开发者需求。

**当你前面两部都搞定了的话，第三步就容易许多，基本根据*api*接口来就行了**

三、进入代码开发

1、在上述的*3*步中，最为开始的疑问想必就是第二步了

当你在修改配置跟填写你自定义的*token*后，你所提交的地址必须就已经存在，并且能够接收微信推送过去的数据，同时进行验证，通过检验*signature*对请求进行校验，若校验成功则原样返回*echostr*，否则连提交保存都无法成功。

那么具体流程是这样子的，当你提交的时候，微信就发送了*4*个字段到你填写的*url*上

下面是我接收这四个字段的代码，由于微信二次开发在网上对*struts*的框架介绍比较少，故此处就介绍*struts2*跟微信对接，而不使用*servlet*，*servlet*虽然是个好东西，笔者比较少用，其实*struts2*也是基于*servlet*的，所以基本都看得懂。简便写上接收代码，假设下面代码是填写url访问后的处理逻辑

  //打印接收的四个字段

 System.out.println("signature:" + request.getParameter("signature"));  
 System.out.println("timestamp:" + request.getParameter("timestamp"));  
 System.out.println("nonce:" + request.getParameter("nonce"));  
 System.out.println("echostr:" + request.getParameter("echostr"));

// 通过检验signature对请求进行校验，若校验成功则原样返回echostr，表示接入成功，否则接入失败

**if** (SignUtil.*checkSignature*(signature,timestamp,nonce)) {

         response.setContentType("text/html;charset=UTF-8");

         response.getWriter().write(echostr);

  }

 return null;

SignUtil工具类如下，主要改了token，与公众平台上填写的要一致

**[java]** [view plain](https://blog.csdn.net/aguang_vip/article/details/45915887) [copy](https://blog.csdn.net/aguang_vip/article/details/45915887)

1. **package** cn.com.aguang.weixin.utils;
3. **import** java.security.MessageDigest;
4. **import** java.security.NoSuchAlgorithmException;
6. /\*\*
7. \* 请求校验工具类
8. \*
9. \* @author aguang
10. \* @Email zhaoguiguang@yeah.net
11. \*
12. \*/
13. **public** **class** SignUtil {
14. // 与接口配置信息中的Token要一致
15. **private** **static** String token = "aguang";
17. /\*\*
18. \* 验证签名
19. \*
20. \* @param signature
21. \* @param timestamp
22. \* @param nonce
23. \* @return
24. \*/
25. **public** **static** **boolean** checkSignature(String signature, String timestamp,
26. String nonce) {
27. String[] arr = **new** String[] { token, timestamp, nonce };
28. // 将token、timestamp、nonce三个参数进行字典序排序
29. sort(arr);
30. StringBuilder content = **new** StringBuilder();
31. **for** (**int** i = 0; i < arr.length; i++) {
32. content.append(arr[i]);
33. }
34. MessageDigest md = **null**;
35. String tmpStr = **null**;
37. **try** {
38. md = MessageDigest.getInstance("SHA-1");
39. // 将三个参数字符串拼接成一个字符串进行sha1加密
40. **byte**[] digest = md.digest(content.toString().getBytes());
41. tmpStr = byteToStr(digest);
42. } **catch** (NoSuchAlgorithmException e) {
43. e.printStackTrace();
44. }
46. content = **null**;
47. // 将sha1加密后的字符串可与signature对比，标识该请求来源于微信
48. **return** tmpStr != **null** ? tmpStr.equals(signature.toUpperCase()) : **false**;
49. }
51. /\*\*
52. \* 将字节数组转换为十六进制字符串
53. \*
54. \* @param byteArray
55. \* @return
56. \*/
57. **private** **static** String byteToStr(**byte**[] byteArray) {
58. String strDigest = "";
59. **for** (**int** i = 0; i < byteArray.length; i++) {
60. strDigest += byteToHexStr(byteArray[i]);
61. }
62. **return** strDigest;
63. }
65. /\*\*
66. \* 将字节转换为十六进制字符串
67. \*
68. \* @param mByte
69. \* @return
70. \*/
71. **private** **static** String byteToHexStr(**byte** mByte) {
72. **char**[] Digit = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A',
73. 'B', 'C', 'D', 'E', 'F' };
74. **char**[] tempArr = **new** **char**[2];
75. tempArr[0] = Digit[(mByte >>> 4) & 0X0F];
76. tempArr[1] = Digit[mByte & 0X0F];
78. String s = **new** String(tempArr);
79. **return** s;
80. }
82. **public** **static** **void** sort(String a[]) {
83. **for** (**int** i = 0; i < a.length - 1; i++) {
84. **for** (**int** j = i + 1; j < a.length; j++) {
85. **if** (a[j].compareTo(a[i]) < 0) {
86. String temp = a[i];
87. a[i] = a[j];
88. a[j] = temp;
89. }
90. }
91. }
92. }
93. }

2、上述搭建后便可以进行提交了，如果保存成功，那么恭喜你，现在你可以进行第三步接口的开发了。

     废话我就不多说了直接上代码，大家可以根据api文档，还有我以下附上的几个封装好的处理工具，以及对接的代码和测试类代码，基本根据参考这几个，也可以进行微信接口开发了，我推荐大家先自己摸索一下，如果有什么地方卡住了可以下面的代码，如果有什么不规范的地方或者写得不好的地方，可以留言说一下，下面的例子只针对几个接口做开发而已，具体后面的功能再加上。不对的地方请指出

3、笔者填写的地址处理类（框架s2sh，大家可以根据各自不同情况进行修改）

**[java]** [view plain](https://blog.csdn.net/aguang_vip/article/details/45915887) [copy](https://blog.csdn.net/aguang_vip/article/details/45915887)

