**哈尔滨慧通信息技术有限公司
一键登录产品技术文档**

|  |
| --- |
| **历史版本** |
| **版本\状态** | **作者** | **修改日期** | **描述** |
| V1.0 | qq58077302 | 2019-6-15 |  |
|  |  |  |  |

目录

[一．产品说明 3](#_Toc23425460)

[1.1概述 3](#_Toc23425461)

[1.2产品介绍 3](#_Toc23425462)

[1.3使用场景 4](#_Toc23425463)

[1.4产品特色 4](#_Toc23425464)

[二．通讯流程图 5](#_Toc23425465)

[三．部署文档 5](#_Toc23425466)

[3.1 服务端部署 5](#_Toc23425467)

[3.2 客户端部署 7](#_Toc23425468)

[3.2.1 IOS 7](#_Toc23425469)

[3.2.2 Android 12](#_Toc23425470)

[3.3 API参考 16](#_Toc23425471)

[3.3.1 iOS 16](#_Toc23425472)

[3.3.2 Android 23](#_Toc23425473)

[3.4错误码 30](#_Toc23425474)

[定义的业务错误码 30](#_Toc23425475)

[四．帮助 38](#_Toc23425476)

# 一．产品说明

## 1.1概述

**本章文档主要为您简单说明关于一键登录的产品相关信息**

## 1.2产品介绍

**一键登录**是哈尔滨慧通信息技术有限公司提供的APP一键登录的身份验证产品。与三大运营商直接合作，采用“通信网关取号”及SIM卡识别等技术，准确识别用户手机号码，替代APP上的注册、登录等场景需要使用到的短信验证码，优化APP用户在移动设备上的使用体验，降低用户流失。

## 1.3使用场景

**一键注册**

在使用手机号进行的移动端注册场景中，用户无需输入手机号即可一键完成注册流程。不必再输入字符验证和确认短信验证。

**免密登录**

在使用手机号进行的移动端登录场景中，用户无需输入手机号即可完成登录，无需再输入密码。

## 1.4产品特色

**降低短信拦截风险**

相较传统的验证码方式，一键免密登录不受制于短信通讯网络，可有效规避短信验证码泄露风险。

**降低登录、注册门槛**

登录注册环节，无需输入账号密码并接收验证码，只需授权手机号码，2秒便可瞬间登录。

**提高用户转化率**

极大地简化了APP交互逻辑，提升用户体验，登录注册环节用户不再“望而却步”，助力APP拉新留存。

**整合三网，减少开发成本**

只用对接一家服务，即可享受三网能力，减少开发工程量。

# 二．通讯流程图



# 三．部署文档

## 3.1 服务端部署

**简要描述：**

获取本机号码接口

**请求URL：**
<https://onepass.entinfo.cn/check_phone>

 **请求方式：**

POST

**数据类型：**
application/json

**请求参数**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| process\_id | 是 | str | 流水号 |
| sign | 是 | str | 用户签名，用于校验用户的key |
| token | 是 | str | 运营商校验用的token |

**请求示例：**

curl -H "Content-Type: application/json" -d '{"token": "客户端返回用于获取结果的token", "sign": "生成的用户签名", "process\_id": "客户端返回的流水号"}' https://onepass.entinfo.cn/check\_phone

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| status | int | 状态码 |
| result | str | 请求成功时，返回手机号或“” |
| error\_msg | str | 请求失败时，返回对应状态码的描述 |

**状态码**

|  |  |
| --- | --- |
| 状态码 | 描述 |
| 200 | 请求成功 |
| 500 | 从运营商获取结果失败 |
| 12000 | process\_id为空 |
| 12001 | token为空 |
| 12002 | sign为空 |
| 12003 | process\_id不为32位 |
| 12100 | process\_id不存在 |
| 12101 | process\_id已使用过 |
| 12102 | 未查询到对应的custom |
| 12103 | 查询到对应的custom不正确 |
| 12104 | 余额不足 |
| 12105 | 解密sign失败 |
| 12106 | sign解密后id不匹配 |
| 12107 | sign解密后key不匹配 |
| 12108 | sign中时间与当前时间差距超过一小时 |

**身份校验：**

将用户id, 以及md5加密后取16进制结果的key值，以及当前时间戳（毫秒为单位）通过”&&”连接。
（ID + “&&” + md5\_encode(KEY) + “&&” + timestamp）

将连接后的字符串进行rsa加密，加密公钥请联系商务获取，填充方法为PKCS1\_v1\_5。

**示例：**

ID： ‘zoekwui1hnmg49x5fwzf5la0ml5dziwn’
KEY： ‘gywzffojtnzl0vd6kcut8fcgyud5wg49’
时间戳： 1542355862990

keyMD5（16进制字符串表示）：’713b4c88400457548819f86a9765a410’

拼接结果：
‘zoekwui1hnmg49x5fwzf5la0ml5dziwn&&713b4c88400457548819f86a9765a410&&1542355862990’

加密后结果：
‘aj844tIWIY0LWe6x+z3RskI8lpWbGj44hk+jh+dnWKbS1O7KEBMEs0/pyT45HI2mvftkWVHLQwqI
2feCrASnM/1jF6K34cFWpN1893WhJoCEwtvGZxdWa+uEGmDbeCr1plYYhsf14kKQRwAJvx9ctQAC
a8NYybh7rFbKFNh8Bvk=’

rsa每次加密结果不同，仅供参考

## 3.2 客户端部署

### 3.2.1 IOS

#### 3.2.1.1 概述及资源

[SDK更新](https://docs.geetest.com/onelogin/deploy/client/ios%22%20%5Cl%20%22SDK%E6%9B%B4%E6%96%B0)

| 版本号 | 更新时间 | 更新内容 |
| --- | --- | --- |
| 1.3.0 | 2019/6/14 | 1、授权页UI自定义内容更新，三网统一，允许插入自定义控件；2、授权页面切换逻辑优化，允许手动控制页面消失时间；3、其他Bug修复。 |

环境需求

|  |  |
| --- | --- |
| 条目 | 资源 |
| 开发目标 | iOS8+ |
| 开发环境 | Xcode 10 |
| 系统依赖 | libc++.1.tbd、libz.1.2.8.tbd |
| SDK 三方依赖 | account\_login\_sdk\_noui\_core.framework、EAccountApiSDK.framework、TYRZSDK.framework、TYRZResource.bundle |

#### 3.2.1.2安装

导入 SDK 到项目工程并配置开发环境

1. 将下载获取的**OneLogin.framework**, account\_login\_sdk\_noui\_core.framework、EAccountApiSDK.framework、TYRZSDK.framework以及在**OneLoginResource.bundle**、TYRZResource.bundle共6个文件添加到工程中, 确保Copy items if needed已被勾选。

