支付宝小程序/生活号支付开发流程

### 支付宝开发平台配置

* 1. 设置支付目录登录开发平台

<https://docs.open.alipay.com/399/106843/>



1.2 创建网页应用



1.3配置





注：授权回调地址，是支付宝用户授权后回调的地址

### 支付宝支付

**流程：用户授权，获取用户openId->调汇付支付接口获取支付宝payInfo->唤起支付宝支付**

#### 用户授权获取用户id

1. 按照规则拼接授权页面的链接，并且引导用户跳转至该链接；

2. 用户在授权页面上确认授权后，将跳转到开发者指定的回调页，并且带上auth\_code；

3. 开发者通过auth\_code换取access\_token及用户的userId；

支付宝文档可见：<https://docs.open.alipay.com/220/105337/>

**第一步：URL拼接与scope详解**

url拼接规则：https://openauth.alipay.com/oauth2/publicAppAuthorize.htm?app\_id=APPID&scope=SCOPE&redirect\_uri=ENCODED\_URL

使用场景举例：开发者通过URL拼接方案，构造授权页面，并且引导用户授权。

url参数说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **是否必须** | **描述** |
| app\_id | 是 | 开发者应用的app\_id |
| scope | 是 | 此处固定填auth\_base字符串 |
| redirect\_uri | 是 | 回调页面，是 经过转义 的url链接（url必须以http或者https开头），比如：http%3A%2F%2Fexample.com注：在请求之前，开发者需要先到开发者中心对应应用内，配置授权回调地址。 |
| state | 否 | 商户自定义参数，用户授权后，重定向到redirect\_uri时会原样回传给商户。 为防止CSRF攻击，建议开发者请求授权时传入state参数，该参数要做到既不可预测，又可以证明客户端和当前第三方网站的登录认证状态存在关联。 |

**关于redirect\_uri的说明：**
接口会校验授权链接中配置的redirect\_uri与应用中配置的授权链接是否一致。详细说明： 如果开发者在应用中配置的授权链接是：https://auth.example.com/authCallBack，则redirect\_uri内容为https://auth.example.com/authCallBack的encode形式https%3A%2F%2Fauth.example.com%2FauthCallBack。授权回调地址对应的域名（auth.example.com）下的页面http://auth.example.com/authCallBack、https://auth.example.com/authRedirect、https://auth.example.com/都可以进行OAuth2.0授权。但与（auth.example.com）关联的二三级域名，如：http://www.example.com/、http://example.com/无法进行OAuth2.0授权。

**第二步：获取auth\_code**

当用户授权成功后，会跳转至开发者定义的回调页面，支付宝会在回调页面请求中加入参数，包括auth\_code、app\_id、scope等，需要注意的是支付宝仅保证auth\_code、app\_id以及scope参数的有效性。支付宝请求开发者回调页面示例如下：

http://example.com/doc/toAuthPage.html?app\_id=2014101500013658&source=alipay\_wallet&scope=auth\_userinfo&auth\_code=ca34ea491e7146cc87d25fca24c4cD11

第三步：使用auth\_code换取接口access\_token及用户userId

接口名称：alipay.system.oauth.token

换取授权访问令牌，开发者可通过获取到的auth\_code换取access\_token和用户userId。auth\_code作为换取access\_token的票据，每次用户授权完成，回调地址中的auth\_code将不一样，auth\_code只能使用一次，一天未被使用自动过期。

接口请求示例

REQUEST URL: https://openapi.alipay.com/gateway.do

REQUEST METHOD: POST

CONTENT:

    app\_id=2014070100171525

    method=alipay.system.oauth.token

    charset=GBK

    sign\_type=RSA2

    timestamp=2014-01-01 08:08:08

    sign=rXaTEfJ7WTDsP1DWRPHARW3uOr19+fzlngMCJBvbhP1XPEa9qZwGGng9oMDloABpJMT2SGeOj46+BUkqCGRO9fH90Vci3hOH01BfYnbhJz3ADK2h7gpjlponx4/sxELN6f2GXi51XKiHKnxMA9XpLLo68q+roY0M/ZFQ1UdnqeM=

    version=1.0

    grant\_type=authorization\_code

    code=4b203fe6c11548bcabd8da5bb087a83b

    refresh\_token=201208134b203fe6c11548bcabd8da5bb087a83b

接口调用示例

AlipayClient alipayClient =new DefaultAlipayClient("https://openapi.alipay.com/gateway.do",APP\_ID,APP\_PRIVATE\_KEY,"json","GBK",ALIPAY\_PUBLIC\_KEY);

AlipaySystemOauthTokenRequest request = new AlipaySystemOauthTokenRequest();

request.setCode("2e4248c2f50b4653bf18ecee3466UC18");

request.setGrantType("authorization\_code");

try {

 AlipaySystemOauthTokenResponse oauthTokenResponse = alipayClient.execute(request);

