腾讯实名核身小程序联调接口说明文档

接口说明

小程序入口

appid：业务方提供微信小程序的appid(微信公众平台上获得)，用以与公众号关联后可以跳转至腾讯实名核身小程序。

1) 参数说明

参数名称	要求	类型	说明
signature	必选	String	接口签名，具体见签名算法(由用户后台生成)
uid	必选	String	传业务方的用户openid
appid	必选	String	腾讯实名核身分配的appid

2) 跳转代码示例

```
wx.navigateToMiniProgram({
  appId: 'wx123cc22c23ed2d26',
  path: 'page/component/pages/index/index',
  extraData: {
    signature: 'xxx',
    uid: 'xxx',
    appid: 'xxx'
  },
  envVersion: 'trial', // 用体验版小程序来联调，上线后改为正式版
  success(res) {
    // 返回成功
  },fail(err) {
    // 返回失败
  }
})
```

小程序出口

1) 参数说明

参数名称	要求	类型	说明
token	必选	String	auth接口返回的token序列号(拿到token后可以调用1.3.的接口去服务器拉取数据)
errCode	必选	String	流程返回的错误码（详情参考3.错误码）

2) 跳转代码示意

```
wx.navigateBackMiniProgram({
  extraData: {
    token: 'xxx'
  },
  success(res) {
    // 返回成功
  },
  fail(err) {
    // 返回失败
  }
})
```

实名信息拉取接口

1) 接口

测试平台接口：https://iauth-test.wecity.qq.com/new/cgi-bin/getdetectinfo.php

2) 描述

拉取实名详细信息接口。

3) 方法

POST

4) HTTP请求格式

a、头部信息

要求	参数名称	类型	说明
signature	是	String	接口签名，具体见签名算法

b、请求包体JSON格式

要求	参数名称	类型	说明
必选	token	String	微信小程序返回的token序列号
必选	appid	String	分配的appid
可选	info_type	Int	获取信息类型；不传时,默认带上所有文件buffer；传"0"表示获取所有信息，含文件buffer；"1"为传文本信息，不含文件buffer。如果需要详细信息，选用"0"；如果只需要得知验证结果，选用"1"。当选用"0"时，接口调用时间较长。

c、请求包体示例

```
"token":"{xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx}",
"appid":"xxxx",
"info_type": 1
```

5) 返回值说明

a、返回主体包的内容

data字段为数据aes加密后再base64编码的字符串(相当于：base64(aes(data，key))),参数参考示例。

b、返回示例

```
{
    "errorcode": 0,
    "errormsg":"success",
    "data": "base64(密文)"
}
```

AES解密data后对应的数据如下：

当`info_type`为"0"时

```
{
  "ID": "4501111994xxxxxxxx",
  "name": "张三",
  "phone": "159********",
  "sex": "男",
  "nation": "汉",
  "ID_address": "广东省深圳市南山区***",
  "ID_birth": "xxxx",
  "ID_authority": "***公安局",
  "ID_valid_date": "xxxx.xx.xx-xxxx.xx.xx",
  "validatedata": "3344",
  "frontpic": "身份证正面照片的base64编码",
  "backpic": "身份证反面照片的base64编码",
  "videopic1": "视频截图1",
  "videopic2": "视频截图2",
  "videopic3": "视频截图3",
  "video": "视频的base64编码",
  "yt_errorcode": 0,
  "yt_errormsg": "通过",
  "livestatus": 0,
  "livemsg": "OK",
  "comparestatus": 0,
  "comparemsg": "OK",
  "type": 0
}
```


当`info_type`为"1"时


```
{
    "ID": "4501111994xxxxxxxx",
    "name": "张三",
    "phone": "159********",
    "sex": "男",
    "nation": "汉",
    "ID_address": "广东省深圳市南山区***",
    "ID_birth": "xxxx",
    "ID_authority": "***公安局",
    "ID_valid_date": "xxxx.xx.xx-xxxx.xx.xx",
    "validatedata": "3344",
    "yt_errorcode": 0,
    "yt_errormsg": "通过",
    "livestatus": 0,
    "livemsg": "OK",
    "comparestatus": 0,
    "comparemsg": "OK",
    "type": 0
}
```


