目录

目录

[目录 1](#_Toc525141581)

[微信公众号 1](#_Toc525141582)

[一 公众号分类 1](#_Toc525141583)

[二 开发前配置 1](#_Toc525141584)

[三 获取access\_token 3](#_Toc525141585)

[四消息接收与响应处理 3](#_Toc525141586)

[五微信网页授权 5](#_Toc525141587)

微信公众号

## 一 公众号分类

（1）订阅号

为媒体和个人提供一种信息传播方式，主要偏于为用户传送资讯，主要是阅读，每天可以群发1条消息。

（2）服务号

偏于服务交互（类似银行提供服务查询），每月只可群发4条消息

（3）企业微信（企业号）

主要用于公司内部通讯使用，需有成员通讯信息验证才可以关注成功企业微信。

## 二 开发前配置

（1）注册微信公众号

（2）接口配置信息

url（外网地址） 和 token配置需要效验才能提交成功。

后台通过检验signature对请求进行效验。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

后台效验步骤java代码事例

（1）url访问的效验方法地址，且地址必须是外网地址。

（2）后台建立方法验证

//get 方法验证

@RequestMapping(value=“/weChat”,method=RequestMethod.GET)

@ResponseBody

例子：public class WechatController{

public String validate(String signature,string timestamp,string nonce, echostr ){

//1将token、timestamp、nonce三个参数进行字典序排序

String[] arr=[wecharUtil, token, timestamp, nonce]

Arrays.sort(arr)

//2将三个参数字符串拼接成一个字符串进行sha1加密

Stringbuilder sb =new StringBuilder();

for (String temp:arr){

sb.appead(temp);

}

//自己加密的签名

String mysignsture = securityUtil.SMA1(sb.toString());

// 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

if(mysignsture.equals(signature)){

//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，

System.out.println(“接入成功”)

return echostr

}

System.out.println(“接入失败”)

return null

}

}

## 三 获取access\_token

（1）公众平台以access\_token 为接口调用凭据，用来调用接口，所有接口的调用需要先获取access\_token.

(2)公众平台接口调用仅支持80端口

注: **调用接口时，请登录“微信公众平台-开发-基本配置”提前将服务器IP地址添加到IP白名单中，点击查看设置方法，否则将无法调用成功。**小程序无需配置IP白名单。

https请求方式：GET

http接口地址：https://api.weixin.qq.com/cgi-bin/token?grant\_type=client\_credential&appid=APPID&secret=APPSECRET

| **参数** | **是否必须** | **说明** |
| --- | --- | --- |
| grant\_type | 是 | 获取access\_token填写client\_credential |
| appid | 是 | 第三方用户唯一凭证 |
| secret | 是 | 第三方用户唯一凭证密钥，即appsecret |

ip白名单配置：配置所调用接口的ip地址

## 四消息接收与响应处理

后台添加处理消息的方法（java）

//POST消息的XML数据包到开发者填写的URL上。

//注：与配置验证的方法**请求路径相同，当方法不同，一个GET，一个POST.**

/\*

\* 创建一个封装消息的实体类，把所有可接收到的参数都放入(具体接收参数： eg:文本类型参数如下[其他类型参考微信公众号开发文档https://mp.weixin.qq.com/wiki?t=resource/res\_main&id=mp1421140453]

| **参数** | **描述** |
| --- | --- |
| ToUserName | 开发者微信号 |
| FromUserName | 发送方帐号（一个OpenID） |
| CreateTime | 消息创建时间 （整型） |
| MsgType | text |
| Content | 文本消息内容 |
| MsgId | 消息id，64位整型 |

)\*/

/\* 开始接收微信传过来的xml数据步骤：

\* **第1步**：在handleMessage方法的形参上添加InMsgEntity类型的参数，并且贴上@RequestBody注解

\*/

@RequestMapping(value=“/weChat”,method=RequestMethod.POST)

@ResponseBody

public Object handleMessage(@ResponseBody InMsgEntity msg){

/\*@RequestBody 该注解用于读取request请求的body部分数据，根据Content-Type来判断把数据当做什么类型来解析，然后把相应的数据绑定到参数上。\*/

/\* 被动回复消息

\*（1）建立OutMsgEntity 实体类并且把ToUserName与FormUserName的值设置为相反即可。\*/

//创建消息响应对象

OutMsgEntity out = new OutMsgEntity()