1. **package** cn.com.aguang.weixin.action.weixin;
3. **import** java.io.InputStream;
4. **import** java.net.URLEncoder;
5. **import** java.util.Date;
6. **import** java.util.List;
8. **import** javax.annotation.Resource;
9. **import** javax.servlet.http.HttpServletRequest;
10. **import** javax.servlet.http.HttpServletResponse;
12. **import** net.sf.json.JSONArray;
13. **import** net.sf.json.JSONObject;
15. **import** org.apache.struts2.ServletActionContext;
16. **import** org.dom4j.Document;
17. **import** org.dom4j.DocumentHelper;
18. **import** org.dom4j.Element;
19. **import** org.springframework.context.annotation.Scope;
20. **import** org.springframework.stereotype.Controller;
22. **import** cn.com.aguang.weixin.bussiness.WeixinOpenIdBus;
23. **import** cn.com.aguang.weixin.bussiness.WeixinUtilBus;
24. **import** cn.com.aguang.weixin.dao.WeixinOpenIdDAO;
25. **import** cn.com.aguang.weixin.po.AgWeixinOpenId;
26. **import** cn.com.aguang.weixin.utils.HttpUtils;
27. **import** cn.com.aguang.weixin.utils.SignUtil;
28. **import** cn.com.aguang.weixin.utils.WeiXinContants;
30. **import** com.opensymphony.xwork2.ActionSupport;
32. @Controller
33. @Scope("prototype")
34. **public** **class** AguangWeiXinSoftAction **extends** ActionSupport {
36. // 微信opendId bus 类
37. **private** WeixinOpenIdBus weixinOpenIdBus;
39. // 为了维护行更高，觉得抽取业务逻辑的工具类，将业务一块一块分开
40. **private** WeixinUtilBus weixinUtilBus;
42. @Resource
43. **public** **void** setWeixinUtilBus(WeixinUtilBus weixinUtilBus) {
44. **this**.weixinUtilBus = weixinUtilBus;
45. }
47. @Resource
48. **public** **void** setWeixinOpenIdBus(WeixinOpenIdBus weixinOpenIdBus) {
49. **this**.weixinOpenIdBus = weixinOpenIdBus;
50. }
52. // 微信对接参数
53. **private** String signature;
55. // 微信对接参数
56. **private** String timestamp;
58. // 微信对接参数
59. **private** String nonce;
61. // 微信对接参数
62. **private** String echostr;
64. **public** String getSignature() {
65. **return** signature;
66. }
68. **public** **void** setSignature(String signature) {
69. **this**.signature = signature;
70. }
72. **public** String getTimestamp() {
73. **return** timestamp;
74. }
76. **public** **void** setTimestamp(String timestamp) {
77. **this**.timestamp = timestamp;
78. }
80. **public** String getNonce() {
81. **return** nonce;
82. }
84. **public** **void** setNonce(String nonce) {
85. **this**.nonce = nonce;
86. }
88. **public** String getEchostr() {
89. **return** echostr;
90. }
92. **public** **void** setEchostr(String echostr) {
93. **this**.echostr = echostr;
94. }
96. /\*\*
97. \*
98. \*/
99. **private** **static** **final** **long** serialVersionUID = 3674789318078769142L;
101. /\*\*
102. \* <p>
103. \* 重大发现,经过api多个接口的测试研究，发现很多接口都通过这里来认证，并且对接
104. \* <p>
105. \*
106. \* 微信接入，token暗号对接，对应公众号微信平台管理开发填写的地址
107. \*/
108. @Override
109. **public** String execute() **throws** Exception {
110. HttpServletResponse response = ServletActionContext.getResponse();
112. // JSONArray array = new JSONArray();
114. HttpServletRequest request = ServletActionContext.getRequest();
116. // 验证消息的真实性接收的参数，即填写url时点击保存提交的时候，能接收到下面4个参数
117. // System.out.println("signature:" + request.getParameter("signature"));
118. // System.out.println("timestamp:" + request.getParameter("timestamp"));
119. // System.out.println("nonce:" + request.getParameter("nonce"));
120. // System.out.println("echostr:" + request.getParameter("echostr"));
122. // 获取腾讯推送的数据流 经测试成功
123. InputStream is = request.getInputStream();
125. // 获取io流数据,如果是验证消息真实性即token，则不会有io流数据，否则一般为微信推送数据给我们，即有人关注了我们的公众号或者发送信息等推送事件
126. String pushData = **this**.weixinUtilBus.getInfoFromIoStream(is);
127. **if** (pushData != **null** && !"".equals(pushData)) {
128. Document document = DocumentHelper.parseText(pushData);
129. Element root = document.getRootElement();
131. String msgType = root.element("MsgType").getText();
133. **if** ("event".equals(msgType)) {
135. // 事件 click、unSubscribe、subscribe
136. String event = root.element("Event").getText();
137. **if** ("click".equals(event.toLowerCase())) {
138. // 点击按钮事件
139. String eventKey = root.element("EventKey").getText();
141. **if** ("aGuangSoftTest".equals(eventKey)) {
142. // 点击者openId
143. String fromUserName = root.element("FromUserName")
144. .getText();
145. // 接收的时间
146. **int** createTime = Integer.parseInt(root
147. .element("CreateTime").getText().toString());
148. // 回复内容
149. String content = WeiXinContants.AGUANG\_SOFT\_TEST\_CLICK\_CONTENT;
150. // 生成微信接收格式的XML字符串
151. String replyAguangSoftTestContentXMLStr = **this**.weixinUtilBus
152. .getReplyConentXmlStr(content, fromUserName,
153. createTime);
154. // 响应回去,注意是以text/xml的形式返回
155. response.setContentType("text/xml;charset=UTF-8");
