

# 阿里云号码认证Android v2.12.0.1（标准版）接入文档

---

## 1. 概述

## 2. 准备工作

## 3. 运行demo工程

## 4. 开发环境搭建

### 4.1 拷贝aar包

### 4.2 APP工程AndroidManifest.xml增加Activity声明

### 4.3 混淆keep规则

### 4.4 若开启资源混淆，需要配置

### 4.5 lib依赖

### 4.6 增加v4或者v7包依赖

### 4.7 权限列表

### 4.8 Android 9.0以上支持http配置

## 5. SDK方法说明

获取认证实例(getInstance)

检查认证环境(checkAuthEnvEnable)

加速本机号码校验 (accelerateVerify)

本机号码校验token(getVerifyToken)

加速授权页拉起 (accelerateLoginPage)

一键登录唤起授权页 (getLoginToken)

退出授权页 (quitLoginPage)

关闭授权页loading (hideLoginLoading)

返回默认上网卡运营商 (getCurrentCarrierName)

使用xml添加自定义控件至一键登录授权页 (addAuthRegisterXmlConfig)

添加代码编写的自定义控件至登录授权页(addAuthRegistViewConfig)

注意：每次调用 getVerifyToken 授权请求之前，都需初始化一次AuthRegisterViewConfig,因为在授权页...

一键登录修改授权页主题 (setAuthUIConfig)

SDK回调说明

- 1) 获取token回调
- 2) 加速唤起授权页/加速本机号码校验回调
- 3) 控件点击事件回调

建议代码调用顺序

一键登录获取手机号

本机号码校验结果

SDK返回码

授权页点击事件响应码

## 6. 一键登录授权页面说明

### 6.1 授权页面设计规范

### 6.2 弹窗授权页面设计规范（支持横竖屏，以竖屏示意）

注意：请勿遮挡协议栏、一键登录按钮以及掩码或者将字体颜色设置为透明，否则获取到一键登录token

### 6.3 授权页配置说明

1. 授权页导航栏
2. 授权页Logo
3. 授权页Slogan
4. 授权页号码栏
5. 授权页登录按钮
6. 授权页隐私栏
7. 切换方式控件
8. 页面相关函数

### 6.7 常见问题

1. 首次取号时，为什么 APP 网络通信正常，号码认证一直失败
2. checkEnvAvailable函数返回false
3. 获取token失败，一般有哪些原因
4. Android双卡手机一键登登录过程中，一键登录逻辑是怎么样的
5. 权限问题
6. 返回错误码600005，手机终端不安全有哪些原因
7. 非法手机号
8. VPN报错
9. 页面非法修改
10. setSDKAuthSDKInfo的秘钥如何获取？

- 11. 当使用移动卡请求一键登录不成功，出现以下报错信息时。
- 12. 当使用移动卡请求一键登录不成功，出现以下报错信息时
- 13. 内存泄漏
- 14. AAR介绍

## 1. 概述

官网下载SDK后进行解压，解压后包含三个文件是

- SDK说明文档
- Realease note
- 4个aar包
- DEMO工程（在控制台下载DEMO压缩包）

SDK同时包含了本机号码校验和一键登录两个功能。

号码认证服务包含本机号码校验和一键登录时两个不同的功能，使用场景不一样，无需一起使用。

- 本机号码校验使用场景：用户输入手机号码，通过SDK获取token，服务端携带输入的手机号码和token去运营商网关进行校验，返回的结果时用户当前上网使用的号码与输入的号码是否一致。
- 一键登录使用场景：用户无需输入手机号码，SDK会拉起授权页，用户确认授权后，SDK会获取token，服务端携带token到运营商网关获取用户当前上网使用的号码，并返回给APP服务端。

## 2. 准备工作

- 请确保您的终端设备已经开启了4G网络（联通、移动支持3G网络，但接口耗时会增加）
- 请确保已经在阿里云控制台开通了号码认证服务并创建了对应的方案，点击进入阿里云控制台。
- 您也可以登录阿里云官网查看接入流程，点击查看完整使用流程。