此外, 需要添加libc++.1.tbd、libz.1.2.8.tbd库进行依赖。

添加完后, 以Linked Frameworks and Libraries方式导入 framework。在拖入OneLogin.framework, account\_login\_sdk\_noui\_core.framework、EAccountApiSDK.framework、TYRZSDK.framework到工程后, 请检查所有的.framework是否被添加到PROJECT -> Build Phases -> Linked Frameworks and Libraries, 以确保正常编译。



1. 针对静态库中的Category, 需要在对应 target 的Build Settings->Other Linker Flags添加-ObjC编译选项。如果依然有问题，再添加-all\_load。



配置接口

开发者集成客户端 SDK 前, 必须先在您的服务器上搭建相应的**服务端获取登录信息的接口**，并配置从龙信通侧获取的AppID。这里以服务端配置成功，客户端开发步骤为例，如下：

1. 用 获取到的appid

|  |
| --- |
| [OneLogin registerWithAppID:@"---<分配的AppID>---"]; |

1. 调用预取号接口

从回调中可以获取到预取号是否成功的状态

|  |
| --- |
| [OneLogin preGetTokenWithCompletion:^(NSDictionary \* \_Nonnull sender) { NSLog(@"sender: %@", sender.description);}]; |

3.调用取号接口

开发者在完成预取号后, 通过该接口获取用于获取该用户手机号信息的访问令牌, 使用该访问令牌去查询用户的手机号及相关信息。

|  |
| --- |
| [OneLogin requestTokenWithViewController:weakSelf viewModel:viewModel completion:^(NSDictionary \* \_Nullable result) { NSNumber \*status = [result objectForKey:@"status"]; if ([@(200) isEqualToNumber:status]) { // 获取token成功 // TO-DO // 校验和获取登录的用户的数据  } else { NSLog(@"result: %@", result); }}]; |

**集成代码参考下方的代码示例。更进一步的示例, 见相关Demo。**

#### 3.2.1.3 代码示例

在工程中的文件头部导入静态库 OneLoginSDK.framework

|  |
| --- |
| #import <OneLoginSDK/OneLoginSDK.h> |

注册及预取号

在相应的控制页初始化方法中对OneLoginSDK实例调用注册方法:

|  |
| --- |
| [OneLogin registerWithAppID:@"---<申请的AppID>---"]; |

预取号

在每次获取免密登录的token之前, 需要调用预取号接口

|  |
| --- |
| [OneLogin preGetTokenWithCompletion:^(NSDictionary \* \_Nonnull sender) { NSLog(@"sender: %@", sender.description); NSNumber \*status = [sender objectForKey:@"status"]; if (status && [@(200) isEqualToNumber:status]) { // 预取号成功  } else { // 预取号失败  }}]; |

获取免密登录 token 并使用 token 获取用户登录信息

通过requestTokenWithViewController:viewModel:completion:获取免密登录token后, 在使用token获取用户登录信息。在此过程中, 需要设置UIViewController, 以展示授权页面, 并让**用户接受使用条款**

|  |
| --- |
| - (void)requestToken { OLAuthViewModel \*viewModel = [[OLAuthViewModel alloc] init]; // TO-DO 自定义viewModel // viewModel...  \_\_weak typeof(self) weakSelf = self; [OneLogin requestTokenWithViewController:weakSelf viewModel:viewModel completion:^(NSDictionary \* \_Nullable result) { NSNumber \*status = [result objectForKey:@"status"]; if ([@(200) isEqualToNumber:status]) { NSString \*processID = [result objectForKey:@"processID"]; NSString \*appID = [result objectForKey:@"appID"]; NSString \*token = [result objectForKey:@"token"];  [weakSelf validateTokenAndGetLoginInfo:token appID:appID processID:processID]; } else { NSLog(@"result: %@", result); } }];}// 使用token进行校验, 并获取用户的登录信息- (void)validateTokenAndGetLoginInfo:(NSString \*)token appID:(NSString \*)appID processID:(NSString \*)processID {  // 根据用户自己接口构造 // 下面以POST, application/json 为例 NSURL \*url = [NSURL URLWithString:@"---<您的校验接口地址>---"];  NSMutableURLRequest \*mRequest = [[NSMutableURLRequest alloc] initWithURL:url cachePolicy:(NSURLRequestCachePolicy)0 timeoutInterval:10.0];  NSMutableDictionary \*params = [[NSMutableDictionary alloc] init]; [params setValue:processID forKey:@"process\_id"]; [params setValue:appID forKey:@"id\_2\_sign"]; [params setValue:token forKey:@"token"];  NSData \*data = [NSJSONSerialization dataWithJSONObject:params options:(NSJSONWritingOptions)0 error:nil];  mRequest.HTTPMethod = @"POST"; mRequest.HTTPBody = data;  [[[NSURLSession sharedSession] dataTaskWithRequest:mRequest completionHandler:^(NSData \* \_Nullable data, NSURLResponse \* \_Nullable response, NSError \* \_Nullable error) { // TO-DO // 处理用户信息 NSLog(@"result data: %@", [[NSString alloc] initWithData:data encoding:NSUTF8StringEncoding]); }] resume]; } |

**更详细示例代码见 Demo**

**更详细的接口说明见接口文档或 SDK 头文件**

### 3.2.2 Android

#### 3.2.2.1 概述及资源

OneLogin Android SDK提供给集成Android原生客户端开发的开发者使用。

SDK更新

| 版本号 | 更新时间 | 更新内容 |
| --- | --- | --- |
| 0.5.0 | 2019/6/13 | 1、授权页UI自定义内容更新，三网统一，允许插入自定义控件；2、授权页面切换逻辑优化，允许手动控制页面消失时间；3、OneLoginThemeConfig类的路径变更 |
| 0.4.2 | 2019/5/31 | 移动版本由5.6.5改为最新的5.6.5.1，解决ssl证书过期问题 |

环境需求

|  |  |
| --- | --- |
| **条目** | **资源** |
| 开发目标 | 4.0以上 |
| 开发环境 | Android Studio 2.1.3 |
| 系统依赖 | v7包 |
| sdk三方依赖 | com.cmic.sso.sdk、com.unicom.xiaowo.login、cn.com.chinatelecom.account.api |

