# 微信授权登录并支付总结

概述：微信第三方授权登录，授权后转向h5页面直接发起支付，应用场景：智能售货机，用户扫码直接支付并记录用户信息。

## 业务配置项

### 1.1注册微信公众号，微信商户号

微信公众号需要认证，认证后才会有相关开发接口权限

### 1.2开通微信支付权限

登录商户平台，将商户号和公众号绑定，产品中心-》appID授权管理

### 1.3微信公众号服务器配置

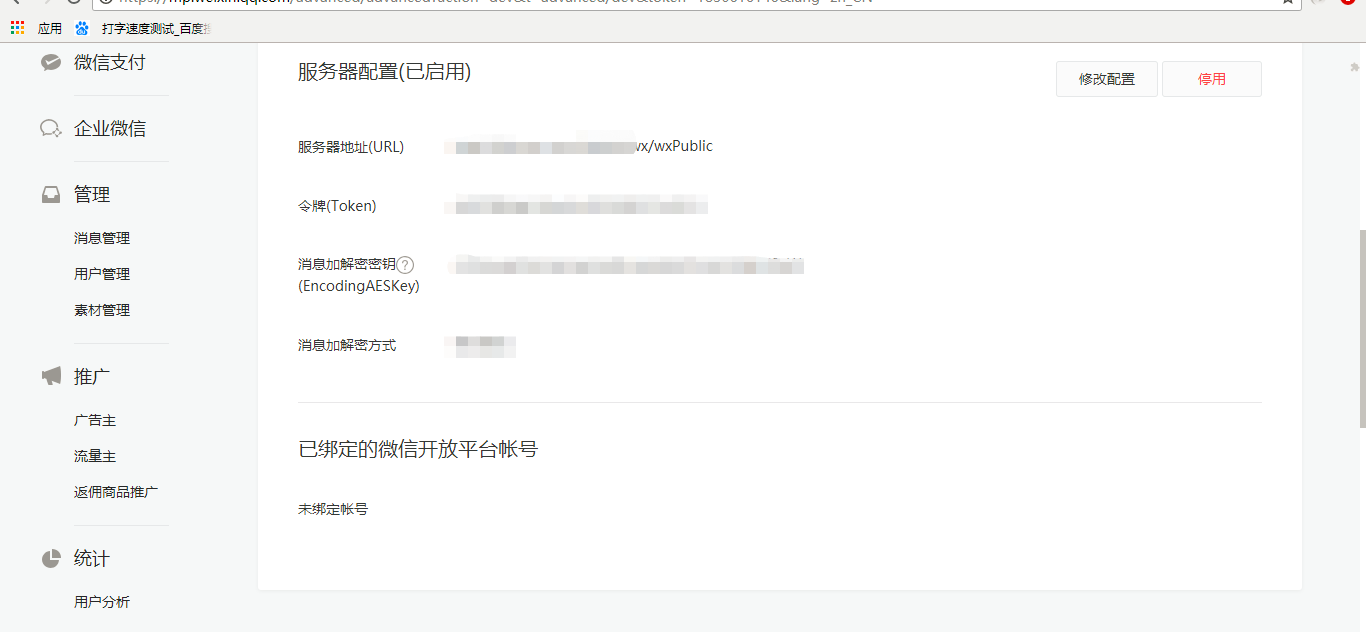
登录微信公众平台，配置服务器信息，注意服务器地址直接配置到请求方法的路径

此项配置为了验证服务器的有效性，微信服务器会发送签名信息到我们服务器，我们收到签名信息后进行校验，如果通过将echostr原样输出，则通过校验。

微信服务器发送签名信息包含：signature，timestamp,nonce,echostr

将timestamp,nonce和配置中的token进行字典排序，排序后结果进行sha1加密得到签名，

将生成端的签名与signature比较，相同则验证签名通过，输出echostr



### 1.4配置网页授权域名

登录微信公众平台，开发-》接口权限-》网页授权-》修改

### 1.5配置jsapi安全域名

登录微信公众平台，开发-》接口权限-》网页授权-》修改

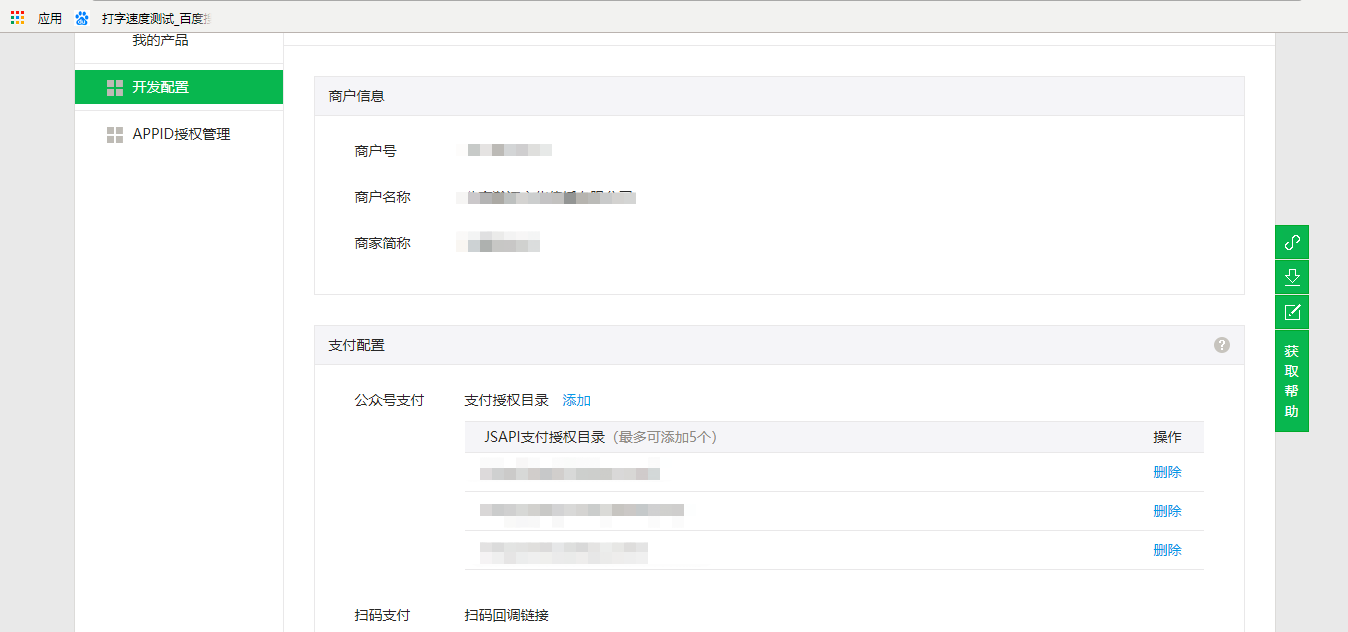
### 1.6配置支付安全目录

登录微信商户平台，产品中心-》开发配置

注意这有个坑，设置目录要设置到当前访问连接的上一级，举个栗子

访问url为：http://www.abc.com/recharge/index

授权目录应为：http://www.abc.com/recharge/



## 开发流程

网页授权官方文档地址：

<https://mp.weixin.qq.com/wiki?t=resource/res_main&id=mp1421140842>

获取用户信息可参照如上文档，这个描述还是比较清晰

Jssdk官方文档地址：

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

这个文档里面就有几个比较坑的地方

1. jssdk发起支付的时候，需要后端生成两个签名，一个是配置签名，一个是支付签名
2. 加密签名的时候时间戳要指定到秒
3. Jsticket需要缓存起来，说是频繁调用会影响使用，但是也没有说次数限制是多少

### 2.1 获取用户授权code

引导用户在微信客户端中打开这个链接

<https://open.weixin.qq.com/connect/oauth2/authorize?appid=wxf0e81c3bee622d60&redirect_uri=http%3A%2F%2Fnba.bluewebgame.com%2Foauth_response.php&response_type=code&scope=snsapi_userinfo&state=STATE#wechat_redirect>

这个链接的几个参数说明

|  |  |  |
| --- | --- | --- |
| 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 |

### 2.2获取accessToken

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

### 2.3获取微信用户信息

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。

https://api.weixin.qq.com/sns/userinfo?access\_token=ACCESS\_TOKEN&openid=OPENID&lang=zh\_CN