156. response.getWriter().write(
157. replyAguangSoftTestContentXMLStr);
158. }
159. } **else** **if** ("unsubscribe".equals(event.toLowerCase())) {
160. // 取消关注事件
162. // 取消者openId
163. String fromUserName = root.element("FromUserName")
164. .getText();
165. // 将该记录查出，然后删除
166. List<AgWeixinOpenId> list = **this**.weixinOpenIdBus
167. .findByProperty(WeixinOpenIdDAO.OPEN\_ID,
168. fromUserName);
169. **if** (list != **null** && list.size() > 0) {
170. **this**.weixinOpenIdBus.delete(list.get(0).getId());
171. }
173. } **else** **if** ("subscribe".equals(event.toLowerCase())) {
174. // 关注事件
175. // 关注者openId
176. String fromUserName = root.element("FromUserName")
177. .getText();
178. AgWeixinOpenId woi = **new** AgWeixinOpenId();
179. woi.setCreateTime(**new** Date());
180. woi.setOpenId(fromUserName);
181. **this**.weixinOpenIdBus.save(woi);
182. } **else** **if** ("scancode\_waitmsg".equals(event)) {
183. String scanResult = root.element("ScanCodeInfo")
184. .element("ScanResult").getText();
185. **int** createTime = Integer.parseInt(root
186. .element("CreateTime").getText());
188. // 点击扫描者
189. String fromUserName = root.element("FromUserName")
190. .getText();
191. // 生成XML字符串数据
192. String scancodeResultXMLStr = **this**.weixinUtilBus
193. .scancodeResultXMLStr(scanResult, fromUserName,
194. createTime);
195. // 响应回去
196. response.setContentType("text/xml;charset=UTF-8");
197. response.getWriter().write(scancodeResultXMLStr);
198. }
199. } **else** **if** ("text".equals(msgType)) {
200. // 接收普通消息信息内容
202. // 发送消息者openId
203. String fromUserName = root.element("FromUserName").getText();
204. // 接收的时间
205. **int** createTime = Integer.parseInt(root.element("CreateTime")
206. .getText().toString());
207. // 接收的消息
208. String content = root.element("Content").getText();
209. Element weather = **this**.getWeatherInfoByCity(content);
210. **if** (weather == **null**) {
211. System.out.println("输入城市名可以查询当天的天气预报情况，如：广州");
212. // 生成微信接收格式的XML字符串
213. String replyAguangSoftTestContentXMLStr = **this**.weixinUtilBus
214. .getReplyConentXmlStr("输入城市名可以查询当天的天气预报情况，如：广州",
215. fromUserName, createTime);
216. // 响应回去,注意是以text/xml的形式返回
217. response.setContentType("text/xml;charset=UTF-8");
218. response.getWriter()
219. .write(replyAguangSoftTestContentXMLStr);
220. } **else** {
221. // 回复内容
222. String weatherInfo = "您输入的城市为："
223. + weather.element("city").getText() + ",天气："
224. + weather.element("status1").getText() + "->"
225. + weather.element("status2").getText() + "。风向："
226. + weather.element("direction1").getText()
227. + "。紫外线指数："
228. + weather.element("pollution").getText() + "。温度："
229. + weather.element("ssd\_l").getText() + "。注意事项："
230. + weather.element("gm\_s").getText() + "。"
231. + weather.element("yd\_s").getText() + "。"
232. + weather.element("xcz\_s").getText() + "。"
233. + weather.element("pollution\_s").getText();
234. System.out.println(weatherInfo);
235. // 生成微信接收格式的XML字符串
236. String replyAguangSoftTestContentXMLStr = **this**.weixinUtilBus
237. .getReplyConentXmlStr(weatherInfo, fromUserName,
238. createTime);
239. // 响应回去,注意是以text/xml的形式返回
240. response.setContentType("text/xml;charset=UTF-8");
241. response.getWriter()
242. .write(replyAguangSoftTestContentXMLStr);
243. }
244. }
245. }
247. is.close();
249. // 如果这3个参数有为空的，即不是验证消息的真实性，那么就返回空，因为是处理上面的其他时间，下面的验证无需执行
250. **if** (signature == **null** || timestamp == **null** || nonce == **null**) {
251. **return** **null**;
252. }
254. // 通过检验signature对请求进行校验，若校验成功则原样返回echostr，表示接入成功，否则接入失败
255. **if** (SignUtil.checkSignature(signature, timestamp, nonce)) {
256. response.setContentType("text/html;charset=UTF-8");
257. response.getWriter().write(echostr);
258. }
260. **return** **null**;
261. }
263. /\*\*
264. \* <b>获取最新accessToken<b>
265. \*
266. \* @return String
267. \*/
268. **public** String getAccessToken() {
269. String apiUrl = "https://api.weixin.qq.com/cgi-bin/token?grant\_type="
270. + WeiXinContants.GRANT\_TYPE + "&appid=" + WeiXinContants.APP\_ID
271. + "&secret=" + WeiXinContants.APP\_SECRET;
273. JSONObject json = JSONObject.fromObject(HttpUtils.getInstance()
274. .getJsonObjectByUrl(apiUrl));
276. String access\_token = json.get("access\_token") + "";
277. System.out.println("access\_token:" + access\_token);
278. **return** access\_token;
279. }
281. /\*\*
282. \* 获取微信服务ip 列表
283. \*/
284. **public** String getWeiXinServiceIp() {
285. String apiUrl = "https://api.weixin.qq.com/cgi-bin/getcallbackip?access\_token="
286. + WeiXinContants.ACCESS\_TOKEN;
288. JSONObject json = JSONObject.fromObject(HttpUtils.getInstance()
289. .getJsonObjectByUrl(apiUrl));
290. String ipStr = json.get("ip\_list") + "";
291. System.out.println(ipStr);