3.2.2.2 安装

导入 SDK 到项目工程并配置开发环境

1. 在 Demo 的libs包下，将获取的.aar文件拖拽到工程中的 libs 文件夹下。
2. 在拖入.aar到 libs 文件夹后, 还要检查.aar是否被添加到Library,要在项目的build.gradle下添加如下代码：

|  |
| --- |
| repositories { flatDir { dirs 'libs' }} |

1. 手动将aar包添加依赖：

|  |
| --- |
| compile(name: mandao\_onelogin\_android\_vx.y.z', ext: 'aar') |

1. 添加权限

|  |
| --- |
| <uses-permission android:name="android.permission.INTERNET" /> <uses-permission android:name="android.permission.READ\_PHONE\_STATE" /> <uses-permission android:name="android.permission.CHANGE\_NETWORK\_STATE" /> <uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" /> <uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" /> <uses-permission android:name="android.permission.WRITE\_SETTINGS" /> <uses-permission android:name="android.permission.GET\_TASKS" /> |

配置接口

开发者集成客户端 SDK 前, 必须先在您的服务器上搭建相应的[服务端获取登录信息的接口](https://docs.geetest.com/onelogin/deploy/server/api%22%20%5Ct%20%22_blank)，并配置从龙信通侧获取的AppID。这里以服务端配置成功，客户端开发步骤为例，如下：

1. 配置初始化接口

//`OneLogin`的授权页面将与此处配置的`Activity`绑定
OneLoginHelper.with().init(MainActivity.this);

1. 调用预取号接口

//第一个参数为所需要配置的App\_ID
//第二个参数为设置的超时时间
//第三个参数为所需实现监听回调结果接口
OneLoginHelper.with().preGetToken(appID,timeout,oneLoginListener);

1. 调用取号接口

//第一个参数为自定义的全局配置接口
//第二个参数为所需实现监听回调结果接口
OneLoginHelper.with().requestToken(oneLoginThemeConfig,oneLoginListener);

**集成代码参考下方的代码示例。更进一步的示例, 见相关 Demo**

代码示例

初始化与较验

初始化

在项目的具体页面的onCreate方法里面进行初始化。使用OneLogin的**授权页面**将与此处配置的Activity绑定。

|  |
| --- |
| OneLoginHelper.with().init(MainActivity.this); |

预取号

|  |
| --- |
| OneLoginHelper.with().preGetToken(app\_id,timeout,oneLoginListener);; |

取号

|  |
| --- |
| OneLoginHelper.with().preGetToken(App\_ID, 5000, new AbstractOneLoginListener() { @Override public void onResult(JSONObject jsonObject) { try { int status = jsonObject.getInt("status"); if (status == 200) { //预取号成功 } else { Toast.makeText(getApplicationContext(), jsonObject.toString(), Toast.LENGTH\_SHORT).show(); } } catch (JSONException e) { e.printStackTrace(); } }}); |

获取免密登录 token 并使用 token 获取用户登录信息

获取 token 后, 通过接口进行校验, 并获取登录信息。

|  |
| --- |
| OneLoginHelper.with().requestToken(new OneLoginThemeConfig.Builder().build(), new AbstractOneLoginListener() { @Override public void onResult(JSONObject jsonObject) { try { int statusResult = jsonObject.getInt("status"); if (statusResult == 200) { new Thread(new Runnable() { @Override public void run() { JSONObject jsonObject = new JSONObject(); try { jsonObject.put("process\_id", jsonObject.getString("process\_id")); jsonObject.put("token", jsonObject.getString("token")); } catch (JSONException e) { e.printStackTrace(); } String result = HttpUtils.requestNetwork(<---获取用户登录信息的接口--->, jsonObject); //根据结果判断是否成功，并在主线程关闭授权页。实现方法为： //OneLoginHelper.with().quitAuthActivity(); } }).start(); } else { Toast.makeText(getApplicationContext(), jsonObject.toString(), Toast.LENGTH\_SHORT).show(); } } catch (JSONException e) { e.printStackTrace(); } }}); |

页面关闭

在页面关闭的时候执行此方法。

|  |
| --- |
| @Overrideprotected void onDestroy() { super.onDestroy(); OneLoginHelper.with().cancel();} |

常见错误

**1. 全机型，全系统兼容吗？**

答：本产品只兼容4.0以上系统。

**2. Demo的运行有什么注意问题？**

答：需要申请对应的app\_id，并需配置相应的服务接口。这样，Demo就可以正常使用了。

**3. 只有一个回调，该怎样处理页面切换等逻辑呢？**

答：页面的所有状态都有错误码输出，可以根据错误码来实现。

**4. 怎样控制关闭授权页的时机？**

答：在requestToken()方法执行完成之后，根据错误码来自主实现关闭授权页的时机，默认授权页返回触发是直接关闭的。当请求成功之后需要页面跳转的时候也是关闭授权页的时机。开发者自定义控件的事件发生时也是关闭授权页的时机。

5. 授权页面的沉浸式状态栏或者设置状态栏的颜色怎么实现呢？

答：在application中registerActivityLifecycleCallbacks，然后在onActivityStarted中根据if (activity instanceof LoginAuthActivity||activity instanceof OneLoginActivity)来实现。

## 3.3 API参考

### 3.3.1 iOS

OneLogin 的主要外部调用接口

**[Protocol](https://docs.geetest.com/onelogin/apirefer/api/ios%22%20%5Cl%20%22Protocol)**

**userDidDismissAuthViewController**

用户点击了授权页面的返回按钮

**Declaration**

用户点击了授权页面的返回按钮

**Declaration**

|  |
| --- |
| - (void)userDidDismissAuthViewController; |

**userDidSwitchAccount 用户点击了授权页面的”切换账户”按钮**

**Declaration**

|  |
| --- |
| - (void)userDidSwitchAccount; |

**Method**

**registerWithAppID:向SDK注册AppID**

**Declaration**

|  |
| --- |
| + (void)registerWithAppID:(NSString \*)appID; |

**Parameters**

| Param | Description |
| --- | --- |
| appID | 产品id, 请在龙信通侧获取 |

**setDelegate:**

设置代理对象

**Declaration**

|  |
| --- |
| + (void)setDelegate:(nullable id<OneLoginDelegate>)delegate; |

**Parameters**

| Param | Description |
| --- | --- |
| delegate | 代理对象 |

**setRequestTimeout:**

设置请求超时时长。默认8秒。

**Declaration**

|  |
| --- |
| + (void)setRequestTimeout:(NSTimeInterval)timeout; |

**Parameters**

| Param | Description |
| --- | --- |
| timeout | 请求超时的时长 |

**preGetTokenWithCompletion:**

预取号接口

**Declaration**

|  |
| --- |
| + (void)preGetTokenWithCompletion:(void(^)(NSDictionary \*sender))completion; |