## 3. 运行demo工程

## 4. 开发环境搭建

## 4.1 拷贝aar包

下载SDK并解压后，将所有aar文件拷贝至工程的libs目录下。

## 4.2 APP工程AndroidManifest.xml增加Activity声明

```
1 <!--联通电信授权页-->
2 <activity
3     android:name="com.mobile.auth.gatewayauth.LoginAuthActivity"
4     android:configChanges="orientation|keyboardHidden|screenSize"
5     android:exported="false"
6     android:theme="@style/authsdk_activity_dialog"           使用弹
    窗模式必须添加!!!
7     android:launchMode="singleTop" />
8 <!--协议页面webview-->
9 <activity
10     android:name="com.mobile.auth.gatewayauth.activity.AuthWebVe
    wActivity"
11     android:configChanges="orientation|keyboardHidden|screenSize"
12     android:exported="false"
13     android:launchMode="singleTop"
14     android:screenOrientation="behind" />
15 <!--移动授权页-->
16 <activity
17     android:name="com.cmic.sso.sdk.activity.LoginAuthActivity"
18     android:configChanges="orientation|keyboardHidden|screenSize"
19     android:exported="false"
20     android:launchMode="singleTop" />
```

## 4.3 混淆keep规则

1 已经将混淆规则写入aar包，不再需要单独配置

## 4.4 若开启资源混淆，需要配置

```
1 "R.drawable.authsdk*",
2 "R.layout.authsdk*",
3 "R.anim.authsdk*",
4 "R.id.authsdk*",
5 "R.string.authsdk*",
6 "R.style.authsdk*",
```

## 4.5 lib依赖

[点击下载依赖包](#)

## 4.6 增加v4或者v7包依赖

使用v4包com.android.support:support-v4版本高于25.4.0或者v7包com.android.support:appcompat-v7版本高于25.4.0

## 4.7 权限列表

```
1 <uses-permission android:name="android.permission.INTERNET" />
   <!-- 网络访问 -->
2 <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" /> <!-- 检查wifi网络状态 -->
3 <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" /> <!-- 检查网络状态 -->
4 <uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" /> <!-- 切换网络通道 -->
5 <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/> <!-- 本地信息缓存 -->
6 <uses-permission android:name="android.permission.CHANGE_WIFI_STATE" /> <!-- 开关wifi状态, 解决国内机型移动网络权限问题需要 -->
```

## 4.8 Android 9.0以上支持http配置

- app AndroidManifest.xml里面, Application节点增加usesCleartextTraffic配置

```

1 <application
2     android:name=".DemoApplication"
3     android:icon="@drawable/ic_launcher"
4     android:label="@string/app_name"
5     android:supportsRtl="true"
6     android:theme="@style/AppTheme"
7     android:usesCleartextTraffic="true">

```

- 目前中国移动提供的个别接口为http请求，对于全局禁用Http的项目，需要设置Http白名单。以下为运营商http接口域名：onekey.cmpassport.com，enrichgw.10010.com。具体可以参考Demo代码。

## 5. SDK方法说明

### 获取认证实例(getInstance)

```

1 /**
2  * 获取号码认证服务实例，此实例为单例，获取多次为同一对象
3  * @param context      Android上下文
4  * @param tokenListener 需要实现的获取token回调
5  * @return PhoneNumberAuthHelper
6  */
7 public static PhoneNumberAuthHelper getInstance(Context context,
8     TokenResultListener tokenListener)

```

### 检查认证环境(checkAuthEnvEnable)

```

1 /**
2  * SDK环境检查函数，检查终端是否支持号码认证，通过TokenResultListener返回code
3  * type 1: 本机号码校验 2: 一键登录
4  * 600024 终端支持认证
5  * 600013 系统维护，功能不可用
6  */
7 public void checkEnvAvailable(@IntRange(from = 1, to = 2) int type
8 )

```

## 加速本机号码校验（accelerateVerify）

```
1 /**
2
3 public void accelerateVerify(int overdueTime, final PreLoginResult
    Listener listener);
```

## 本机号码校验token(getVerifyToken)

```
1 /**
2  * 获取认证token
3  *
4  * @param totalTimeout 超时时间 单位ms
5  */
6 public void getVerifyToken(final int totalTimeout)
```

## 加速授权页拉起（accelerateLoginPage）

```
1 /**
2  * 加速授权页唤起
3  *
4  * @param overdueTime 预取号有效期
5  * @param listener 预取号回调
6  */
7 public void accelerateLoginPage(final int overdueTime, final PreLo
    ginResultListener listener)
```

## 一键登录唤起授权页（getLoginToken）

```
1 /**
2  * 获取登录token 调起一键登录授权页面，在用户授权后获取一键登录的Token
```

```
3 *
4 * @param totalTimeout 超时时间 单位ms
5 */
6 public void getLoginToken(final Context context, final int totalTimeout)
```

