**微信公众号平台项目**

# 什么是微信公众平台

微信公众号主要面向名人、政府、媒体、企业等机构推出的合作推广业务。在这里可以通过微信渠道将品牌推广给上亿的微信用户，减少宣传成本，提高品牌知名度，打造更具影响力的品牌形象。

# 初始微信公众平台

## 官方网址

<https://mp.weixin.qq.com/cgi-bin/loginpage?t=wxm2-login&lang=zh_CN>

### 帐号分类



## 微信公众平台介绍

### 消息管理

### 用户管理

### 素材管理

### 基本设置

#### 服务器配置(未启用)

启用并设置服务器配置后，用户发给公众号的消息以及开发者需要的事件推送，将被微信转发到该URL中

#### 外网映射工具

●windows用户:

1,下载windows版本的客户端，解压到你喜欢的目录  
2,在命令行下进入到path/to/windows\_386/下  
3,执行 ngrok -config=ngrok.cfg -subdomain xxx 80 //(xxx 是你自定义的域名前缀)

4,如果开启成功 你就可以使用 xxx.tunnel.qydev.com 来访问你本机的 127.0.0.1:80 的服务啦  
5,如果你自己有顶级域名，想通过自己的域名来访问本机的项目，那么先将自己的顶级域名解析到123.57.165.240(域名需要已备案哦)，然后执行./ngrok -config=ngrok.cfg -hostname xxx.xxx.xxx 80 //(xxx.xxx.xxx是你自定义的顶级域名)  
6，如果开启成功 你就可以使用你的顶级域名来访问你本机的 127.0.0.1:80 的服务啦  
7,如果失败 就加下交流群 反馈下问题 本屌会看看什么原因....吧

## 公众测试平台

因为是非认证公众号，部分权限不足，建议大家测试中使用公众测试平台。

https://mp.weixin.qq.com/debug/cgi-bin/sandbox?t=sandbox/login

# 微信环境搭建

## 使用技术

Springboot、Dubbo、Zookeeper、[weixin-java-tools](https://github.com/Wechat-Group/weixin-java-tools)