**Parameters**

| Param | Description |
| --- | --- |
| completion | 预取号的处理回调 |

**Discussion**

|  |
| --- |
| // 成功的返回格式{status = 200; // NSNumber, 200为成功的状态码processID = 47dab9b7c26629cd9bc117f88e2f9233; // NSString, 流水号appID = 2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*d; // NSString, 产品IDoperatorType = CU; // NSString, 运营商类型(CM/CU/CT)msg = "\U83b7\U53d6accessCode\U6210\U529f"; // NSString, 相关的描述消息}// 失败的返回格式{status = 500; // NSNumber, 500为失败的状态码processID = 47dab9b7c26629\*\*\*\*\*\*\*\*f88e2f9233; // NSString, 流水号appID = 2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*d; // NSString, 产品IDoperatorType = CT; // NSString, 运营商类型(CM/CU/CT)errorCode = -30003, // NSNumber, 当运营商不成功时的错误码msg = "\U83b7\U53d6accessCode\U6210\U529f"; // NSString, 相关的描述消息} |

**requestTokenWithViewController:viewModel:completion:**

取号接口

**Declaration**

|  |
| --- |
| + (void)requestTokenWithViewController:(UIViewController \*)viewController viewModel:(nullable OLAuthViewModel \*)viewModel completion:(void(^)(NSDictionary \* \_Nullable result))completion; |

**Parameters**

| Param | Description |
| --- | --- |
| viewController | 视图控制器 |
| viewModel | 自定义的试图模型 |
| completion | 取号接口的处理回调 |

**Discussion**

|  |
| --- |
| // 成功返回的格式:{status = 200; // NSNumber, 200为成功的状态码processID = 47dab9b7c26629cd9bc117\*\*\*\*\*\*\*\*33; // NSString, 流水号appID = 2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*d; // NSString, 产品IDtoken = 62718774ad1247188bc678\*\*\*\*\*\*\*\*d3; // NSString, 运营商返回的accessToken, 用于查询真实的本机号operatorType = CU; // NSString, 运营商类型(CM/CU/CT)msg = "\U83b7\U53d6accessCode\U6210\U529f"; // NSString, 运营商返回的Msg}// 失败返回的格式{status = 500; // NSNumber, 500为失败的状态码processID = 47dab9b7c26629cd9bc117\*\*\*\*\*\*\*\*33; // NSString, 流水号appID = 2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*d; // NSString, 产品IDoperatorType = CT; // NSString, 运营商类型(CM/CU/CT)errorCode = -30003, // NSNumber, 运营商返回的错误码msg = "\U83b7\U53d6accessCode\U6210\U529f"; // NSString, 相关的描述消息} |

**sdkVersion**

获取SDK版本号

**Declaration**

|  |
| --- |
| + (NSString \*)sdkVersion; |

**Return Value**

当前的SDK版本号

**OLAuthViewModel**

授权页面自定义UI模型

**Property**

**statusBarStyle**

状态栏样式。 默认 UIStatusBarStyleDefault。

**Declaration**

|  |
| --- |
| @property (nonatomic, assign) UIStatusBarStyle statusBarStyle; |

**naviTitle**

授权页导航的标题。默认为空字符串。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) NSAttributedString \*naviTitle; |

**naviBgColor**

授权页导航的背景颜色。默认白色。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) UIColor \*naviBgColor; |

**naviBackImage**

授权页导航左边的返回按钮的图片。默认白色系统样式返回图片。最大尺寸44x44, 推荐13x21。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) UIImage \*naviBackImage; |

**naviRightControl**

授权页导航右边的自定义控件。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) UIView \*naviRightControl; |

**naviHidden**

导航栏隐藏。默认不隐藏。

**Declaration**

|  |
| --- |
| @property (nonatomic, assign) BOOL naviHidden; |

**appLogo**

授权页面上展示的图标。默认为 “OneLogin” 图标。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) UIImage \*appLogo; |

**logoOffsetY**

Logo 视图竖直方向偏移。

**Declaration**

|  |
| --- |
| @property (nonatomic, assign) CGFloat logoOffsetY; |

**Discussion**

如果导航栏没有隐藏, 偏移距离等于该视图顶部到导航底部的距离。
如果导航栏隐藏, 偏移距离等于该视图顶部到状态栏之间的距离。

**logoHidden**

Logo 图片隐藏。默认不隐藏。

**Declaration**

|  |
| --- |
| @property (nonatomic, assign) BOOL logoHidden; |

**phoneNumAttributes**

号码预览文字的属性。默认黑色、24pt。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) NSDictionary<NSAttributedStringKey, id> \*phoneNumAttributes; |

**phoneNumLabelOffsetY**

号码预览竖直方向偏移。

**Declaration**

|  |
| --- |
| @property (nonatomic,assign) CGFloat phoneNumLabelOffsetY; |

**Discussion**

如果导航栏没有隐藏, 偏移距离等于该视图顶部到导航底部的距离。
如果导航栏隐藏, 偏移距离等于该视图顶部到状态栏之间的距离。

**switchButtonColor**

授权页切换账号按钮的颜色。默认蓝色。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) UIColor \*switchButtonColor; |

**switchButtonOffsetY**

授权页切换账号按钮竖直方向偏移。

**Declaration**

|  |
| --- |
| @property (nonatomic,assign) CGFloat switchButtonOffsetY; |

**Discussion**

如果导航栏没有隐藏, 偏移距离等于该视图顶部到导航底部的距离。
如果导航栏隐藏, 偏移距离等于该视图顶部到状态栏之间的距离。

**switchButtonHidden**

隐藏切换账号按钮。默认不隐藏。

**Declaration**

|  |
| --- |
| @property (nonatomic, assign) BOOL switchButtonHidden; |

**authButtonImages**

授权页认证按钮的背景图片, @[正常状态的背景图片, 不可用状态的背景图片, 高亮状态的背景图片]。默认正常状态为蓝色纯色, 不可用状态的背景图片时为灰色, 高亮状态为灰蓝色。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) NSArray<UIImage \*> \*authButtonImages; |

**authButtonTitle**

授权按钮文案。默认白色的”一键登录”。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) NSAttributedString \*authButtonTitle; |