签名算法

签名

上述提供的API接口，通过签名来验证请求的合法性。开发者通过将签名授权给调用方，使其具备使用API接口的能力。 密钥的获取及签名的生成方法如下：

Step 1 获取应用appid、密钥secretkey和过期时间expired

应用appid: 用来标识唯一的应用；

密钥secretkey: 使用加密算法时使用的秘钥；

expired: 当次请求的时间戳的有效时间；

Step 2 拼接有效签名串

a=xxxxx&m=xxxxxxx&t=1427786065&e=600

a为appid

m为调用的apiName，

t为当前时间戳，是一个符合UNIX Epoch时间戳规范的数值，单位为秒

e为此签名的凭证有效期，是一个符合UNIX Epoch时间戳规范的数值，单位为秒,

同appid和secretkey一样，由API提供方给定。

拼接有效签名串的结果,下文称之为orignal

Step 3 生成签名串

(1)API提供方 使用 HMAC-SHA1 算法对请求进行签名。

(2)签名串需要使用 Base64 编码。

根据签名方法signature= Base64(HMAC-SHA1(secretkey, orignal) + original)，其中secretkey为Step1获取，orignal为Step2中拼接好的签名串，对orignal使用HMAC-SHA1算法进行签名，然后将orignal附加到签名结果的末尾，再进行Base64编码，得到最终的sign。

注：此处使用的是标准的Base64编码，不是urlsafe的Base64编码，请注意。

Step 4 使用签名串

将Step 3生成的signature，填充到http请求的head头部的signature字段中即可。

NodeJS参考代码

```
var crypto = require('crypto');
//生成签名
function getAppSign(apiName, appId, appSecretKey, signExpired) {
    if (!apiName || !appId || !appSecretKey || !signExpired)
        return '';
    var now = parseInt(Date.now() / 1000);
    var plainText = 'a=' + appId + '&m=' + apiName + '&t=' + now + '&e=' + signExpired;
    var data = new Buffer(plainText, 'utf8');
    var res = crypto.createHmac('sha1', appSecretKey).update(data).digest();
    var bin = Buffer.concat([res, data]);
    var sign = bin.toString('base64');
    return sign;
}
```

错误码

类型:String

1) 成功

取值	说明
"0"	SUCESS 成功

2) 错误码说明

取值	说明
"601"	"调用引擎接口出错"
"602"	"视频活体检测没通过",
"603"	"视频活体比对没通过",
"604"	"没有达到匹配阈值",
"605"	"二次验证照片信息拉取为空",
"606"	"身份证照片信息拉取失败",
"607"	"视频像素太低，建议更换手机",
"608"	"视频声音太小"
"609"	"嘴唇动作幅度过小"
"610"	"脸部未完整露出"
"611"	"声音识别失败"
"612"	"未检测到声音"

注意要点

1. 兼容性问题：低版本的微信对小程序的支持不够。如果用低版本的微信进入小程序，会出现无任何反应的情况（不报错也无法进入）。这个有待微信小程序的改进，该版本的实名核身小程序未对API做兼容处理。
2. 版本要求问题：目前小程序跳转接口`wx.navigateToMiniProgram()`和`wx.navigateBackMiniProgram`对用户的微信客户端有版本要求，具体为：iOS 微信客户端 6.5.9 版本开始支持，Android 客户端即将在 6.5.10 版本开始支持。业务方需要**在跳转前判断用户手机上的微信版本是否能支持'小程序跳转'接口**，如果版本过低，则需要做相应的提示。

> 参考：https://mp.weixin.qq.com/debug/wxadoc/dev/api/navigateToMiniProgram.html

3. 手机系统差异：目前所知小程序对于ios系统的支持比对于Android系统的好。在ios系统中点击home键，然后回到小程序，页面仍然停留在离开时的页面；而Android系统中，小程序会出现类似重启的情况，页面会回到首页。