## 工程结构

-----itmayiedu-weixin-parent

---itmayiedu-weixin-service-api

-----itmayiedu-weixin-service

-----itmayiedu-weixin-web

## 相关依赖

|  |
| --- |
| **<parent>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-parent</artifactId>**  **<version>1.5.3.RELEASE</version>**  **</parent>**  **<properties>**  **<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>**  **<maven.compiler.source>1.8</maven.compiler.source>**  **<maven.compiler.target>1.8</maven.compiler.target>**  **</properties>**  **<dependencies>**  **<!-- 集成commons工具类 -->**  **<dependency>**  **<groupId>org.apache.commons</groupId>**  **<artifactId>commons-lang3</artifactId>**  **<version>3.4</version>**  **</dependency>**  **<!-- 集成lombok 框架 -->**  **<dependency>**  **<groupId>org.projectlombok</groupId>**  **<artifactId>lombok</artifactId>**  **</dependency>**  **<!-- 集成redis -->**  **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-data-redis</artifactId>**  **</dependency>**  **<!-- 集成aop -->**  **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-aop</artifactId>**  **</dependency>**  **<!-- 集成web-->**  **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-web</artifactId>**  **</dependency>**  **<!-- 集成发送邮件-->**  **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-mail</artifactId>**  **</dependency>**  **<!-- 集成mysql -->**  **<dependency>**  **<groupId>mysql</groupId>**  **<artifactId>mysql-connector-java</artifactId>**  **</dependency>**  **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-test</artifactId>**  **<scope>test</scope>**  **</dependency>**  **<dependency>**  **<groupId>org.mybatis</groupId>**  **<artifactId>mybatis</artifactId>**  **<version>3.4.0</version>**  **</dependency>**  **<dependency>**  **<groupId>tk.mybatis</groupId>**  **<artifactId>mapper</artifactId>**  **<version>3.3.7</version>**  **</dependency>**  **<!-- 阿里巴巴数据源 -->**  **<dependency>**  **<groupId>com.alibaba</groupId>**  **<artifactId>druid</artifactId>**  **<version>1.0.14</version>**  **</dependency>**  **<!-- httpclient -->**  **<dependency>**  **<groupId>commons-httpclient</groupId>**  **<artifactId>commons-httpclient</artifactId>**  **<version>3.1</version>**  **</dependency>**  **<dependency>**  **<groupId>org.apache.httpcomponents</groupId>**  **<artifactId>httpclient</artifactId>**  **</dependency>**  **<dependency>**  **<groupId>com.alibaba</groupId>**  **<artifactId>fastjson</artifactId>**  **<version>1.2.30</version>**  **</dependency>**  **<dependency>**  **<groupId>junit</groupId>**  **<artifactId>junit</artifactId>**  **</dependency>**  **<dependency>**  **<groupId>org.springframework</groupId>**  **<artifactId>spring-context-support</artifactId>**  **</dependency>**  **<dependency>**  **<groupId>commons-net</groupId>**  **<artifactId>commons-net</artifactId>**  **<version>3.3</version>**  **</dependency>**  **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-web</artifactId>**  **</dependency>**  **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-actuator</artifactId>**  **</dependency>**  **<dependency>**  **<groupId>org.apache.zookeeper</groupId>**  **<artifactId>zookeeper</artifactId>**  **<version>3.4.8</version>**  **</dependency>**  **<dependency>**  **<groupId>com.101tec</groupId>**  **<artifactId>zkclient</artifactId>**  **<version>0.3</version>**  **</dependency>**  **<dependency>**  **<groupId>com.alibaba</groupId>**  **<artifactId>dubbo</artifactId>**  **<version>2.5.3</version>**  **<exclusions>**  **<exclusion>**  **<groupId>org.springframework</groupId>**  **<artifactId>spring</artifactId>**  **</exclusion>**  **</exclusions>**  **</dependency>**  **</dependencies>**  **<build>**  **<finalName>springboot-demo</finalName>**  **<plugins>**  **<plugin>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-maven-plugin</artifactId>**  **<configuration>**  **<fork>true</fork>**  **<mainClass>${start-class}</mainClass>**  **</configuration>**  **<executions>**  **<execution>**  **<goals>**  **<goal>repackage</goal>**  **</goals>**  **</execution>**  **</executions>**  **</plugin>**  **<plugin>**  **<artifactId>maven-assembly-plugin</artifactId>**  **<version>2.2-beta-5</version>**  **<configuration>**  **<archive>**  **<manifest>**  **<addClasspath>true</addClasspath>**  **<mainClass>com.springboot.server.demo.SampleController</mainClass>**  **</manifest>**  **</archive>**  **<descriptorRefs>**  **<descriptorRef>jar-with-dependencies</descriptorRef>**  **</descriptorRefs>**  **</configuration>**  **<executions>**  **<execution>**  **<id>assemble-all</id>**  **<phase>package</phase>**  **<goals>**  **<goal>single</goal>**  **</goals>**  **</execution>**  **</executions>**  **</plugin>**  **</plugins>**  **</build>** |

## 配置文件

application.yml

|  |
| --- |
| **server:**  **port: 8080**  **context-path:** |

# NatApp外网映射工具使用

1. 关于注册用户 -authtoken 参数

成功购买隧道以后,在 我的隧道  看到已拥有的隧道界面

[](http://jingyan.baidu.com/album/7f41ecec01c130593c095c5f.html?picindex=6)

1. 2

在命令行模式中 运行 ,

如果是windows,点击开始 运行 cmd  进入natapp.exe的目录

输入命令:  natapp -authtoken=xxxxx 回车即可

authtoken 是从上面界面的得到的,一个隧道对应一个authtoken,因此要切换不同的隧道,只需要输入不同的authtoken 即可

# 微信常用api

## 关注与取消

https://mp.weixin.qq.com/wiki?t=resource/res\_main&id=mp1421140454

## 推广支持

### 生成临时二维码

创建临时二维码ticket

https://api.weixin.qq.com/cgi-bin/qrcode/create?access\_token=sAo7AgbK-yFISxBPuWjTphNgwX4ByV0uvH24cwx\_PZClDhv3KQKCGHXybs9SsICJ-3l0\_8YpDS2neSMYNE27i0kxa6wOhvLfK90o5t4B2aoXXkqW0VyLONxR6x8pEZHHXAZeADASVU

ticket换区二维码

<https://mp.weixin.qq.com/cgi-bin/showqrcode?ticket=gQEq8DwAAAAAAAAAAS5odHRwOi8vd2VpeGluLnFxLmNvbS9xLzAyTEt0ZEV6aDRhQ1QxM19wZk5xMTQAAgR-3wVaAwSAOgkA>