## 退出授权页 (quitLoginPage)

```
1 /**
2 * 退出授权认证页
3 * sdk完成回调之后不会关闭授权页，需要开发者主动调用quitLoginPage退出授权页
4 */
5 public void quitLoginPage()
```

## 关闭授权页loading (hideLoginLoading)

```
1 /**
2 * 关闭授权页loading
3 * sdk完成回调之后不会关闭loading，需要开发者主动调用hideLoginLoading关闭loading
4 */
5 public void hideLoginLoading()
```

## 返回默认上网卡运营商 (getCurrentCarrierName)

```
1 /**
2 * 返回默认上网卡运营商
3 *
4 * @return CMCC、CUCC、CTCC
5 */
6 public String getCurrentCarrierName()
```



## 使用xml添加自定义控件至一键登录授权页 (addAuthRegisterXmlConfig)

```
1 /**
2  * 添加自定义View
3  *
4  * @param xmlConfig
5  */
6 public void addAuthRegisterXmlConfig(AuthRegisterXmlConfig xmlConfig)
```

一次add，XML内绘制的自定义控件全部添加完成

初始化 addAuthRegisterXmlConfig 类时需要先调静态内部类Builder()里面的 2 个方法

- setLayout： 开发者传入自定义的控件的xml资源ID
- AbstractPnsViewDelegate： 授权页使用xml添加自定义布局时，可以配合该Delegate类实现xml中相关view的操作，比如事件监听以及动态UI改动等等，当xml对应的view加载后SDK将调用 onViewCreated(View)方法通知view已经创建OK，此时可以获取xml中的view并进行相关事件绑定等操作。**特别注意：onViewCreated(View)中返回的View不要用强引用，或者用完要及时释放，否则容易造成内存泄漏。**

调用示例：

```
1 mAlicomAuthHelper.addAuthRegisterXmlConfig(new AuthRegisterXmlConfig.Builder()
2 .setLayout(R.layout.xxxxxx, new AbstractPnsViewDelegate() {
3     @Override public void onViewCreated(View view) {
4         //这里返回的View，不建议用强引用，如果要用，请及时释放，否则容易造成内存泄漏
5         findViewById(R.id.xxxx).setOnClickListener(new View.OnClickListener() {
6             @Override public void onClick(View v) {
7                 //do something
8             }
9         });
10    }
11 })
```

```
12 .build());
```

## 添加代码编写的自定义控件至登录授权页(addAuthRegistViewConfig)

```
1 /**
2  * 动态添加控件
3  *
4  * @param viewID 开发者自定义控件名称
5  * @param viewConfig 配置开发者自定义控件的控件来源、位置和处理逻辑
6  */
7 public void addAuthRegistViewConfig(String viewID, AuthRegisterViewConfig viewConfig)
```

**注意：**每次调用 `getVerifyToken` 授权请求之前，都需初始化一次 `AuthRegisterViewConfig`，因为在授权页关闭时都会清空注入进去的 `AuthRegisterViewConfig`，具体实现请见 `demo` 工程。

初始化 `AuthRegisterViewConfig` 类时需要先调静态内部类 `Builder()` 里面的 3 个方法

- `setView`：开发者传入自定义的控件，开发者需要提前设置好控件的布局属性，SDK 只支持 `RelativeLayout` 布局
- `setRootViewId`：设置控件的位置，目前SDK 授权页允许在3个位置插入开发者控件  
`RootViewId.ROOT_VIEW_ID_TITLE_BAR`，标题栏  
`RootViewId.ROOT_VIEW_ID_BODY`，授权页空白处  
`RootViewId.ROOT_VIEW_ID_NUMBER`，授权页号码掩码区域
- `setCustomInterface`：设置控件事件  
`public Builder setCustomInterface(CustomInterface customInterface)`  
调用示例：

```
1 mAlicomAuthHelper.addAuthRegistViewConfig("switch_acc_tv", new AuthRegisterViewConfig.Builder()
2 .setView(mRL)
3 .setRootViewId(AuthRegisterViewConfig.RootViewId.ROOT_VIEW_ID_BODY)
4 .setCustomInterface(new CustomInterface() {
```

```

5     @Override
6     public void onClick(Context context) {
7         startActivityForResult(new Intent(context, SecondActivity.class), 1234);
8     }
9 }).build());