//把原来的接收方设置为发送方

out.setFromUserName(msg.getToUserName());

//获取接收的消息类型

String msgType = msg.getMsgType();

//设置消息的响应类型

out.setMsgType(msgType);

//设置消息创建时间

out.setCreateTime(new Date().getTime());

//根据类型设置不同的消息数据

if("text".equals(msgType)){

out.setContent(msg.getContent());

}else if("image".equals(msgType)){

out.setMediaId(new String[]{msg.getMediaId()});

}

return out;

}

## 五微信网页授权

注：获取用户信息，需要用到网页授权。

（1）授权前，开发者需要先到公众平台官网中的“开发 - 接口权限 - 网页服务 - 网页帐号 - 网页授权获取用户基本信息”的配置选项中，修改授权回调域名。请注意，这里填写的是域名（是一个字符串），而不是URL，因此请勿加 http:// 等协议头；

（2）网页授权的两种scope区别

2.1 、以snsapi\_base为scope发起的网页授权，是用来获取进入页面的用户的openid的，并且是静默授权并自动跳转到回调页的。

2.2、以snsapi\_userinfo为scope发起的网页授权，是用来获取用户的基本信息的。但这种授权需要**用户手动同意，并且由于用户同意过**，所以无须关注，就可在授权后获取该用户的基本信息。

2.3、用户管理类接口中的“获取用户基本信息接口”，是在用户和公众号产生消息交互或关注后事件推送后，才能根据用户OpenID来获取用户基本信息。这个接口，包括其他微信接口，都是需要该用户（即openid）**关注了公众号后**，才能调用成功的。

（3）网页授权流程分为四步：

3.1、引导用户进入授权页面同意授权，**获取code**

在确保微信公众账号拥有授权作用域（scope参数）的权限的前提下（服务号获得高级接口后，默认拥有scope参数中的snsapi\_base和snsapi\_userinfo），引导关注者打开如下页面：https://open.weixin.qq.com/connect/oauth2/authorize?appid=APPID&redirect\_uri=REDIRECT\_URI&response\_type=code&scope=SCOPE&state=STATE#wechat\_redirect

参数说明

| **参数** | **是否必须** | **说明** |
| --- | --- | --- |
| appid | 是 | 公众号的唯一标识 |
| redirect\_uri | 是 | 授权后重定向的回调链接地址， 请使用 urlEncode 对链接进行处理 |
| response\_type | 是 | 返回类型，请填写code |
| scope | 是 | 应用授权作用域，snsapi\_base （不弹出授权页面，直接跳转，只能获取用户openid），snsapi\_userinfo （弹出授权页面，可通过openid拿到昵称、性别、所在地。并且， 即使在未关注的情况下，只要用户授权，也能获取其信息 ） |
| state | 否 | 重定向后会带上state参数，开发者可以填写a-zA-Z0-9的参数值，最多128字节 |
| #wechat\_redirect | 是 | 无论直接打开还是做页面302重定向时候，必须带此参数 |

**用户同意授权后**

如果用户同意授权，页面将跳转至 redirect\_uri/?code=CODE&state=STATE。

code说明： code作为换取access\_token的票据，每次用户授权带上的code将不一样，code只能使用一次，5分钟未被使用自动过期。

3.2、通过code换取网页授权**access\_token**（与基础支持中的access\_token不同）

注意：由于公众号的secret和获取到的access\_token安全级别都非常高，必须只保存在服务器，不允许传给客户端。后续刷新access\_token、通过access\_token获取用户信息等步骤，也必须从服务器发起。

请求方法：

获取code后，请求以下链接获取access\_token： https://api.weixin.qq.com/sns/oauth2/access\_token?appid=APPID&secret=SECRET&code=CODE&grant\_type=authorization\_code

参数说明

| **参数** | **是否必须** | **说明** |
| --- | --- | --- |
| appid | 是 | 公众号的唯一标识 |
| secret | 是 | 公众号的appsecret |
| code | 是 | 填写第一步获取的code参数 |
| grant\_type | 是 | 填写为authorization\_code |

正确时返回的JSON数据包如下：

{ "access\_token":"ACCESS\_TOKEN", //网页授权接口调用凭证,注意  
"expires\_in":7200, // access\_token接口调用凭证超时时间，单位（秒）  
"refresh\_token":"REFRESH\_TOKEN", //用户刷新access\_token  
"openid":"OPENID", //用户唯一标识，请注意，在未关注公众号时，用户访问公众号的网页，也会产生一个用户和公众号唯一的OpenID  
"scope":"SCOPE" //用户授权的作用域，使用逗号（,）分隔}