代码:

|  |
| --- |
| **@RequestMapping(value = "/dispatCherServlet", method = RequestMethod.*POST*)**  **public void getDispatCherServlet(HttpServletRequest request, HttpServletResponse response, String signature,**  **String timestamp, String nonce, String echostr) throws Exception {**  **request.setCharacterEncoding("UTF-8");**  **response.setCharacterEncoding("UTF-8");**  **Map<String, String> result = XmlUtils.*parseXml*(request);**  **String toUserName = result.get("ToUserName");**  **String fromUserName = result.get("FromUserName");**  **String msgType = result.get("MsgType");**  **String content = result.get("Content");**  **String event = result.get("Event");**  **PrintWriter out = response.getWriter();**  **String resultXml = null;**  **switch (msgType) {**  **case "text":**  **if (content.equals("蚂蚁课堂")) {**  **resultXml = messageToXml(fromUserName, toUserName, "蚂蚁课堂中国高端java培训");**  **} else {**  **String resultAPiStr = HttpClientUtil**  **.*doGet*("http://api.qingyunke.com/api.php?key=free&appid=0&msg=" + content);**  **JSONObject jsonObject = new JSONObject().*parseObject*(resultAPiStr);**  **Integer state = jsonObject.getInteger("result");**  **if (state != null) {**  **String contentAPi = jsonObject.getString("content");**  **resultXml = messageToXml(fromUserName, toUserName, contentAPi);**  **}**  **}**  **break;**  **case "event":**  **// 已经关注**  **if (event.equals("subscribe")) {**  **resultXml = messageToXml(fromUserName, toUserName, "哈哈,谢谢您的关注!!");**  **}**  **case "SCAN":**  **// 已经关注**  **String eventKey = result.get("EventKey");**  **if (eventKey.equals("123")) {**  **resultXml = messageToXml(fromUserName, toUserName, "您在内容存放的值为123");**  **}**  **default:**  **break;**  **}**  **out.println(resultXml);**  **out.close();**  **}** |

## 微信开发测试工具

https://mp.weixin.qq.com/wiki?t=resource/res\_main&id=mp1455784140

### JSSDK

<https://mp.weixin.qq.com/wiki?t=resource/res_main&id=mp1421141115>

### 分享功能

#### SignUtil验签

|  |
| --- |
| **public class** **SignUtil {**  **/\*\***  **\* 获得分享链接的签名。**  **\* @param ticket**  **\* @param nonceStr**  **\* @param timeStamp**  **\* @param url**  **\* @return**  **\* @throws Exception**  **\*/**  **public static String getSignature(String ticket, String nonceStr, String timeStamp, String url) throws Exception {**  **String sKey = "jsapi\_ticket=" + ticket**  **+ "&noncestr=" + nonceStr + "&timestamp=" + timeStamp**  **+ "&url=" + url;**  **System.*out*.println(sKey);**  **return *getSignature*(sKey);**  **}**  **/\*\***  **\* 验证签名。**  **\***  **\* @param signature**  **\* @param timestamp**  **\* @param nonce**  **\* @return**  **\*/**  **public static String getSignature(String sKey) throws Exception {**  **String ciphertext = null;**  **MessageDigest md = MessageDigest.*getInstance*("SHA-1");**  **byte[] digest = md.digest(sKey.toString().getBytes());**  **ciphertext = *byteToStr*(digest);**  **return ciphertext.toLowerCase();**  **}**  **/\*\***  **\* 将字节数组转换为十六进制字符串**  **\***  **\* @param byteArray**  **\* @return**  **\*/**  **private static String byteToStr(byte[] byteArray) {**  **String strDigest = "";**  **for (int i = 0; i < byteArray.length; i++) {**  **strDigest += *byteToHexStr*(byteArray[i]);**  **}**  **return strDigest;**  **}**  **/\*\***  **\* 将字节转换为十六进制字符串**  **\***  **\* @param mByte**  **\* @return**  **\*/**  **private static String byteToHexStr(byte mByte) {**  **char[] Digit = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F' };**  **char[] tempArr = new char[2];**  **tempArr[0] = Digit[(mByte >>> 4) & 0X0F];**  **tempArr[1] = Digit[mByte & 0X0F];**  **String s = new String(tempArr);**  **return s;**  **}**  **}** |