292. **return** ipStr;
293. }
295. /\*\*
296. \* 创建菜单
297. \*/
298. **public** String createMenu() **throws** Exception {
299. String apiUrl = "https://api.weixin.qq.com/cgi-bin/menu/create?access\_token="
300. + **new** AguangWeiXinSoftAction().getAccessToken();
301. // 推送给微信服务器的json数据
302. JSONObject data = **new** JSONObject();
304. // 按钮json数组数据
305. JSONArray buttonJsonArray = **new** JSONArray();
306. // 菜单按钮1
307. JSONObject button1 = **new** JSONObject();
308. button1.put("type", "click");
309. button1.put("name", "普通按钮");
310. button1.put("key", "aGuangSoftTest");
311. // 添加按钮1
312. buttonJsonArray.add(button1);
314. // 菜单按钮2
315. JSONObject button2 = **new** JSONObject();
316. button2.put("name", "多级菜单");
317. JSONArray sub\_button2 = **new** JSONArray();
318. // 菜单按钮2中的子按钮1
319. JSONObject button2Sub1 = **new** JSONObject();
320. // view表示点击后跳转链接
321. button2Sub1.put("type", "view");
322. button2Sub1.put("name", "谁的服务");
323. button2Sub1.put("url", "http://www.aguang.cc/weixin/");
325. // 菜单按钮2中的子按钮2
326. JSONObject button2Sub2 = **new** JSONObject();
327. // view表示点击后跳转链接
328. button2Sub2.put("type", "view");
329. button2Sub2.put("name", "我的信息");
330. button2Sub2.put("url", "http://www.aguang.cc/weixin/wx/");
332. // 菜单按钮2中的子按钮3
333. JSONObject button2Sub3 = **new** JSONObject();
334. // 表示点击后跳转链接
335. button2Sub3.put("type", "view");
336. button2Sub3.put("name", "赞一个");
337. button2Sub3.put("url", "http://www.aguang.cc/weixin/wx/");
339. sub\_button2.add(button2Sub1);
340. sub\_button2.add(button2Sub2);
341. sub\_button2.add(button2Sub3);
343. // 将子菜单绑定到菜单2按钮中
344. button2.put("sub\_button", sub\_button2.toString());
346. // 添加菜单按钮2
347. buttonJsonArray.add(button2);
349. // 菜单按钮3
350. JSONObject button3 = **new** JSONObject();
351. button3.put("name", "最新按钮");
353. // 定义菜单按钮3的子按钮json数组
354. JSONArray sub\_button3 = **new** JSONArray();
355. // 菜单按钮3中的子按钮1
356. JSONObject button3Sub1 = **new** JSONObject();
357. button3Sub1.put("type", "scancode\_waitmsg");
358. button3Sub1.put("name", "扫码带提示");
359. button3Sub1.put("key", "searchCode");
360. button3Sub1.put("sub\_button", "[]");
361. // 将子按钮绑定到菜单按钮3中
362. sub\_button3.add(button3Sub1.toString());
364. // 菜单按钮3中的子按钮2
365. JSONObject button3Sub2 = **new** JSONObject();
366. button3Sub2.put("type", "scancode\_push");
367. button3Sub2.put("name", "扫码推事件");
368. button3Sub2.put("key", "searchCodePushEvent");
369. button3Sub2.put("sub\_button", "[]");
370. sub\_button3.add(button3Sub2.toString());
372. button3.put("sub\_button", sub\_button3.toString());
373. buttonJsonArray.add(button3);
375. // 最终的按钮自定义菜单数据
376. data.put("button", buttonJsonArray.toString());
378. **try** {
379. System.out.println(data.toString());
380. HttpUtils.getInstance().createPostHttpRequest(apiUrl,
381. data.toString());
382. } **catch** (Exception e) {
383. e.printStackTrace();
384. }
385. **return** **null**;
386. }
388. /\*\*
389. \*
390. \* <b>微信菜单2的多级菜单1<b>
391. \*
392. \* @return String
393. \*/
394. **public** String menu2Button1() {
395. **return** SUCCESS;
396. }
398. /\*\*
399. \*
400. \* <b>微信菜单2的多级菜单2<b>
401. \*
402. \* @return String
403. \*/
404. **public** String menu2Button2() {
405. **return** SUCCESS;
406. }
408. /\*\*
409. \*
410. \* <b>微信菜单2的多级菜单1<b>
411. \*
412. \* @return String
413. \*/
414. **public** String menu2Button3() {
415. **return** SUCCESS;
416. }
418. /\*\*
419. \* <b>根据openId获取用户信息<b>
420. \*
421. \* @return String
422. \*/
423. **public** String getUserInfoByOpenId() {
424. String openId = "";
425. String apiUrl = "https://api.weixin.qq.com/cgi-bin/user/info?access\_token="
426. + getAccessToken() + "&openid=" + openId + "&lang=zh\_CN";
428. **try** {
429. HttpUtils.getInstance().createPostHttpRequest(apiUrl, "");
430. } **catch** (Exception e) {
431. e.printStackTrace();
432. }
433. **return** SUCCESS;
434. }
436. /\*\*
437. \*
438. \* <b>根据城市名获取该城市的天气预报<b>
439. \*
440. \* @return String
441. \*/
442. **private** Element getWeatherInfoByCity(String city) {
443. **try** {
444. String apiUrl = "http://php.weather.sina.com.cn/xml.php?city="
445. + URLEncoder.encode(city, "gbk")
446. + "&password=DJOYnieT8234jlsK&day=0";
447. String data = HttpUtils.getInstance().getXmlDataByUrl(apiUrl);
448. System.out.println(data);
449. Document document = DocumentHelper.parseText(data);
450. Element root = document.getRootElement();
451. // 事件 click、unSubscribe、subscribe
452. Element weather = root.element("Weather");
453. **if** (weather == **null**) {
454. **return** **null**;
455. } **else** {
456. **return** weather;
457. }
458. } **catch** (Exception e) {
459. // e.printStackTrace();
460. System.out.println("转码失败");
461. **return** **null**;
462. }
463. }
464. }