 System.out.println(oauthTokenResponse.getAccessToken());

} catch (AlipayApiException e) {

 //处理异常

 e.printStackTrace();

}

请求参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **参数** | **参数名称** | **类型（长度范围）** | **参数说明** | **是否可为空** | **样例** |
| grant\_type | 授权类型 | String | 值为authorization\_code时，代表用code换取；值为refresh\_token时，代表用refresh\_token换取 | 不可空 | authorization\_code |
| code | 授权码 | String | 用户对应用授权后得到，即第二步中开发者获取到的auth\_code值 | 与refresh\_token二选一 | 4b203fe6c11548bcabd8da5bb087a83b |
| refresh\_token | 刷新令牌 | String | 刷新access\_token时使用 | 与code二选一 | 201208134b203fe6c11548bcabd8da5bb087a83b |

同步响应结果示例

{

    "alipay\_system\_oauth\_token\_response": {

        "access\_token": "publicpBa869cad0990e4e17a57ecf7c5469a4b2",

        "user\_id": "2088411964574197",

        "alipay\_user\_id": "20881007434917916336963360919773",

        "expires\_in": 300,

        "re\_expires\_in": 300,

        "refresh\_token": "publicpB0ff17e364f0743c79b0b0d7f55e20bfc"

    },

    "sign": "xDffQVBBelDiY/FdJi4/a2iQV1I7TgKDFf/9BUCe6+l1UB55YDOdlCAir8CGlTfa0zLYdX0UaYAa43zY2jLhCTDG+d6EjhCBWsNY74yTdiM95kTNsREgAt4PkOkpsbyZVXdLIShxLFAqI49GIv82J3YtzBcVDDdDeqFcUhfasII="

}

同步响应参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **参数** | **参数名称** | **类型（长度范围）** | **参数说明** | **是否可为空** | **样例** |
| access\_token | 交换令牌 | String | 用于获取用户信息 | 不可空 | publicpBa869cad0990e4e17a57ecf7c5469a4b2 |
| user\_id | 用户的userId | String | 支付宝用户的唯一userId | 不可空 | 2.08841E+15 |
| expires\_in | 令牌有效期 | Number | 交换令牌的有效期，单位秒 | 不可空 | 300 |
| re\_expires\_in | 刷新令牌有效期 | Number | 刷新令牌有效期，单位秒 | 不可空 | 300 |
| refresh\_token | 刷新令牌 | String | 通过该令牌可以刷新access\_token | 不可空 | publicpB0ff17e364f0743c79b0b0d7f55e20bfc |

#### 调用支付管家app支付

发起支付管家app支付接口请求，接口详情参考api文档： <https://api.cloudpnr.com/hfpay/wxAliPay.html#app>

注意参数：

app\_id = 开发者应用的app\_id

buyer\_id = 上一步骤取得的openid

app\_pay\_type = 06（支付宝小程序/生活号）

获取接口返回的pay\_info字段

#### JS唤起支付宝界面

js初始化支付宝浏览器的AlipayJSBridge对象，唤起支付宝支付界面，并定义不同返回码的事件。

例：

|  |
| --- |
| */\*\* \* 确保支付宝JS已经加载完成 \** ***@param*** *callback \*/***function** *alipayReady*(callback) { **if** (**window**.AlipayJSBridge) { callback && callback(); } **else** { **document**.addEventListener(**'AlipayJSBridgeReady'**, callback, **false**); }}*/\*\* \* JS 调起支付宝支付 \** ***@param*** *payInfo \*/***function** *alipayPay*(payInfo) { *alipayReady*(**function** () { **var** jsonMsg = payInfo.replace(/\"/g, **"\""**); **var** payInfoJson = ***JSON***.parse(jsonMsg); AlipayJSBridge.call(**'tradePay'**, payInfoJson, **function** (result) { **var** resultCode = result[**'resultCode'**]; **switch** (resultCode) { **case "9000"**: *// 订单支付成功 paySuccess*(); **case "8000"**: *// 正在处理中 queryPayStat*(); **break**; **case "6002"**: *// 网络连接出错 showMiniToast*(**"网络出了点问题"**) **break**; **case "6001"**: *// 用户中途取消* **case "99"**: *// 用户点击忘记密码导致快捷界面退出(only iOS) $*(**'#toAuthCodeUrl'**).remove(); ***selectedTransType*** = 22; **break**; **case "4000"**: *// 订单支付失败* **default**: *showMiniToast*(**"支付宝支付失败！"**, 5000); **break**; } }); });} |

注：支付成功后的回调页 为 支付管家app支付接口请求时传入的ret\_url。