#### Java代码

|  |
| --- |
| **/\*\***  **\***  **\* 微信demo**  **\***  **\* @author Administrator**  **\***  **\*/**  **@Controller**  **public class IndexController {**  **private static final String *INDEX* = "index";**  **private static final String *SHARE* = "share";**  **@Autowired**  **private WxMpService wxMpService;**  **@Value("${wechat.mp.appId}")**  **private String appid;**  **@RequestMapping("/index")**  **public String index(HttpServletRequest req) throws Exception {**  **// 获取js临时票据**  **String jsapiTicket = wxMpService.getJsapiTicket();**  **// 获取时间戳**  **String noncestr = UUID.*randomUUID*().toString();**  **String timestamp = Long.*toString*(System.*currentTimeMillis*() / 1000);**  **//url**  **String url="http://yushengjun.s1.natapp.cc/index";**  **String signature = SignUtil.*getSignature*(jsapiTicket, noncestr, timestamp, url);**  **Map<String, Object> resultMap = new HashMap<>();**  **resultMap.put("jsapiTicket", jsapiTicket);**  **resultMap.put("url", url);**  **resultMap.put("noncestr", noncestr);**  **resultMap.put("timestamp", timestamp);**  **resultMap.put("signature", signature);**  **resultMap.put("appid", appid);**  **req.setAttribute("resultMap", resultMap);**  **return *INDEX*;**  **}**  **@RequestMapping("/share")**  **public String share() {**  **return *SHARE*;**  **}**  **}** |

#### 前端页面

|  |
| --- |
| **<!DOCTYPE html>**  **<head>**  **<title>微信js分享接口</title>**  **<meta charset=*"UTF-8"*>**  **</head>**  **<html>**  **<button class=*"btn btn\_primary"* id=*"onMenuShareTimeline"*>onMenuShareTimeline</button>**  **<script src=*"http://res.wx.qq.com/open/js/jweixin-1.0.0.js"*></script>**  **<script>**  **/\***  **\* 注意：**  **\* 1. 所有的JS接口只能在公众号绑定的域名下调用，公众号开发者需要先登录微信公众平台进入“公众号设置”的“功能设置”里填写“JS接口安全域名”。**  **\* 2. 如果发现在 Android 不能分享自定义内容，请到官网下载最新的包覆盖安装，Android 自定义分享接口需升级至 6.0.2.58 版本及以上。**  **\* 3. 常见问题及完整 JS-SDK 文档地址：http://mp.weixin.qq.com/wiki/7/aaa137b55fb2e0456bf8dd9148dd613f.html**  **\***  **\* 开发中遇到问题详见文档“附录5-常见错误及解决办法”解决，如仍未能解决可通过以下渠道反馈：**  **\* 邮箱地址：weixin-open@qq.com**  **\* 邮件主题：【微信JS-SDK反馈】具体问题**  **\* 邮件内容说明：用简明的语言描述问题所在，并交代清楚遇到该问题的场景，可附上截屏图片，微信团队会尽快处理你的反馈。**  **\*/**  **wx.config({**  **debug : true,**  **appId : '${resultMap['appid']}',**  **timestamp : '${resultMap['timestamp']}',**  **nonceStr : '${resultMap['noncestr']}',**  **signature : '${resultMap['signature']}',**  **jsApiList : [ 'checkJsApi', 'onMenuShareTimeline',**  **'onMenuShareAppMessage', 'onMenuShareQQ', 'onMenuShareWeibo',**  **'onMenuShareQZone', 'hideMenuItems', 'showMenuItems',**  **'hideAllNonBaseMenuItem', 'showAllNonBaseMenuItem',**  **'translateVoice', 'startRecord', 'stopRecord',**  **'onVoiceRecordEnd', 'playVoice', 'onVoicePlayEnd',**  **'pauseVoice', 'stopVoice', 'uploadVoice', 'downloadVoice',**  **'chooseImage', 'previewImage', 'uploadImage', 'downloadImage',**  **'getNetworkType', 'openLocation', 'getLocation',**  **'hideOptionMenu', 'showOptionMenu', 'closeWindow',**  **'scanQRCode', 'chooseWXPay', 'openProductSpecificView',**  **'addCard', 'chooseCard', 'openCard' ]**  **});**  **</script>**  **<script src=*"http://203.195.235.76/jssdk/js/zepto.min.js"*></script>**  **<script type=*"text/javascript"*>**  **wx.ready(function(){**  **alert('加载成功...');**  **// config信息验证后会执行ready方法，所有接口调用都必须在config接口获得结果之后，config是一个客户端的异步操作，所以如果需要在页面加载时就调用相关接口，则须把相关接口放在ready函数中调用来确保正确执行。对于用户触发时才调用的接口，则可以直接调用，不需要放在ready函数中。**    **wx.onMenuShareTimeline({**  **title: '正在给学员讲微信开发jssdk分享案例，请勿点击。',**  **desc: '该链接为教学链接，请勿点击...',**  **link: 'http://yushengjun.s1.natapp.cc/share',**  **imgUrl: 'http://demo.open.weixin.qq.com/jssdk/images/p2166127561.jpg',**  **trigger: function (res) {**  **// 不要尝试在trigger中使用ajax异步请求修改本次分享的内容，因为客户端分享操作是一个同步操作，这时候使用ajax的回包会还没有返回**  **alert('用户点击分享到朋友圈');**  **},**  **success: function (res) {**  **alert('已分享');**  **},**  **cancel: function (res) {**  **alert('已取消');**  **},**  **fail: function (res) {**  **alert(JSON.stringify(res));**  **}**  **});**  **alert('已注册获取“分享到朋友圈”状态事件');**    **});**  **wx.error(function(res){**  **alert(res.errMsg);**  **// config信息验证失败会执行error函数，如签名过期导致验证失败，具体错误信息可以打开config的debug模式查看，也可以在返回的res参数中查看，对于SPA可以在这里更新签名。**  **});**  **</script>**  **</html>** |