3.3、如果需要，开发者可以刷新网页授权access\_token，避免过期

由于access\_token拥有较短的有效期，当access\_token超时后，可以使用refresh\_token进行刷新，**refresh\_token**有效期为30天，当refresh\_token失效之后，需要用户重新授权。

**请求方法**

获取第二步的refresh\_token后，请求以下链接获取access\_token：  
https://api.weixin.qq.com/sns/oauth2/refresh\_token?appid=APPID&grant\_type=refresh\_token&refresh\_token=REFRESH\_TOKEN

| **参数** | **是否必须** | **说明** |
| --- | --- | --- |
| appid | 是 | 公众号的唯一标识 |
| grant\_type | 是 | 填写为refresh\_token |
| refresh\_token | 是 | 填写通过access\_token获取到的refresh\_token参数 |

返回说明

正确时返回的JSON数据包如下：

{ "access\_token":"ACCESS\_TOKEN",  
"expires\_in":7200,  
"refresh\_token":"REFRESH\_TOKEN",  
"**openid**":"OPENID",  
"scope":"SCOPE" }

3.4、通过网页授权access\_token和openid获取用户基本信息（支持UnionID机制）

如果网页授权作用域为snsapi\_userinfo，则此时开发者可以通过access\_token和openid拉取用户信息了。

请求方法

http：GET（请使用https协议） https://api.weixin.qq.com/sns/userinfo?access\_token=ACCESS\_TOKEN&openid=OPENID&lang=zh\_CN

参数说明

| **参数** | **描述** |
| --- | --- |
| access\_token | 网页授权接口调用凭证,注意：此access\_token与基础支持的access\_token不同 |
| openid | 用户的唯一标识 |
| lang | 返回国家地区语言版本，zh\_CN 简体，zh\_TW 繁体，en 英语 |

返回说明

正确时返回的JSON数据包如下：

{ "openid":" OPENID",  
" nickname": NICKNAME,  
"sex":"1",  
"province":"PROVINCE"  
"city":"CITY",  
"country":"COUNTRY",  
"headimgurl": "http://thirdwx.qlogo.cn/mmopen/g3MonUZtNHkdmzicIlibx6iaFqAc56vxLSUfpb6n5WKSYVY0ChQKkiaJSgQ1dZuTOgvLLrhJbERQQ4eMsv84eavHiaiceqxibJxCfHe/46",  
"privilege":[ "PRIVILEGE1" "PRIVILEGE2" ],  
"unionid": "o6\_bmasdasdsad6\_2sgVt7hMZOPfL"  
}

## 六JSSDK使用步骤

6.1 绑定域名

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”。

6.2 引入js文件（在html页面引入）

在需要调用JS接口的页面引入如下JS文件，（支持https）：<http://res.wx.qq.com/open/js/jweixin-1.4.0.js>

6.3 通过 config接口注入权限验证配置（在html<script>中写入配置）

wx.config({  
 debug: true, // 开启调试模式,调用的所有api的返回值会在客户端alert出来，若要查看传入的参数，可以在pc端打开，参数信息会通过log打出，仅在pc端时才会打印。  
 appId: '', // 必填，公众号的唯一标识  
 timestamp: , // 必填，生成签名的时间戳 //测试可随机填写，否则需后台提供  
 nonceStr: '', // 必填，生成签名的随机串 //测试可随机填写，否则需后台提供  
 signature: '',// 必填，签名 //后台根据前面几项计算签名  
 jsApiList: [' updateTimelineShareData '] //eg:分享接口 必填，需要使用的JS接口列表  
});

**自定义“分享到朋友圈”及“分享到QQ空间”按钮的分享内容**

wx.ready(function(){  
 // 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.ready(function () { //需在用户可能点击分享按钮前就先调用  
 wx.updateTimelineShareData({   
 title: '', // 分享标题  
 link: '', // 分享链接，该链接域名或路径必须与当前页面对应的公众号JS安全域名一致  
 imgUrl: '', // 分享图标  
 }, function(res) {   
 //这里是回调函数

alert(“分享成功”)  
 });   
});