### 2.4生成预支付订单

调用微信统一下单接口生成预支付订单，注意这个地方要选择jsapi支付的文档

<https://pay.weixin.qq.com/wiki/doc/api/jsapi.php?chapter=9_1>

### 2.5获取jsticket

api\_ticket 是用于调用微信卡券JS API的临时票据，有效期为7200 秒，通过access\_token 来获取。

开发者注意事项：

1.此用于卡券接口签名的api\_ticket与步骤三中通过config接口注入权限验证配置使用的jsapi\_ticket不同。

2.由于获取api\_ticket 的api 调用次数非常有限，频繁刷新api\_ticket 会导致api调用受限，影响自身业务，开发者需在自己的服务存储与更新api\_ticket。

****接口调用请求说明****

http请求方式: GET  
https://api.weixin.qq.com/cgi-bin/ticket/getticket?access\_token=ACCESS\_TOKEN&type=wx\_card

### 2.6生成jssdk签名

Map<String,Object> configSignMap = **new** HashMap<String,Object>();

String jsTicket = jsTicketJson.getString("ticket");

String url = WxConstants.***JS\_SDK\_CONFIG\_URL***;

configSignMap.put("jsapi\_ticket", jsTicket);

configSignMap.put("timestamp", timestamp);

configSignMap.put("url", url);

configSignMap.put("noncestr", noncestr);//WeiXinPayUtil.getRandomString()

String configSign = WxPublicAccountUtil.*getSignature*(configSignMap);

### 2.7生成支付签名

//生成支付签名 paySign 需要加密的参数 appId, timeStamp, nonceStr, package, signType

Map<String,String> signMap = **new** HashMap<String,String>();

signMap.put("nonceStr", noncestr);

signMap.put("timeStamp", timestamp);

signMap.put("appId", WxConstants.***WX\_PUBLIC\_ACCOUNT\_APPID***);

signMap.put("package", "prepay\_id="+prepay\_id);

signMap.put("signType", "MD5");

String paySign = WeiXinPayUtil.*getSign*(signMap);

### 2.8通过jssdk调起支付

$(**function**(){

**var** appId = $("#appId").val();

**var** nonceStr = $("#nonceStr").val();

**var** timeStamp = $("#timeStamp").val();

**var** prepay\_id = $("#prepay\_id").val();

**var** paySign = $("#paySign").val();

**var** configSign = $("#configSign").val();

**var** pack\_age = "prepay\_id="+prepay\_id;

wx.config({

        debug: **false**, // 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: appId, // 必填，公众号的唯一标识

        timestamp:timeStamp , // 必填，生成签名的时间戳

        nonceStr: nonceStr, // 必填，生成签名的随机串

        signature: configSign,// 必填，签名，见附录1

        jsApiList: ['chooseWXPay'] // 必填，需要使用的JS接口列表，所有JS接口列表见附录2，如果只是支付，只用这一个参数就够了

});

wx.error(**function**(res){

// 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。

});

wx.ready(**function**(){

    wx.chooseWXPay({

        timestamp:timeStamp, // 支付签名时间戳，注意微信jssdk中的所有使用timestamp字段均为小写。但最新版的支付后台生成签名使用的timeStamp字段名需大写其中的S字符

        nonceStr:nonceStr, // 支付签名随机串，不长于 32 位

**package**: pack\_age, // 统一支付接口返回的prepay\_id参数值，提交格式如：prepay\_id=\*\*\*）

        signType: 'MD5', // 签名方式，默认为'SHA1'，使用新版支付需传入'MD5'

        paySign: paySign, // 支付签名

        success: **function** (res) {

            // 支付成功后的回调函数

WeixinJSBridge.call('closeWindow');

        },

        //如果你按照正常的jQuery逻辑，下面如果发送错误，一定是error，那你就太天真了，当然，jssdk文档中也有提到

        fail: **function**(res) {

            //接口调用失败时执行的回调函数。

        },

        complete: **function**(res) {

            //接口调用完成时执行的回调函数，无论成功或失败都会执行。

        },

        cancel: **function**(res) {

            //用户点击取消时的回调函数，仅部分有用户取消操作的api才会用到。

        },

        trigger: **function**(res) {

            //监听Menu中的按钮点击时触发的方法，该方法仅支持Menu中的相关接口。

        }

    });

});

});