# 微信事件通知验证

https://mp.weixin.qq.com/wiki?t=resource/res\_main&id=mp1421135319

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请求携带参数如下表所示：

### CheckUtil

|  |
| --- |
| public class CheckUtil {  public static final String *tooken* = "itmayiedu"; // 开发者自行定义Tooken  public static boolean checkSignature(String signature, String timestamp, String nonce) {  // 1.定义数组存放tooken，timestamp,nonce  String[] arr = { *tooken*, timestamp, nonce };  // 2.对数组进行排序  Arrays.*sort*(arr);  // 3.生成字符串  StringBuffer sb = new StringBuffer();  for (String s : arr) {  sb.append(s);  }  // 4.sha1加密,网上均有现成代码  String temp = *getSha1*(sb.toString());  // 5.将加密后的字符串，与微信传来的加密签名比较，返回结果  return temp.equals(signature);  }  public static String getSha1(String str) {  if (str == null || str.length() == 0) {  return null;  }  char hexDigits[] = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f' };  try {  MessageDigest mdTemp = MessageDigest.*getInstance*("SHA1");  mdTemp.update(str.getBytes("UTF-8"));  byte[] md = mdTemp.digest();  int j = md.length;  char buf[] = new char[j \* 2];  int k = 0;  for (int i = 0; i < j; i++) {  byte byte0 = md[i];  buf[k++] = hexDigits[byte0 >>> 4 & 0xf];  buf[k++] = hexDigits[byte0 & 0xf];  }  return new String(buf);  } catch (Exception e) {  return null;  }  }  } |

### XmlUtils

|  |
| --- |
| **public class XmlUtils {**  **/\*\***  **\* 解析微信发来的请求（XML）**  **\***  **\* @param request**  **\* @return Map<String, String>**  **\* @throws Exception**  **\*/**  **@SuppressWarnings("unchecked")**  **public static Map<String, String> parseXml(HttpServletRequest request) throws Exception {**  **// 将解析结果存储在HashMap中**  **Map<String, String> map = new HashMap<String, String>();**  **// 从request中取得输入流**  **InputStream inputStream = request.getInputStream();**  **// 读取输入流**  **SAXReader reader = new SAXReader();**  **Document document = reader.read(inputStream);**  **// 得到xml根元素**  **Element root = document.getRootElement();**  **// 得到根元素的所有子节点**  **List<Element> elementList = root.elements();**  **// 遍历所有子节点**  **for (Element e : elementList)**  **map.put(e.getName(), e.getText());**  **// 释放资源**  **inputStream.close();**  **inputStream = null;**  **return map;**  **}**  **/\*\***  **\* 文本消息对象转换成xml**  **\***  **\* @param textMessage**  **\* 文本消息对象**  **\* @return xml**  **\*/**  **public static String messageToXml(TextMessage textMessage) {**  ***xstream*.alias("xml", textMessage.getClass());**  **return *xstream*.toXML(textMessage);**  **}**  **/\*\***  **\* 扩展xstream使其支持CDATA**  **\*/**  **private static XStream *xstream* = new XStream();**  **}** |