```

获取 token 成功之后，需把通过setView()注入进去的view 置为 null

## 一键登录修改授权页主题（setAuthUIConfig）

```

1 /**
2  * 修改授权页面主题，开发者可以通过 此方法修改授权页面主题，需在 getLoginToken
   接口之前调用
3  *
4  * @param authUIConfig 登录授权页UI自定义配置
5  */
6 public void setAuthUIConfig(AuthUIConfig authUIConfig)

```

调用示例：

```

1 setAuthUIConfig(new AuthUIConfig.Builder()
2     .setLogBtnText("一键登录")
3     .setLogBtnClickableColor(Color.BLACK)
4     .setLogBtnUnClickableColor(Color.BLUE)
5     .setLogBtnTextColor(Color.WHITE).setLogoHidden(false)
6     .setNavColor(0xff026ED2)
7     .setNavText("免密登录")
8     .setNavTextColor(Color.WHITE)
9     .setNumberColor(Color.WHITE)
10    .setNumberSize(28)
11    .setNumberColor(0xff000000).create());

```

## SDK回调说明

## 1) 获取token回调

- 回调返回的ret都通过 `TokenRet tokenRet = JSON.parseObject(ret, TokenRet.class)` 解析。
- 授权页唤起成功、获取token成功都会回调onTokenSuccess方法（要区分两次成功可以通过返回码来区分）。
- 获取token失败会回调onTokenFailed。
- 获取token回调示例代码:

```
1 mTokenListener = new TokenResultListener() {
2     @Override
3     public void onTokenSuccess(final String ret) {
4         MainActivity.this.runOnUiThread(new Runnable() {
5             @Override
6             public void run() {
7                 /*
8                  *   setText just show the result for get tok
9                  *   en.
10                 *   use ret to verfiy number.
11                 */
12                 //ResultCode#CODE_START_AUTHPAGE_SUCCESS是授权
13                 //页唤起成功码，若不需要处理，则过滤
14                 if (ResultCode.CODE_START_AUTHPAGE_SUCCESS.e
15                     quals(tokenRet.getCode())) {
16                     return;
17                 }
18                 TokenRet tokenRet = JSON.parseObject(ret, Tok
19                     enRet.class);
20                 if (tokenRet != null) {
21                     token = tokenRet.getToken();
22                 }
23                 mAlicomAuthHelper.quitLoginPage();
24             }
25         });
26     }
27 }
```

```

25         @Override
26         public void onTokenFailed(final String ret) {
27             MainActivity.this.runOnUiThread(new Runnable() {
28                 @Override
29                 public void run() {
30                     /*
31                      * setText just show the result for get token
32                      * do something when getToken failed, such as use sms verify code.
33                      */
34                     TokenRet tokenRet = JSON.parseObject(ret, TokenRet.class);
35                     mAlicomAuthHelper.quitLoginPage();
36                 }
37             });
38         }
39     };

```

## 2) 加速唤起授权页/加速本机号码校验回调

```

1 public interface PreLoginResultListener {
2     /**
3      * @param vendor 返回预取成功运营商
4      */
5     void onTokenSuccess(String vendor);
6     /**
7      * @param vendor 返回预取失败运营商
8      * @param ret 返回失败原因
9      */
10    void onTokenFailed(String vendor, String ret);
11 }

```

预取号回调示例代码

```

1 mPreLoginResultListener = new PreLoginResultListener() {

```

```

2         @Override
3         public void onTokenSuccess(final String s) {
4             MainActivity.this.runOnUiThread(new Runnable() {
5                 @Override
6                 public void run() {
7                     /*
8                      * 推荐在登录页初始化的时候调用
9                      * 如果没有合适的调用时机
10                     * 不调用此接口也没关系
11                     * 千万不要APP冷启动初始化就调用!!!!
12                     * 不要调用完预取号后马上调用getLoginToke
13
14                     n!!!!
15
16                     * 最好判断用户是否登录，若已登录不要使用此接
17                     □!!!!
18
19                     */
20                     mRetTV.setText("预取号成功:" + s);
21                 }
22             });
23         }
24     }
25
26     @Override
27     public void onTokenFailed(final String s, final Strin
28     g s1) {
29         /*
30          * 预取号调用失败
31          * 不用太关注，还是可以直接在用户点击"登录"时，调用getLog
32          inToken
33
34          */
35         mRetTV.setText("预取号失败:" + s + s1);
36     }
37 }