**authButtonOffsetY**

授权按钮竖直方向偏移。

**Declaration**

|  |
| --- |
| @property (nonatomic, assign) CGFloat authButtonOffsetY; |

**Discussion**

如果导航栏没有隐藏, 偏移距离等于该视图顶部到导航底部的距离。
如果导航栏隐藏, 偏移距离等于该视图顶部到状态栏之间的距离。**sloganOffsetY**

Slogan竖直方向偏移。

**Declaration**

|  |
| --- |
| @property (nonatomic, assign) CGFloat sloganOffsetY; |

**Discussion**

偏移距离等于该视图顶部到导航底部的距离。

**sloganTextColor**

Slogan 文字颜色。默认灰色, 12pt。

**Declaration**

|  |
| --- |
| @property (nonatomic, strong) UIColor \*sloganTextColor; |

**defaultCheckBoxState**

授权页面上条款勾选框初始状态。默认 YES。

**Declaration**

|  |
| --- |
| @property (nonatomic, assign) BOOL defaultCheckBoxState; |

**checkedImage**

授权页面上勾选框勾选的图标。默认为蓝色图标。最大尺寸32x32, 推荐18x18。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) UIImage \*checkedImage; |

**uncheckedImage**

授权页面上勾选框未勾选的图标。默认为白色图标。最大尺寸32x32, 推荐18x18。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) UIImage \*uncheckedImage; |

**privacyTermsAttributes**

隐私条款文字属性。默认基础文字灰色, 条款蓝色高亮, 12pt。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) NSDictionary<NSAttributedStringKey, id> \*privacyTermsAttributes; |

**additionalPrivacyTerms**

额外的条款。默认为空。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) NSArray<OLPrivacyTermItem \*> \*additionalPrivacyTerms; |

**Seealso**

**[OLPrivacyTermItem](http://docs.geetest.com/onelogin/apirefer/api/ios%22%20%5Cl%20%22OLPrivacyTermItem)**

**termTextColor**

服务条款普通文字的颜色。默认灰色。

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) UIColor \*termTextColor; |

**termsOffsetY**

隐私条款竖直方向偏移

**Declaration**

|  |
| --- |
| @property (nonatomic,assign) CGFloat termsOffsetY; |

**Discussion**

偏移距离等于该视图底部到屏幕底部(视图控制器)底部的距离。

**customUIHandler**

自定义区域视图的处理block

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, copy) OLCustomUIHandler customUIHandler; |

**Discussion**

提供的视图容器使用NSLayoutConstraint与相关的视图进行布局约束。
如果导航栏没有隐藏, 顶部与导航栏底部对齐, 左边与屏幕对齐, 右边与屏幕对齐, 底部与屏幕对齐。
如果导航栏隐藏, 顶部与状态栏底部对齐, 左边与屏幕对齐, 右边与屏幕对齐, 底部与屏幕对齐。

**Seealso**

|  |
| --- |
| typedef void(^OLCustomUIHandler)(UIView \*customAreaView); |

**backgroundImage**

授权页面背景图片

**Declaration**

|  |
| --- |
| @property (nullable, nonatomic, strong) UIImage \*backgroundImage; |

**OLPrivacyTermItem**

隐私和条款条目

**Property**

**termTitle**

隐私和条款的标题

**Declaration**

|  |
| --- |
| @property (nonatomic, strong) NSString \*termTitle; |

**termTitle**

隐私和条款的链接

**Declaration**

|  |
| --- |
| @property (nonatomic, strong) NSURL \*termLink; |

**Method**

**initWithTitle:linkURL:**

使用标题和链接创建实例

### 3.3.2 Android

 **[SDK方法说明](http://docs.geetest.com/onelogin/apirefer/api/android%22%20%5Cl%20%22SDK%E6%96%B9%E6%B3%95%E8%AF%B4%E6%98%8E)**

**初始化**

**方法描述**

|  |
| --- |
| public void init(Context context) |

**参数说明**

| 参数 | 类型 | 说明 |
| --- | --- | --- |
| context | Context | 上下文 |

预取号

**方法描述**

app\_id：产品id，请在龙信通侧获取

|  |
| --- |
| public void preGetToken(String app\_id,int timeout, AbstractOneLoginListener listener) |

**参数说明**

| 参数 | 类型 | 说明 |
| --- | --- | --- |
| app\_id | String | 龙信通配置唯一id |
| timeout | int | 设置的预取号超时时间 |
| listener | AbstractOneLoginListener | 回调监听器，需要开发者自己实现 |

**AbstractOneLoginListener实现接口**

**回调**

方法说明

整个流程的结果返回

|  |
| --- |
| public onResult(JSONObject jsonObject) |

参数说明

| 参数 | 类型 | 说明 |
| --- | --- | --- |
| jsonObject | JSONObject | 错误信息 |

jsonObject的参数说明

**预取号成功**

| 参数名 | 必须 | 类型 | 说明 |
| --- | --- | --- | --- |
| msg | 是 | String | 运营商返回的状态信息 |
| process\_id | 是 | String | 流水号 |
| app\_id | 是 | String | 龙信通配置唯一id |
| operatorType | 是 | String | 客户端获取的运营商 |
| clienttype | 是 | String | 客户端,1表示Android |
| sdk | 是 | String | sdk的版本号 |
| status | 是 | int | 状态码,状态码为200 |
| accessCode | 否 | String | 运营商返回的accessCode |
| number | 否 | String | 运营商返回的去敏手机号 |

**预取号失败**

| 参数名 | 必须 | 类型 | 说明 |
| --- | --- | --- | --- |
| errorCode | 是 | String | 错误码 |
| msg | 否 | String | 运营商返回的状态信息 |
| process\_id | 否 | String | 流水号 |
| app\_id | 否 | String | 龙信通配置唯一id |
| metadata | 是 | JSONObject | 具体的错误原因 |
| operatorType | 否 | String | 客户端获取的运营商 |
| clienttype | 是 | String | 客户端,1表示Android |
| sdk | 是 | String | sdk的版本号 |
| status | 是 | int | 状态码,状态码为500 |

**取号**

**方法描述**

|  |
| --- |
| public void requestToken(OneLoginThemeConfig oneLoginThemeConfig, AbstractOneLoginListener listener) |

**参数说明**

| 参数 | 类型 | 说明 |
| --- | --- | --- |
| oneLoginThemeConfig | OneLoginThemeConfig | 全局配置接口 |
| listener | AbstractOneLoginListener | 回调监听器，需要开发者自己实现 |

**OneLoginThemeConfig实现接口**

**创建实例**

|  |
| --- |
| OneLoginThemeConfig oneLoginThemeConfig=new OneLoginThemeConfig.Builder().build(); |