### DispatCherServlet

|  |
| --- |
| **/\*\***  **\* 微信事件通知**  **\***  **\* @author 余胜军**  **\***  **\*/**  **@RestController**  **@Slf4j**  **public class DispatCherServlet {**  **/\*\***  **\* 微信验证**  **\***  **\* @param signature**  **\* @param timestamp**  **\* @param nonce**  **\* @param echostr**  **\* @return**  **\*/**  **@RequestMapping(value = "/dispatCherServlet", method = RequestMethod.*GET*)**  **public String getDispatCherServlet(String signature, String timestamp, String nonce, String echostr) {**  **boolean checkSignature = CheckUtil.*checkSignature*(signature, timestamp, nonce);**  **if (!checkSignature) {**  **return null;**  **}**  **return echostr;**  **}**  **/\*\***  **\* 功能说明:微信事件通知**  **\***  **\* @param request**  **\* @return**  **\* @throws Exception**  **\*/**  **@RequestMapping(value = "/dispatCherServlet", method = RequestMethod.*POST*)**  **public void postdispatCherServlet(HttpServletRequest request, HttpServletResponse response) throws Exception {**  **request.setCharacterEncoding("UTF-8");**  **response.setCharacterEncoding("UTF-8");**  **Map<String, String> mapResult = XmlUtils.*parseXml*(request);**  **if (mapResult == null) {**  **return;**  **}**  **String msgType = mapResult.get("MsgType");**  **PrintWriter out = response.getWriter();**  **switch (msgType) {**  **case "text":**  **String content = mapResult.get("Content");**  **String toUserName = mapResult.get("ToUserName");**  **String fromUserName = mapResult.get("FromUserName");**  **String textMessage = null;**  **if (content.equals("蚂蚁课堂")) {**  **// 判断蚂蚁课堂**  **textMessage = setTextMessage("蚂蚁课堂 中国高端IT培训教学网站 www.itmayiedu.com", toUserName, fromUserName);**  **} else if (content.equals("你知道余胜军吗")) {**  **// 判断蚂蚁课堂**  **textMessage = setTextMessage("知道啊,余胜军,出生于1997年10月17日 在2016年7月15日创办了蚂蚁课堂教学网站，现任上海每特教育科技有限公司CEO",**  **toUserName, fromUserName);**  **} else if (content.equals("蚂蚁课堂第一期就业薪资")) {**  **// 判断蚂蚁课堂**  **textMessage = setTextMessage("平均薪资15k,哈哈哈 我也想去学习...", toUserName, fromUserName);**  **} else {**  **// 调用第三方智能接口**  **String resultStr = HttpClientUtil**  **.*doGet*("http://api.qingyunke.com/api.php?key=free&appid=0&msg=" + content);**  **JSONObject jsonObject = new JSONObject().*parseObject*(resultStr);**  **Integer integer = jsonObject.getInteger("result");**  **if (integer == null || integer != 0) {**  **textMessage = setTextMessage("亲,系统出错啦!", toUserName, fromUserName);**  **} else {**  **String result = jsonObject.getString("content");**  **textMessage = setTextMessage(result, toUserName, fromUserName);**  **}**  **}**  ***log*.info("postdispatCherServlet() info:{}", textMessage);**  **out.print(textMessage);**  **break;**  **default:**  **break;**  **}**  **out.close();**  **}**  **public String setTextMessage(String content, String toUserName, String fromUserName) {**  **TextMessage textMessage = new TextMessage();**  **textMessage.setCreateTime(new Date().getTime());**  **textMessage.setFromUserName(toUserName);**  **textMessage.setToUserName(fromUserName);**  **textMessage.setContent(content);**  **textMessage.setMsgType("text");**  **String messageToXml = XmlUtils.*messageToXml*(textMessage);**  **return messageToXml;**  **}**  **}** |