```

### 3) 控件点击事件回调

授权页控件点击事件通过此回调返回

```

1 public interface AuthUIControlClickListener {
2     /**
3      *
4      * @param code      控件点击事件code
5      * @param context  Android上下文
6      * @param jsonObj  点击事件返回的具体内容，不同控件返回的事件内容有所不同
7      */
8     void onClick(String code, Context context, JSONObject jsonObj)
9     ;
10 }

```

控件点击事件回调示例代码

```

1 mAlicomAuthHelper.setUIClickListener(new AuthUIControlClickListener() {
2     @Override
3     public void onClick(String code, Context context, JSONObject jsonObj) {
4         Log.e("xxxxxx", "OnUIControlClick:code=" + code +
5             ", jsonObj=" + (jsonObj == null ? "" : jsonObj.toJSONString()));
6     }
7 });

```

## 建议代码调用顺序

```

1 /*
2 *    1.初始化获取token实例
3 */
4 mTokenListener = new TokenResultListener() {}
5
6 /*
7 *    2.初始化SDK实例
8 */
9 mAlicomAuthHelper = PhoneNumberAuthHelper.getInstance(context, mTokenListener);
10

```

```

11  /*
12  *    3. 设置SDK密钥
13  */
14  mAlicomAuthHelper.setAuthSDKInfo();
15
16  /*
17  *    4. 检测终端网络环境是否支持一键登录或者号码认证，根据回调结果确定是否可以使
    用一键登录功能
18  */
19  mAlicomAuthHelper.checkEnvAvailable(PhoneNumberAuthHelper#SERVICE
    _TYPE_LOGIN);
20
21  /*
22  *    5. 若步骤4返回true，则根据业务情况，调用预取号或者一键登录接口
23  *        详见demo接入工程
24  */
25  mAlicomAuthHelper.getLoginToken(context, 5000);

```

## 一键登录获取手机号

- 当您成功获取到getLoginToken成功获取token后，将token传递至您的服务端，服务端携带token调用阿里云的getMobile接口即可进行最终的取号操作。

## 本机号码校验结果

- 当您成功获取到getVerifyToken成功获取token后，将token传递至您的服务端，服务端携带token调用阿里云的VerifyMobile接口即可进行最终的取号操作。

## SDK返回码

返回码	返回码描述	建议
600000	获取token成功	无
600001	唤起授权页成功	无
600002	唤起授权页失败	建议切换到其他登录方式



600004	获取运营商配置信息失败	创建工单联系工程师
600005	手机终端不安全	切换到其他登录方式
600007	未检测到sim卡	提示用户检查 SIM 卡后重试
600008	蜂窝网络未开启	提示用户开启移动网络后重试
600009	无法判断运营商	创建工单联系工程师
600010	未知异常	创建工单联系工程师
600011	获取token失败	切换到其他登录方式
600012	预取号失败	无
600013	运营商维护升级，该功能不可用	创建工单联系工程师
600014	运营商维护升级，该功能已达最大调用次数	创建工单联系工程师
600015	接口超时	切换到其他登录方式
600017	AppID、Appkey解析失败	秘钥未设置或者设置错误，请先检查秘钥信息，如果无法解决问题创建工单联系工程师
600021	点击登录时检测到运营商已切换	提示用户退出授权页，重新登录
600023	加载自定义控件异常	检查自定义控件添加是否正确
600024	终端环境检查支持认证	无
600025	终端检测参数错误	检查传入参数类型与范围是否正确
600026	授权页已加载时不允许调用加速或预取号接口	检查是否有授权页拉起后，去调用preLogin或者accelerateAuthPage的接口，该行为不允许

除阿里云SDK返回码外，运营商返回码见阿里云官网[链接](#)