4、action的测试类（自定义菜单主要是在测试类运行）

**[java]** [view plain](https://blog.csdn.net/aguang_vip/article/details/45915887) [copy](https://blog.csdn.net/aguang_vip/article/details/45915887)

1. **package** cn.com.aguang.weixin.action.weixin;
3. **import** java.net.URLEncoder;
4. **import** java.util.Date;
6. **import** net.sf.json.JSONArray;
7. **import** net.sf.json.JSONObject;
9. **import** org.dom4j.Document;
10. **import** org.dom4j.DocumentHelper;
11. **import** org.dom4j.Element;
12. **import** org.junit.BeforeClass;
13. **import** org.junit.Test;
14. **import** org.springframework.context.ApplicationContext;
15. **import** org.springframework.context.support.ClassPathXmlApplicationContext;
17. **import** cn.com.aguang.commons.business.BusinessException;
18. **import** cn.com.aguang.weixin.bussiness.WeixinOpenIdBus;
19. **import** cn.com.aguang.weixin.po.AgWeixinOpenId;
20. **import** cn.com.aguang.weixin.utils.HttpUtils;
21. **import** cn.com.aguang.weixin.utils.WeiXinContants;
23. **import** com.opensymphony.xwork2.interceptor.annotations.After;
24. **import** com.opensymphony.xwork2.interceptor.annotations.Before;
26. **public** **class** AguangWeiXinSoftTest {
28. **private** **static** WeixinOpenIdBus weixinOpenIdBus;
30. /\*\*
31. \* @throws java.lang.Exception
32. \*/
33. @BeforeClass
34. **public** **static** **void** setUpBeforeClass() {
35. ApplicationContext ctx = **new** ClassPathXmlApplicationContext(
36. "applicationContext.xml");
37. weixinOpenIdBus = (WeixinOpenIdBus) ctx.getBean("weixinOpenIdBus");
38. }
40. /\*\*
41. \* @throws java.lang.Exception
42. \*/
43. @Before
44. **public** **void** setUp() **throws** Exception {
45. System.out.println("---------------Begin");
46. }
48. /\*\*
49. \* @throws java.lang.Exception
50. \*/
51. @After
52. **public** **void** tearDown() **throws** Exception {
53. System.out.println("---------------End");
54. }
56. /\*\*
57. \* 更新access\_token
58. \*/
59. **public** **void** updateAccessToken() {
60. String access\_token = getAccessToken();
61. System.out.println("更新之前的access\_token:" + WeiXinContants.ACCESS\_TOKEN);
62. **if** (access\_token != **null** && "".equals(access\_token)) {
63. WeiXinContants.setAccessToken(access\_token);
64. }
65. System.out.println("更新之后的access\_token:" + access\_token);
66. }
68. /\*\*
69. \* 获取微信服务ip 列表
70. \*/
71. **public** **void** getWeiXinServiceIp() {
72. String apiUrl = "https://api.weixin.qq.com/cgi-bin/getcallbackip?access\_token="
73. + WeiXinContants.ACCESS\_TOKEN;
75. JSONObject json = JSONObject.fromObject(HttpUtils.getInstance()
76. .getJsonObjectByUrl(apiUrl));
77. String ipStr = json.get("ip\_list") + "";
78. System.out.println(ipStr);
79. }
81. /\*\*
82. \* 创建自定义菜单
83. \*/
84. **public** **void** createMenu() {
85. String apiUrl = "https://api.weixin.qq.com/cgi-bin/menu/create?access\_token="
86. + getAccessToken();
87. // 推送给微信服务器的json数据
88. JSONObject data = **new** JSONObject();
90. // 按钮json数组数据
91. JSONArray buttonJsonArray = **new** JSONArray();
92. // 菜单按钮1
93. JSONObject button1 = **new** JSONObject();
94. button1.put("type", "click");
95. button1.put("name", "普通按钮");
96. button1.put("key", "aGuangSoftTest");
97. // 添加按钮1
98. buttonJsonArray.add(button1);
100. // 菜单按钮2
101. JSONObject button2 = **new** JSONObject();
102. button2.put("name", "多级菜单");
103. JSONArray sub\_button2 = **new** JSONArray();
104. // 菜单按钮2中的子按钮1
105. JSONObject button2Sub1 = **new** JSONObject();
106. // view表示点击后跳转链接
107. button2Sub1.put("type", "view");
108. button2Sub1.put("name", "谁的服务");
109. button2Sub1.put("url",
110. "http://www.aguang.cc/weixin/wx/menu2Button1.action");
112. // 菜单按钮2中的子按钮2
113. JSONObject button2Sub2 = **new** JSONObject();
114. // view表示点击后跳转链接
115. button2Sub2.put("type", "view");
116. button2Sub2.put("name", "我的信息");
117. button2Sub2.put("url",
118. "http://www.aguang.cc/weixin/wx/menu2Button2.action");
120. // 菜单按钮2中的子按钮3
121. JSONObject button2Sub3 = **new** JSONObject();
122. // 表示点击后跳转链接
123. button2Sub3.put("type", "view");
124. button2Sub3.put("name", "赞一个");
125. button2Sub3.put("url",
126. "http://www.aguang.cc/weixin/wx/menu2Button3.action");
128. sub\_button2.add(button2Sub1);
129. sub\_button2.add(button2Sub2);
130. sub\_button2.add(button2Sub3);
132. // 将子菜单绑定到菜单2按钮中
133. button2.put("sub\_button", sub\_button2.toString());
135. // 添加菜单按钮2
136. buttonJsonArray.add(button2);
138. // 菜单按钮3
139. JSONObject button3 = **new** JSONObject();
140. button3.put("name", "最新按钮");
142. // 定义菜单按钮3的子按钮json数组
143. JSONArray sub\_button3 = **new** JSONArray();
144. // 菜单按钮3中的子按钮1
145. JSONObject button3Sub1 = **new** JSONObject();
146. button3Sub1.put("type", "scancode\_waitmsg");
147. button3Sub1.put("name", "扫码带提示");
148. button3Sub1.put("key", "searchCode");
149. button3Sub1.put("sub\_button", "[]");
150. // 将子按钮绑定到菜单按钮3中
151. sub\_button3.add(button3Sub1.toString());
153. // 菜单按钮3中的子按钮2
154. JSONObject button3Sub2 = **new** JSONObject();
155. button3Sub2.put("type", "scancode\_push");
156. button3Sub2.put("name", "扫码推事件");
157. button3Sub2.put("key", "searchCodePushEvent");
158. button3Sub2.put("sub\_button", "[]");
159. sub\_button3.add(button3Sub2.toString());
161. button3.put("sub\_button", sub\_button3.toString());
162. buttonJsonArray.add(button3);
164. // 最终的按钮自定义菜单数据
165. data.put("button", buttonJsonArray.toString());
167. **try** {
168. System.out.println(data.toString());
169. HttpUtils.getInstance().createPostHttpRequest(apiUrl,
170. data.toString());
171. } **catch** (Exception e) {
172. e.printStackTrace();
173. }
174. }
176. /\*\*
177. \* <b>根据openId获取用户信息，需要认证才可以，暂时无法使用，还没有认证<b>
178. \*
179. \* @return String
180. \*/
181. **public** **void** getUserInfoByOpenId() {
182. String openId = "cJCc8GeFCJu8kHocFLqg";
183. String apiUrl = "https://api.weixin.qq.com/cgi-bin/user/info?access\_token="
184. + getAccessToken() + "&openid=" + openId + "&lang=zh\_CN";
186. **try** {
187. JSONObject json = JSONObject.fromObject(HttpUtils.getInstance()
188. .createPostHttpRequest(apiUrl, ""));
189. System.out.println("国家" + json.get("country"));
190. System.out.println("省份" + json.get("province"));
191. System.out.println("城市" + json.get("city"));
192. System.out.println("昵称" + json.get("nickname"));
193. System.out.println("头像地址" + json.get("headimgurl"));
194. } **catch** (Exception e) {
195. e.printStackTrace();
196. }
197. }
199. /\*\*
200. \* <b>获取最新accessToken<b>
201. \*
202. \* @return String
203. \*/
204. **private** String getAccessToken() {
205. String apiUrl = "https://api.weixin.qq.com/cgi-bin/token?grant\_type="
206. + WeiXinContants.GRANT\_TYPE + "&appid=" + WeiXinContants.APP\_ID
207. + "&secret=" + WeiXinContants.APP\_SECRET;
209. JSONObject json = JSONObject.fromObject(HttpUtils.getInstance()
210. .getJsonObjectByUrl(apiUrl));
212. String access\_token = json.get("access\_token") + "";
213. System.out.println("access\_token:" + access\_token);
214. **return** access\_token;
215. }
217. /\*\*
218. \* 添加关注记录
219. \*/
220. **public** **void** addWeixinOpenId() {
221. AgWeixinOpenId woi = **new** AgWeixinOpenId();
222. woi.setOpenId("123");
223. woi.setCreateTime(**new** Date());
225. **try** {
226. weixinOpenIdBus.save(woi);
227. } **catch** (BusinessException e) {
228. e.printStackTrace();
229. }
230. }
232. /\*\*
233. \*
234. \* 根据城市获取天气预报信息
235. \*
236. \* @param city
237. \*/
238. @Test
239. **public** **void** getWeatherInfoByCity() {
240. String city = "广州";
241. **try** {
242. String apiUrl = "http://php.weather.sina.com.cn/xml.php?city="
243. + URLEncoder.encode(city, "gbk")
244. + "&password=DJOYnieT8234jlsK&day=0";
245. String data = HttpUtils.getInstance().getXmlDataByUrl(apiUrl);
246. System.out.println(data);
247. Document document = DocumentHelper.parseText(data);
248. Element root = document.getRootElement();
249. // 事件 click、unSubscribe、subscribe
250. Element weather = root.element("Weather");
251. **if** (weather == **null**) {
252. System.out.println("无数据");
253. } **else** {
254. // 回复内容
255. String weatherInfo = "您输入的城市为："
256. + weather.element("city").getText() + "。天气："
257. + weather.element("status1").getText() + "->"
258. + weather.element("status2").getText() + "。风向:"
259. + weather.element("direction1").getText() + "。紫外线指数："
260. + weather.element("pollution").getText() + "。温度："
261. + weather.element("ssd\_l").getText() + "。注意事项："
262. + weather.element("gm\_s").getText() + "。"
263. + weather.element("yd\_s").getText() + "。"
264. + weather.element("xcz\_s").getText() + "。"
265. + weather.element("pollution\_s").getText();
266. System.out.println(weatherInfo);
267. }
268. } **catch** (Exception e) {
269. // e.printStackTrace();
270. System.out.println("转码失败");
271. }
272. }
273. }