方法说明

| **方法名** | **传参类型** | **说明** |
| --- | --- | --- |
| setNavColor | int | 设置标题栏的整个背景颜色(如0xffffffff,以下与之相同) |
| setNavText | String | 设置标题栏中间的文字 |
| setNavTextColor | int | 设置标题栏中间的文字颜色 |
| setNavReturnImgPath | String | 设置标题栏的返回按钮图片(放在drawable文件下的文件名,不包括后缀。如:ic\_launcher。以下的图片路径与之相同) |
| setAuthNavTransparent | boolean | 设置授权页导航栏是否透明 |
| setAuthBGImgPath | String | 设置整个页面的背景图片 |
| setLogoImgPath | String | 设置logo的地址 |
| setLogoWidthDip | int | 设置logo的宽度，单位dp |
| setLogoHeightDip | int | 设置logo的高度，单位dp |
| setLogoHidden | boolean | 隐藏logo |
| setLogoOffsetY | int | 设置logo相对于标题栏下边缘y偏移 |
| setLogoOffsetY\_B | int | 设置logo相对于底部y偏移 |
| setNumberColor | int | 设置手机号码字体颜色 |
| setNumberSize | String | 设置号码栏字体大小 |
| setNumFieldOffsetY | int | 设置号码栏相对于标题栏下边缘y偏移 |
| setNumFieldOffsetY\_B | int | 设置号码栏相对于底部y偏移 |
| setSloganTextColor | int | 设置slogan字体颜色 |
| setSloganOffsetY | int | 设置slogan相对于标题栏下边缘y偏移 |
| setSloganOffsetY\_B | int | 设置slogan相对于底部y偏移 |
| setLogBtnText | String | 设置登录按钮文字 |
| setLogBtnTextColor | int | 设置登录按钮文字颜色 |
| setLogBtnImgPath | String | 设置登录按钮图片 |
| setLogBtnOffsetY | int | 设置登录按钮相对于标题栏下边缘y偏移 |
| setLogBtnOffsetY\_B | int | 设置登录按钮相对于底部y偏移 |
| setSwitchAccTextColor | int | 设置切换账号的字体颜色 |
| setSwitchAccHidden | boolean | 隐藏切换账号 |
| setSwitchOffsetY | int | 设置切换账号相对于标题栏下边缘y偏移 |
| setSwitchOffsetY\_B | int | 设置切换账号相对于底部y偏移 |
| setClauseColor | int,int | 设置隐私条款名称颜色(基础文字颜色，协议文字颜色) |
| setUncheckedImgPath | String | 设置选择框未选中时候的图片 |
| setCheckedImgPath | String | 设置选择框选中时候的图片 |
| setPrivacyState | boolean | 隐私条款check默认状态 |
| setClauseOne | String,String | 设置开发者隐私条款1名称和URL(名称，url) |
| setClauseTwo | String,String | 设置开发者隐私条款2名称和URL(名称，url) |
| setPrivacyOffsetY | int | 设置隐私条相对于标题栏下边缘y偏移 |
| setPrivacyOffsetY\_B | int | 设置隐私条相对于底部y偏移 |

**AbstractOneLoginListener实现接口**

**回调**

方法说明

整个流程的结果返回

|  |
| --- |
| public onResult(JSONObject jsonObject) |

参数说明

| 参数 | 类型 | 说明 |
| --- | --- | --- |
| jsonObject | JSONObject | 错误信息 |

jsonObject的参数说明

**取号成功**

| 参数名 | 必须 | 类型 | 说明 |
| --- | --- | --- | --- |
| msg | 是 | String | 运营商返回的状态信息 |
| process\_id | 是 | String | 流水号 |
| app\_id | 是 | String | 龙信通配置唯一id |
| operatorType | 是 | String | 客户端获取的运营商 |
| clienttype | 是 | String | 客户端,1表示Android |
| sdk | 是 | String | sdk的版本号 |
| status | 是 | int | 状态码,状态码为200 |
| token | 是 | String | 运营商返回的accessToken |

**取号失败**

| **参数名** | **必须** | **类型** | **说明** |
| --- | --- | --- | --- |
| errorCode | 是 | String | 错误码 |
| msg | 是 | String | 运营商返回的状态信息 |
| process\_id | 是 | String | 流水号 |
| app\_id | 是 | String | 龙信通配置唯一id |
| metadata | 是 | JSONObject | 具体的错误原因 |
| operatorType | 是 | String | 客户端获取的运营商 |
| clienttype | 是 | String | 客户端,1表示Android |
| sdk | 是 | String | sdk的版本号 |
| status | 是 | int | 状态码,状态码为500 |

**开发者自定义控件**

**方法描述**

在requestToken()方法之前实现。允许开发者在授权页面titlebar和body添加自定义的控件
注意：自定义的控件不允许覆盖SDK默认的UI。

|  |
| --- |
| public OneLoginHelper addOneLoginRegisterViewConfig(String id, AuthRegisterViewConfig authRegisterViewConfig) |

**代码示例**

|  |
| --- |
| OneLoginHelper.with().addOneLoginRegisterViewConfig(id, authRegisterViewConfig); |

参数说明

| 参数 | 类型 | 说明 |
| --- | --- | --- |
| id | String | 开发者自定义控件名称 |
| authRegisterViewConfig | AuthRegisterViewConfig | 配置开发者自定义控件的控件来源，位置和处理逻辑等 |

**AuthRegisterViewConfig实现接口**

**创建实例**

|  |
| --- |
| AuthRegisterViewConfig authRegisterViewConfig =new AuthRegisterViewConfig.Builder().build(); |

方法说明

| 方法名 | 传参类型 | 说明 |
| --- | --- | --- |
| setView | View | 开发者传入自定义的控件，需要设置好控件的布局属性，SDK只支持RelativeLayout布局 |
| setRootViewId | int | 设置控件的位置。RootViewId.ROOT\_VIEW\_ID\_TITLE\_BAR指标题栏，RootViewId.ROOT\_VIEW\_ID\_BODY指授权页空白处 |
| setCustomInterface | CustomInterface | 设置控件事件。CustomInterface是一个接口，实现了点击方法 |

**finish授权页**

SDK默认除了返回按钮触以外，默认是不finsih授权页的，需要开发自行实现。

**方法描述**

|  |
| --- |
| public void quitAuthActivity() |

**代码示例**

|  |
| --- |
| OneLoginHelper.with().quitAuthActivity(); |

**参数说明**

无

**关闭验证**

**方法描述**

在activity的onDestroy()方法中实现

|  |
| --- |
| public void cancel() |

**代码示例**

|  |
| --- |
| OneLoginHelper.with().cancel(); |

**参数说明**

无

**内部方法**

无

**方法描述**

获取SDK版本号

|  |
| --- |
| public void sdkVersion() |

**参数说明**

无

**代码示例**

|  |
| --- |
| OneLoginHelper.with().sdkVersion() |

**混淆规则**

|  |
| --- |
| -dontwarn com.geetest.onelogin.\*\*-keep class com.geetest.onelogin.\*\* {\*;}-dontwarn com.cmic.sso.sdk.\*\*-keep class com.cmic.sso.sdk.\*\* {\*;}-dontwarn com.unicom.xiaowo.login.\*\*-keep class com.unicom.xiaowo.login.\*\* {\*;}-dontwarn cn.com.chinatelecom.account.api.\*\*-keep class cn.com.chinatelecom.account.api.\*\* {\*;} |