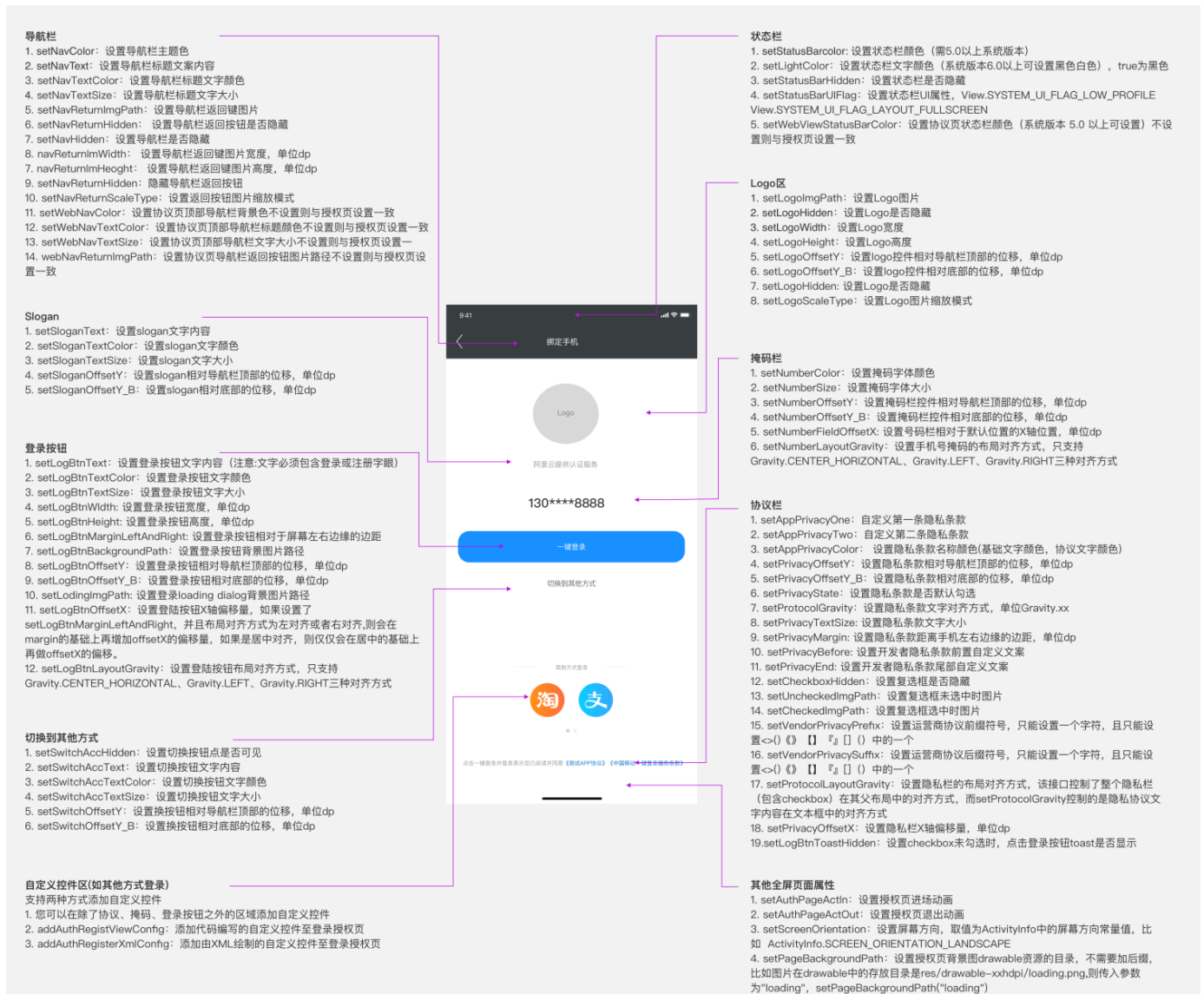
## 授权页点击事件响应码

响应码	响应码描述
700000	点击返回，用户取消免密登录
700001	点击切换按钮，用户取消免密登录
700002	点击登录按钮事件
700003	点击check box事件

## 6. 一键登录授权页面说明

注意：涉及图片路径的参数，仅仅为图片名称（不带路径或后缀名），并且图片需要放置在drawable、drawable-xxhdpi等目录下。

### 6.1 授权页面设计规范



### 6.2 弹窗授权页面设计规范（支持横竖屏，以竖屏示意）

标题栏

1. setNavColor: 设置标题栏主题色  
2. setNavText: 设置标题栏标题文案内容  
3. setNavTextColor: 设置标题栏标题文字颜色  
4. setNavTextSize: 设置标题栏标题文字大小  
5. setNavReturningPath: 设置标题栏返回键图片  
6. setNavReturnHidden: 设置标题栏返回按钮是否隐藏  
7. setNavHidden: 设置标题栏是否隐藏  
8. navReturnimWidth: 设置标题栏返回键图片宽度, 单位dp  
9. navReturnimHeight: 设置标题栏返回键图片高度, 单位dp  
10. setNavReturnHidden: 隐藏标题栏返回按钮  
11. setNavReturnScaleType: 设置返回按钮图片缩放模式  
12. setWebNavColor: 设置协议页顶部导航栏背景色不设置则与授权页设置一致  
13. setWebNavTextColor: 设置协议页顶部导航栏标题颜色不设置则与授权页设置一致  
14. setWebNavTextSize: 设置协议页顶部导航栏文字大小不设置则与授权页设置一致  
15. webNavReturningPath: 设置协议页导航栏返回按钮图片路径不设置则与授权页设置一致