5、几个相关的bus类,直接上实现类就行了

**[java]** [view plain](https://blog.csdn.net/aguang_vip/article/details/45915887) [copy](https://blog.csdn.net/aguang_vip/article/details/45915887)

1. **package** cn.com.aguang.weixin.bussiness.impl;
3. **import** javax.annotation.Resource;
5. **import** org.springframework.stereotype.Service;
7. **import** cn.com.aguang.commons.business.AbstractBaseBus;
8. **import** cn.com.aguang.commons.dao.BaseDAO;
9. **import** cn.com.aguang.weixin.bussiness.WeixinOpenIdBus;
10. **import** cn.com.aguang.weixin.dao.WeixinOpenIdDAO;
11. **import** cn.com.aguang.weixin.po.AgWeixinOpenId;
13. @Service("weixinOpenIdBus")
14. **public** **class** WeixinOpenIdBusImpl **extends**
15. AbstractBaseBus<AgWeixinOpenId, String> **implements** WeixinOpenIdBus {
17. **private** WeixinOpenIdDAO weixinOpenIdDAO;
19. @Resource
20. **public** **void** setWeixinOpenIdDAO(WeixinOpenIdDAO weixinOpenIdDAO) {
21. **this**.weixinOpenIdDAO = weixinOpenIdDAO;
22. }
24. @Override
25. **protected** BaseDAO<AgWeixinOpenId, String> getDAO() {
26. **return** weixinOpenIdDAO;
27. }
29. @Override
30. **protected** **void** checkRequires(AgWeixinOpenId paramT) {
31. // do nothing
32. }
34. }

**[java]** [view plain](https://blog.csdn.net/aguang_vip/article/details/45915887) [copy](https://blog.csdn.net/aguang_vip/article/details/45915887)