## 3.4错误码

IOS

[状态码](http://docs.geetest.com/onelogin/apirefer/errorcode/ios%22%20%5Cl%20%22%E7%8A%B6%E6%80%81%E7%A0%81)

在返回中的"status"字段下的内容

| 状态码 | 说明 |
| --- | --- |
| 200 | 操作成功 |
| 500 | 操作失败 |

错误码

在返回中的"errorCode"字段下的内容

### 定义的业务错误码

| 错误码 | 说明 |
| --- | --- |
| -200 | 不支持的运营商类型 |
| -500 | 预取号阶段, 移动返回数据为空 |
| -501 | 取号阶段, 移动返回数据为空 |
| -502 | 无效的AccessCode。可能预取号未成功, 进行了取号操作。 |
| -503 | 重复使用的AccessCode。重复进行了取号操作, AccessCode仅能使用一次。 |

**Cocoa 定义的错误码**

来自NSURLErrorDomain的错误

| **ErrorCode** | **Description** |
| --- | --- |
| -999 | NSURLErrorCancelled用户操作导致的请求中断, 一般忽略处理 |
| -1000 | NSURLErrorBadURLURL异常 |
| -1001 | NSURLErrorTimedOut请求超时 |
| -1002 | NSURLErrorUnsupportedURL不支持的URL |
| -1003 | NSURLErrorCannotFindHost无法找到主机 |
| -1004 | NSURLErrorCannotConnectToHost无法连接到服务器 |
| -1005 | NSURLErrorNetworkConnectionLost网络丢失, 一般弱网或者网络突然中断导致 |
| -1006 | NSURLErrorDNSLookupFailedDNS查询失败 |
| -1007 | NSURLErrorHTTPTooManyRedirects过多的请求跳转, 服务器返回过多的302 |
| -1008 | NSURLErrorResourceUnavailable访问的资源不可用 |
| -1009 | NSURLErrorNotConnectedToInternet未连接到互联网 |
| -1010 | NSURLErrorRedirectToNonExistentLocation重定向到不存在的地址 |
| -1011 | NSURLErrorBadServerResponse服务器无响应 |
| -1012 | NSURLErrorUserCancelledAuthentication客户端取消了安全认证, 或者证书不匹配或服务端不支持ssl和tls |
| -1013 | NSURLErrorUserAuthenticationRequired客户端要求安全认证, 服务端不支持ssl或tls |
| -1014 | NSURLErrorZeroByteResource返回字节流为空 |
| -1015 | NSURLErrorCannotDecodeRawData无法解析的原始数据 |
| -1016 | NSURLErrorCannotDecodeContentData解析返回内容错误 |
| -1017 | NSURLErrorCannotParseResponse无法解析响应体 |
| -1102 | NSURLErrorNoPermissionsToReadFile无资源访问权限, 一般为challenge等参数有误, challenge只可被用来请求一次, 失效后可能会遇到该问题 |
| -1200 | NSURLErrorSecureConnectionFailed创建安全连接失败 |
| -1201 | NSURLErrorServerCertificateHasBadDate服务端证书异常 |
| -1202 | NSURLErrorServerCertificateUntrusted服务端证书不可信 |
| -1203 | NSURLErrorServerCertificateHasUnknownRoot服务端使用未知的根证书 |

Android

ErrorCode

**OneLogin**

OneLogin产品的错误代码

| Code | Description |
| --- | --- |
| -20101 | app\_id未传 |
| -20200 | 当前网络不可用 |
| -20201 | 当前手机没有电话卡 |
| -20202 | 当前没有开启流量 |
| -20203 | 未获取到运营商 |
| -20204 | 获取运营商错误 |
| -20205 | 预取号超时 |
| -20301 | 退出取号页面 |
| -20302 | 按返回键退出取号页面 |
| -20303 | 切换账号登陆方式 |
| -40101 | 移动运营商预取号失败 |
| -40102 | 移动运营商取号失败 |
| -40201 | 联通运营商预取号失败 |
| -40202 | 联通运营商取号失败 |
| -40301 | 电信运营商预取号失败 |
| -40302 | 电信运营商取号失败 |
| -50100 | SDK内部请求pre\_get\_token接口返回异常 |
| -50101 | SDK内部请求pre\_get\_token接口解密失败 |
| -50102 | SDK内部请求pre\_get\_token接口返回错误 |

运营商

**[ErrorCode](http://docs.geetest.com/onelogin/apirefer/errorcode/operator%22%20%5Cl%20%22ErrorCode)**

**OneLogin**

运营商接口返回失败常见的错误代码,位于错误回调JSON中meta\_data内的运营商错误码

移动运营商

| Code | Description |
| --- | --- |
| 102102 | 网络异常 |
| 102507 | 登录超时(授权页点登录按钮时) |
| 103101 | appkey错误 |
| 103102 | 包签名错误 |
| 103108 | 短信验证码错误 |
| 103109 | 短信验证码校验超时 |
| 103111 | 网关IP错误(运营商误判) |
| 103119 | appid不存在 |
| 103125 | 手机号码格式错误 |
| 103203 | 缓存失效 |
| 103211 | 其他错误 |
| 103901 | 短验下发次数已达上限(5次/min,10次/day) |
| 103902 | scrip失效 |
| 103911 | token请求过于频繁，10分钟内获取token且未使用的数量不超过30个 |
| 105001 | 联通取号失败 |
| 105002 | 移动取号失败 |
| 105003 | 电信取号失败 |
| 105021 | 已达当天取号限额 |
| 105302 | appid不在白名单 |
| 200005 | 用户未授权(READ\_PHONE\_STATE) |
| 200010 | 取号请求时获取imsi失败 |
| 200014 | 手机号码不存在 |
| 200020 | 用户取消登录 |
| 200021 | 数据解析异常 |
| 200022 | 无网络 |
| 200023 | 请求超时 |
| 200024 | 数据网络切换失败 |
| 200025 | 其他错误(socket、系统未授权数据蜂窝权限等） |
| 200026 | 输入参数错误 |
| 200027 | 未开启数据网络 |
| 200028 | 网络请求出错 |
| 200030 | 没有初始化参数 |
| 200038 | 电信重定向失败 |
| 200039 | 电信取号接口返回失败 |
| 200040 | UI资源加载异常 |
| 200048 | 未安装sim卡 |
| 200050 | EOF异常 |
| 200060 | 切换账号(未使用SDK短验时返回) |
| 200061 | 授权页面异常 |
| 200062 | 预取号不支持联通 |
| 200063 | 预取号不支持电信 |