### JSONObject

|  |
| --- |
| **// 调用第三方智能接口**  **String resultStr = HttpClientUtil**  **.*doGet*("http://api.qingyunke.com/api.php?key=free&appid=0&msg=" + content);**  **JSONObject jsonObject = new JSONObject().*parseObject*(resultStr);**  **Integer integer = jsonObject.getInteger("result");**  **if (integer == null || integer != 0) {**  **textMessage = setTextMessage("亲,系统出错啦!", toUserName, fromUserName);**  **} else {**  **String result = jsonObject.getString("content");**  **textMessage = setTextMessage(result, toUserName, fromUserName);**  **}** |

### HttpClientUtil

|  |
| --- |
| import java.io.IOException;  import java.net.URI;  import java.util.ArrayList;  import java.util.List;  import java.util.Map;  import org.apache.http.NameValuePair;  import org.apache.http.client.entity.UrlEncodedFormEntity;  import org.apache.http.client.methods.CloseableHttpResponse;  import org.apache.http.client.methods.HttpGet;  import org.apache.http.client.methods.HttpPost;  import org.apache.http.client.utils.URIBuilder;  import org.apache.http.entity.ContentType;  import org.apache.http.entity.StringEntity;  import org.apache.http.impl.client.CloseableHttpClient;  import org.apache.http.impl.client.HttpClients;  import org.apache.http.message.BasicNameValuePair;  import org.apache.http.util.EntityUtils;  public class HttpClientUtil {  public static String doGet(String url, Map<String, String> param) {  // 创建Httpclient对象  CloseableHttpClient httpclient = HttpClients.createDefault();  String resultString = "";  CloseableHttpResponse response = null;  try {  // 创建uri  URIBuilder builder = new URIBuilder(url);  if (param != null) {  for (String key : param.keySet()) {  builder.addParameter(key, param.get(key));  }  }  URI uri = builder.build();  // 创建http GET请求  HttpGet httpGet = new HttpGet(uri);  // 执行请求  response = httpclient.execute(httpGet);  // 判断返回状态是否为200  if (response.getStatusLine().getStatusCode() == 200) {  resultString = EntityUtils.toString(response.getEntity(), "UTF-8");  }  } catch (Exception e) {  e.printStackTrace();  } finally {  try {  if (response != null) {  response.close();  }  httpclient.close();  } catch (IOException e) {  e.printStackTrace();  }  }  return resultString;  }  public static String doGet(String url) {  return doGet(url, null);  }  public static String doPost(String url, Map<String, String> param) {  // 创建Httpclient对象  CloseableHttpClient httpClient = HttpClients.createDefault();  CloseableHttpResponse response = null;  String resultString = "";  try {  // 创建Http Post请求  HttpPost httpPost = new HttpPost(url);  // 创建参数列表  if (param != null) {  List<NameValuePair> paramList = new ArrayList<>();  for (String key : param.keySet()) {  paramList.add(new BasicNameValuePair(key, param.get(key)));  }  // 模拟表单  UrlEncodedFormEntity entity = new UrlEncodedFormEntity(paramList);  httpPost.setEntity(entity);  }  // 执行http请求  response = httpClient.execute(httpPost);  resultString = EntityUtils.toString(response.getEntity(), "utf-8");  } catch (Exception e) {  e.printStackTrace();  } finally {  try {  response.close();  } catch (IOException e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  }  return resultString;  }  public static String doPost(String url) {  return doPost(url, null);  }    public static String doPostJson(String url, String json) {  // 创建Httpclient对象  CloseableHttpClient httpClient = HttpClients.createDefault();  CloseableHttpResponse response = null;  String resultString = "";  try {  // 创建Http Post请求  HttpPost httpPost = new HttpPost(url);  // 创建请求内容  StringEntity entity = new StringEntity(json, ContentType.APPLICATION\_JSON);  httpPost.setEntity(entity);  // 执行http请求  response = httpClient.execute(httpPost);  resultString = EntityUtils.toString(response.getEntity(), "utf-8");  } catch (Exception e) {  e.printStackTrace();  } finally {  try {  response.close();  } catch (IOException e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  }  return resultString;  }  } |

### 青云客智能聊天机器人API

http://api.qingyunke.com/

### 微信开发架构流程