1. **package** cn.com.aguang.weixin.bussiness.impl;
3. **import** java.io.IOException;
4. **import** java.io.InputStream;
6. **import** org.springframework.stereotype.Service;
8. **import** cn.com.aguang.weixin.bussiness.WeixinUtilBus;
9. **import** cn.com.aguang.weixin.utils.WeiXinContants;
11. @Service("weixinUtilBus")
12. **public** **class** WeixinUtilBusImpl **implements** WeixinUtilBus {
14. /\*\*
15. \* 根据IO流获取IO流的数据
16. \*/
17. @Override
18. **public** String getInfoFromIoStream(InputStream is) {
19. String pushData = "";
20. **try** {
21. **if** (is != **null**) {
22. String charSet = **null**;
23. charSet = "utf-8";
24. **byte** b[] = **new** **byte**[200];
25. **int** numRead = is.read(b);
26. **if** (numRead != -1) {
27. String content = **new** String(b, 0, numRead);
28. **while** (numRead != -1) {
29. numRead = is.read(b);
30. **if** (numRead != -1) {
31. String newContent = **new** String(b, 0, numRead,
32. charSet);
33. content += newContent;
34. }
35. }
36. pushData = content;
37. System.out.println("接收到推送的信息：" + content);
38. }
39. }
40. **return** pushData;
41. } **catch** (IOException e) {
42. **return** "";
43. }
44. }
46. /\*\*
47. \* 根据内容，接收者openId获取应该相应的XML字符串
48. \*/
49. @Override
50. **public** String getReplyConentXmlStr(String content, String toUserName,
51. **int** createTime) {
52. String replyAguangSoftTestContentXMLStr = "<xml><ToUserName><![CDATA["
53. + toUserName
54. + "]]></ToUserName><FromUserName><![CDATA["
55. + WeiXinContants.WEI\_XIN\_NO
56. + "]]></FromUserName><CreateTime>"
57. + createTime
58. + "</CreateTime><MsgType><![CDATA[text]]></MsgType><Content><![CDATA["
59. + content + "]]></Content></xml>";
60. **return** replyAguangSoftTestContentXMLStr;
61. }
63. /\*\*
64. \* 根据扫描内容，接收者openId获取应该相应的XML字符串
65. \*/
66. @Override
67. **public** String scancodeResultXMLStr(String scanResult, String toUserName,
68. **int** createTime) {
69. String scancodeResultXMLStr = "<xml><ToUserName><![CDATA["
70. + toUserName
71. + "]]></ToUserName><FromUserName><![CDATA["
72. + WeiXinContants.WEI\_XIN\_NO
73. + "]]></FromUserName><CreateTime>"
74. + createTime
75. + "</CreateTime><MsgType><![CDATA[text]]></MsgType><Content><![CDATA["
76. + scanResult + "]]></Content></xml>";
77. **return** scancodeResultXMLStr;
78. }
80. }

6、相关的工具类，主要是请求封装的工具类，微信公众平台的appid等参数定义

**[java]** [view plain](https://blog.csdn.net/aguang_vip/article/details/45915887) [copy](https://blog.csdn.net/aguang_vip/article/details/45915887)

1. **package** cn.com.aguang.weixin.utils;
3. **public** **class** WeiXinContants {
5. **public** **static** **final** String WEI\_XIN\_NO = "AGuangSoftTest"; //公众平台设置的微信号，发送信息时需要
7. **public** **static** **final** String APP\_ID = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*";//根据自己的替换
9. **public** **static** **final** String APP\_SECRET = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"; 根据自己的替换
11. **public** **static** **final** String TOKEN = "\*\*\*";  //根据自己的替换
13. **public** **static** **final** String GRANT\_TYPE = "client\_credential";
15. **public** **static** String ACCESS\_TOKEN = "Cx\_JPR-Cs791HRhDcA2jjm9U\_UtHEPaOXT3q-LL7six96i9u0\_-olxXoaHUbhGz3Cal7sp2C2KM0qerk4fpPnU7\_iZCKhi42nOaKE2SA-Os";
17. **public** **static** String AGUANG\_SOFT\_TEST\_CLICK\_CONTENT = "接口点击事件回复调试成功，这只是一小步";
19. **public** **static** **void** setAccessToken(String access\_token) {
20. ACCESS\_TOKEN = access\_token;
21. }
22. }

封装的请求工具类

**[java]** [view plain](https://blog.csdn.net/aguang_vip/article/details/45915887) [copy](https://blog.csdn.net/aguang_vip/article/details/45915887)