电信运营商

| Code | Description |
| --- | --- |
| -64 | permission-denied(无权限访问) |
| -65 | API-request-rates-Exceed-Limitations(调用接口超限) |
| -10001 | 取号失败 |
| -10002 | 参数错误 |
| -10003 | 解密失败 |
| -10004 | ip受限 |
| -10005 | 异网取号回调参数异常 |
| -10006 | Mdn取号失败，且属于电信网络 |
| -10007 | 重定向到异网取号 |
| -10008 | 超过预设取号阈值 |
| -10009 | 时间戳过期 |
| -20005 | sign-invalid(签名错误） |
| -20006 | 应用不存在 |
| -20007 | 公钥数据不存在 |
| -20100 | 内部解析错误 |
| -20102 | 加密参数解析失败 |
| -30001 | 时间戳非法 |
| -30003 | topClass失效，请查看5.3常见问题。 |
| 51002 | 参数为空 |
| 51114 | 无法获取手机号数据 |
| -8001 | 请求网络异常 |
| -8002 | 请求参数异常 |
| -8003 | 请求超时 |
| -8004 | 移动网络未开启 |
| -8005 | 请先初始化SDK |
| -8100 | 网络无连接 |
| -8101 | 响应码错误 |
| -8102 | Socket超时异常（读取超时、连接超时） |
| -8103 | 域名解析异常 |
| -8104 | IO异常 |
| -8201 | 响应码错误 |
| -720001 | 切换异常 |
| -720002 | 切换异常超时 |

联通运营商

| Code | Description |
| --- | --- |
| 100 | 应用未授权 |
| 101 | 应用秘钥错误 |
| 102 | 应用无效 |
| 103 | 应用未授权该IP访问 |
| 104 | 应用访问次数不足 |
| 105 | 应用包名不正确 |
| 200 | tokenId无效 |
| 201 | token已失效 |
| 202 | token未授权该应用访问 |
| 203 | 登录鉴权级别不满足接口鉴权要求 |
| 300 | 接口未开放 |
| 301 | 应用未授权码访问该接口 |
| 302 | IP未授权码访问该接口 |
| 303 | 应用访问接口次数超日限额 |
| 400 | 请求参数为空 |
| 401 | 请求参数不完整 |
| 402 | 请求参数非法 |
| 600 | 请求非法 |
| 1000 | 请求解析错误 |
| 1001 | 请求已失效 |
| 1002 | 验签失败 |
| 1003 | 授权码已过期 |
| 1004 | 加密方式不支持 |
| 1005 | RSA加密错误 |
| 1010 | 服务间访问失败 |
| 1011 | 服务间访问错误 |
| 2004 | 用户不存在 |
| 3001 | unikey无效 |
| 3002 | 跳转异网取号 |
| 3003 | 本网执行取号失败,不需要重定向 |
| 3004 | NET取号失败 |
| 3005 | 上网方式为WIFI，无法取号 |
| 3006 | urlencode编码失败 |
| 3007 | 请求认证接口异常 |
| 3008 | imsi取号失败 |
| 3009 | 非联通号码 |
| 3010 | 网关取号错误 |
| 3011 | 源IP鉴权失败 |
| 3012 | 网关取号失败 |
| 3013 | 电信网关取号失败 |
| 3014 | 电信网关取号错误 |
| 3015 | 获取accessCode请求参数失败 |
| 3016 | 移动网关取号失败 |
| 3017 | 移动网关取号错误 |

# 四．帮助

4.1常见问题

[OneLogin支持哪些运营商和网络环境？](http://docs.geetest.com/onelogin/help/faq%22%20%5Cl%20%22OneLogin%E6%94%AF%E6%8C%81%E5%93%AA%E4%BA%9B%E8%BF%90%E8%90%A5%E5%95%86%E5%92%8C%E7%BD%91%E7%BB%9C%E7%8E%AF%E5%A2%83%EF%BC%9F)

目前支持移动 2G、3G、4G，联通 3G、4G，电信 4G。

双卡双待手机如何使用OneLogin？

双卡手机以开启数据流量的 SIM 卡进行认证。

WiFi环境下如何通过网关验证？

在数据网络开通的前提下，WiFi 环境中会强制使用数据网络触发网关请求，请求仅为几K，可正常校验。

一键登录取号原理是怎样的？

通过运营商网关数据获取当前流量卡的号码。

是否支持携号转网用户？

SDK 不提供接口判断用户是否为携号转网用户，但可以判断用户流量卡当前的运营商。即携号转网用户可正常使用 OneLogin。

OneLogin支持虚拟运营商吗？

仅支持三大运营商（移动、联通、电信）的手机号进行一键登录。

目前存在某些卡只有流量功能，是否可以通过一键登录？

三大运营商（移动、联通、电信）的卡，能够正常使用数据网络，运营商通过网关能够正常取号，均会被作为正常的卡来判断。

OneLogin支持哪些终端？

支持 Android 和IOS，不支持 H5 和小程序。

OneLogin是如何收费的？

预取号不收费，返回 token 也不收费，成功返回 token 对应的手机号时进行计费。

用户后台可以看到每条请求的客户端类型等具体返回结果？

目前后台只有统计结果，建议客户保存日志，有客户端类型等明细的返回，方便查看明细数据并且快速定位问题。

OneLogin 与 OnePass 有什么区别？

OneLogin 经授权后自动获取用户手机号，适用于登录和注册场景；OnePass 不返回号码， 仅返回待校验号码是否本机的校验结果，适用于基于手机号码的风控场景（例如充值交易）。两者具体区别如下，客户可自主选择：

|  | **OneLogin** | **OnePass** |
| --- | --- | --- |
| 业务流程 | 自动获取手机号 | 手动输入手机号 |
| UI风格 | 局部自定义 | 全部自定义 |
| 隐私性 | 需要用户授权 | 不需要用户授权 |

OneLogin 界面 UI 客户是否可自定义？

授权页 UI 客户仅可局部自定义，其余页面客户可自定义

安卓授权页设计规范



苹果授权页设计规范