Slogan

1. setSloganText: 设置slogan文字内容  
2. setSloganTextColor: 设置slogan文字颜色  
3. setSloganTextSize: 设置slogan文字大小  
4. setSloganOffsetY: 设置slogan相对导航栏顶部的位移, 单位dp  
5. setSloganOffsetY\_B: 设置slogan相对底部的位移, 单位dp

登录按钮

1. setLogBtnText: 设置登录按钮文字内容 (注意: 文字必须包含登录或注册字眼)  
2. setLogBtnTextColor: 设置登录按钮文字颜色  
3. setLogBtnTextSize: 设置登录按钮文字大小  
4. setLogBtnWidth: 设置登录按钮宽度, 单位dp  
5. setLogBtnHeight: 设置登录按钮高度, 单位dp  
6. setLogBtnMarginLeftAndRight: 设置登录按钮相对于屏幕左右边缘的边距  
7. setLogBtnBackgroundPath: 设置登录按钮背景图片路径  
8. setLogBtnOffsetY: 设置登录按钮相对导航栏顶部的位移, 单位dp  
9. setLogBtnOffsetY\_B: 设置登录按钮相对底部的位移, 单位dp  
10. setLodingImgPath: 设置登录loading dialog背景图片路径  
11. setLogBtnOffsetX: 设置登录按钮X轴偏移量, 如果设置了setLogBtnMarginLeftAndRight, 并且布局对齐方式为左对齐或者右对齐, 则会在margin的基础上再增加offsetX的偏移量, 如果是居中对齐, 则仅仅会在居中的基础上再做offsetX的偏移。  
12. setLogBtnLayoutGravity: 设置登录按钮布局对齐方式, 只支持Gravity.CENTER\_HORIZONTAL、Gravity.LEFT、Gravity.RIGHT三种对齐方式

切换到其他方式

1. setSwitchAccHidden: 设置切换按钮点是否可见  
2. setSwitchAccText: 设置切换按钮文字内容  
3. setSwitchAccTextColor: 设置切换按钮文字颜色  
4. setSwitchAccTextSize: 设置切换按钮文字大小  
5. setSwitchOffsetY: 设置切换按钮相对导航栏顶部的位移, 单位dp  
6. setSwitchOffsetY\_B: 设置切换按钮相对底部的位移, 单位dp

自定义控件区(如其他方式登录)

支持两种方式添加自定义控件  
1. 您可以在除了协议、掩码、登录按钮之外的区域添加自定义控件  
2. addAuthRegistViewConfig: 添加代码编写的自定义控件至登录授权页  
3. addAuthRegisterXmlConfig: 添加由XML绘制的自定义控件至登录授权页

状态栏

弹窗模式状态栏不通过号码认证SDK设置, APP自行设置

Logo区

1. setLogoimgPath: 设置Logo图片  
2. setLogoHidden: 设置Logo是否隐藏  
3. setLogoWidth: 设置Logo宽度  
4. setLogoHeight: 设置Logo高度  
5. setLogoOffsetY: 设置logo控件相对导航栏顶部的位移, 单位dp  
6. setLogoOffsetY\_B: 设置logo控件相对底部的位移, 单位dp  
7. setLogoHidden: 设置Logo是否隐藏  
8. setLogoScaleType: 设置Logo图片缩放模式

掩码栏


1. setNumberColor: 设置掩码字体颜色  
2. setNumberSize: 设置掩码字体大小  
3. setNumberOffsetY: 设置掩码栏控件相对导航栏顶部的位移, 单位dp  
4. setNumberOffsetY\_B: 设置掩码栏控件相对底部的位移, 单位dp  
5. setNumberFieldOffsetX: 设置号码栏相对于默认位置的X轴位置, 单位dp  
6. setNumberLayoutGravity: 设置手机号掩码的布局对齐方式, 只支持Gravity.CENTER\_HORIZONTAL、Gravity.LEFT、Gravity.RIGHT三种对齐方式