1. **package** cn.com.aguang.weixin.utils;
3. **import** java.io.BufferedReader;
4. **import** java.io.InputStream;
5. **import** java.io.InputStreamReader;
6. **import** java.io.OutputStreamWriter;
7. **import** java.net.HttpURLConnection;
8. **import** java.net.URL;
10. **import** net.sf.json.JSONArray;
11. **import** net.sf.json.JSONObject;
13. /\*\*
14. \*
15. \* 饿汉试 单例 直接实例化一个，线程安全(以前一直用懒汉式，第一次调用实例化，但是有可能会出现多个实例，非常小的概率)
16. \*
17. \* @author aGuang
18. \*
19. \*/
20. **public** **class** HttpUtils {
22. **private** **static** HttpUtils instance = **new** HttpUtils();
24. /\*\*
25. \* 私有构造方法
26. \*/
27. **private** HttpUtils() {
29. }
31. **public** **static** HttpUtils getInstance() {
32. **return** instance;
33. }
35. /\*\*
36. \* 根据api访问地址获取地址返回的json数据
37. \*
38. \* @param apiUrl
39. \*
40. \* @return json
41. \*/
42. **public** String getJsonObjectByUrl(String apiUrl) {
43. **try** {
44. URL url = **new** URL(apiUrl);
45. HttpURLConnection con = (HttpURLConnection) url.openConnection();
46. con.setAllowUserInteraction(**false**);
47. InputStream urlStream = url.openStream();
48. String charSet = **null**;
49. charSet = "utf-8";
50. **byte** b[] = **new** **byte**[200];
51. **int** numRead = urlStream.read(b);
52. String content = **new** String(b, 0, numRead);
53. **while** (numRead != -1) {
54. numRead = urlStream.read(b);
55. **if** (numRead != -1) {
56. String newContent = **new** String(b, 0, numRead, charSet);
57. content += newContent;
58. }
59. }
60. // System.out.println("content:" + content);
61. JSONObject json = JSONObject.fromObject(**new** String(content
62. .getBytes(), "utf-8"));
64. urlStream.close();
65. **return** json.toString();
66. } **catch** (Exception e) {
67. e.printStackTrace();
68. **return** **null**;
69. }
70. }
72. /\*\*
73. \* 根据api访问地址获取地址返回的json数据
74. \*
75. \* @param apiUrl
76. \*
77. \* @return
78. \*/
79. **public** String getJsonArrayByUrl(String apiUrl) {
80. **try** {
81. URL url = **new** URL(apiUrl);
82. HttpURLConnection con = (HttpURLConnection) url.openConnection();
83. con.setAllowUserInteraction(**false**);
84. InputStream urlStream = url.openStream();
85. String charSet = **null**;
86. charSet = "utf-8";
87. **byte** b[] = **new** **byte**[200];
88. **int** numRead = urlStream.read(b);
89. String content = **new** String(b, 0, numRead);
90. **while** (numRead != -1) {
91. numRead = urlStream.read(b);
92. **if** (numRead != -1) {
93. String newContent = **new** String(b, 0, numRead, charSet);
94. content += newContent;
95. }
96. }
97. // System.out.println("content:" + content);
98. JSONArray array = JSONArray.fromObject(**new** String(content
99. .getBytes(), "utf-8"));
101. urlStream.close();
102. **return** array.toString();
103. } **catch** (Exception e) {
104. // e.printStackTrace();
105. **return** **null**;
106. }
107. }
109. /\*\*
110. \* 根据api访问地址获取地址返回的XML数据
111. \*
112. \* @param apiUrl
113. \*
114. \* @return
115. \*/
116. **public** String getXmlDataByUrl(String apiUrl) {
117. **try** {
118. URL url = **new** URL(apiUrl);
119. HttpURLConnection con = (HttpURLConnection) url.openConnection();
120. con.setAllowUserInteraction(**false**);
121. InputStream urlStream = url.openStream();
122. String charSet = **null**;
123. charSet = "utf-8";
124. **byte** b[] = **new** **byte**[6000];
125. **int** numRead = urlStream.read(b);
126. String content = **new** String(b, 0, numRead);
127. **while** (numRead != -1) {
128. numRead = urlStream.read(b);
129. **if** (numRead != -1) {
130. String newContent = **new** String(b, 0, numRead, charSet);
131. content += newContent;
132. }
133. }
134. // System.out.println("content:" + content);
135. String xmlString = **new** String(content.getBytes(), "utf-8");
137. urlStream.close();
138. **return** xmlString;
139. } **catch** (Exception e) {
140. // e.printStackTrace();
141. **return** **null**;
142. }
143. }
145. /\*\*
146. \* <b>用post请求访问一个http<b>
147. \*
148. \* @param apiUrl
149. \*            请求地址
150. \* @return
151. \*/
152. **public** String createPostHttpRequest(String apiUrl, String data)
153. **throws** Exception {
154. // Post请求的url，与get不同的是不需要带参数
155. URL postUrl = **new** URL(apiUrl);
156. // 打开连接
157. HttpURLConnection connection = (HttpURLConnection) postUrl
158. .openConnection();
159. // 25秒连接超时
160. connection.setConnectTimeout(25000);
161. // 读取超时 --服务器响应比较慢,增大时间
162. connection.setReadTimeout(25000);
164. HttpURLConnection.setFollowRedirects(**true**);
166. // 设置是否向connection输出，因为这个是post请求，参数要放在
167. // http正文内，因此需要设为true
168. connection.setDoOutput(**true**);
169. // Read from the connection. Default is true.
170. connection.setDoInput(**true**);
171. // Set the post method. Default is GET
172. connection.setRequestMethod("POST");
173. // Post cannot use caches
174. // Post 请求不能使用缓存
175. connection.setUseCaches(**false**);
176. // URLConnection.setFollowRedirects是static函数，作用于所有的URLConnection对象。
177. // connection.setFollowRedirects(true);
179. // URLConnection.setInstanceFollowRedirects是成员函数，仅作用于当前函数
180. connection.setInstanceFollowRedirects(**true**);
181. // 配置本次连接的Content-type，配置为application/x-www-form-urlencoded的
182. // 意思是正文是urlencoded编码过的form参数，下面我们可以看到我们对正文内容使用URLEncoder.encode
183. // 进行编码
184. // connection.setRequestProperty("Content-Type",
185. // "application/x-www-form-urlencoded");
186. connection
187. .setRequestProperty("User-Agent",
188. "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:21.0) Gecko/20100101 Firefox/21.0");
189. connection.setRequestProperty("Referer", "https://api.weixin.qq.com/");
190. // 连接，从postUrl.openConnection()至此的配置必须要在connect之前完成，
191. // 要注意的是connection.getOutputStream会隐含的进行connect。
192. connection.connect();
193. OutputStreamWriter out = **new** OutputStreamWriter(
194. connection.getOutputStream());
195. // The URL-encoded contend
196. // 正文，正文内容其实跟get的URL中'?'后的参数字符串一致
197. // String content = URLEncoder.encode("中国聚龙", "utf-8");
198. String content = data;
199. // DataOutputStream.writeBytes将字符串中的16位的unicode字符以8位的字符形式写道流里面
200. out.write(content);
201. out.flush();
202. out.close(); // flush and close
203. BufferedReader reader = **new** BufferedReader(**new** InputStreamReader(
204. connection.getInputStream(), "utf-8"));// 设置编码,否则中文乱码
205. String line = "";
206. System.out.println("=============================");
207. System.out.println("Contents of post request");
208. System.out.println("=============================");
209. **while** ((line = reader.readLine()) != **null**) {
210. // line = new String(line.getBytes(), "utf-8");
211. System.out.println("返回的信息:" + line);
212. }
213. System.out.println("=============================");
214. System.out.println("Contents of post request ends");
215. System.out.println("=============================");
216. reader.close();
217. connection.disconnect();
218. **return** "";
219. }
220. }

上面的几个代码就是我的公众号测试开发上写的，如果有什么不对的，请指出