协议栏

1. setAppPrivacyOne: 自定义第一条隐私条款  
2. setAppPrivacyTwo: 自定义第二条隐私条款  
3. setAppPrivacyColor: 设置隐私条款名称颜色(基础文字颜色, 协议文字颜色)  
4. setPrivacyOffsetY: 设置隐私条款相对导航栏顶部的位移, 单位dp  
5. setPrivacyOffsetY\_B: 设置隐私条款相对底部的位移, 单位dp  
6. setPrivacyState: 设置隐私条款是否默认勾选  
7. setProtocolGravity: 设置隐私条款文字对齐方式, 单位Gravity.xx  
8. setPrivacyTextSize: 设置隐私条款文字大小  
9. setPrivacyMargin: 设置隐私条款距离手机左右边缘的边距, 单位dp  
10. setPrivacyBefore: 设置开发者隐私条款前置自定义文案  
11. setPrivacyEnd: 设置开发者隐私条款尾部自定义文案  
12. setCheckboxHidden: 设置复选框是否隐藏  
13. setUncheckedImPath: 设置复选框未选中时图片  
14. setCheckedImPath: 设置复选框选中时图片  
15. setVendorPrivacyPrefix: 设置运营商协议前缀符号, 只能设置一个字符, 且只能设置<>()《》[]『』{}()中的一个  
16. setVendorPrivacySuffix: 设置运营商协议后缀符号, 只能设置一个字符, 且只能设置<>()《》[]『』{}()中的一个  
17. setProtocolLayoutGravity: 设置隐私栏的布局对齐方式, 该接口控制了整个隐私栏(包含checkbox)在其父布局中的对齐方式, 而setProtocolGravity控制的是隐私协议文字内容在本框中的对齐方式  
18. setPrivacyOffsetX: 设置隐私栏X轴偏移量, 单位dp  
19. setLogBtnToastHidden: 设置checkbox未勾选时, 点击登录按钮toast是否显示

弹窗页面属性

1. setAuthPageActIn: 设置授权页进场动画  
2. setAuthPageActOut: 设置授权页退出动画  
3. setScreenOrientation: 设置屏幕方向, 取值为ActivityInfo中的屏幕方向常量值, 比如 ActivityInfo.SCREEN\_ORIENTATION\_LANDSCAPE  
4. setPageBackgroundPath: 设置授权页背景图drawable资源的目录, 不需要加后缀, 比如图片在drawable中的存放目录是res/drawable-xxhdpi/loading.png, 则传入参数为"loading", setPageBackgroundPath("loading")  
5. setDialogWidth: 设置弹窗模式授权页宽度, 单位dp, 设置大于0即为弹窗模式  
6. setDialogHeight: 设置弹窗模式授权页高度, 单位dp, 设置大于0即为弹窗模式  
7. setDialogOffsetX: 设置弹窗模式授权页X轴偏移, 单位dp  
8. setDialogOffsetY: 设置弹窗模式授权页Y轴偏移, 单位dp  
9. setDialogBottom: 设置授权页是否居于底部  
10. 蒙版设置: 通过修改授权页的弹窗主体实现, 默认为android:theme="@style/authsdk\_activity\_dialog", 可通过开发者自己实现的逻辑设定蒙版的样式



注意：请勿遮挡协议栏、一键登录按钮以及掩码或者将字体颜色设置为透明，否则获取到一键登录token

## 6.3 授权页配置说明

### 1. 授权页导航栏

方法	参数类型	说明
setStatusBarColor	int	设置状态栏颜色（系统版本 5.0 以上可设置）
setLightColor	int	设置状态栏字体颜色（系统版本 6.0 以上可

19

		设置黑色、白色)。 true 为黑色	
setNavColor	int	设置导航栏颜色	
setNavText	String	设置导航栏标题文字	
setNavTextColor	int	设置导航栏标题文字颜色	
setNavTextSize	int	设置导航栏标题文字大小	
setNavReturnImgPath	String	设置导航栏返回键图片	
setNavReturnHidden	boolean	设置导航栏返回按钮隐藏	
setNavHidden	boolean	设置默认导航栏是否隐藏	
setStatusBarHidden	boolean	设置状态栏是否隐藏	
setStatusBarUIFlag	int	设置状态栏UI属性 View.SYSTEM_UI_FLAG_LOW_PROFILE View.SYSTEM_UI_FLAG_LAYOUT_FULLSCREEN	
setWebViewStatusBarColor	int	设置协议页状态栏颜色 (系统版本 5.0 以上可设置) 不设置则与授权页设置一致	
setWebNavColor	int	设置协议页顶部导航栏背景色 不设置则与授权页设置一致	
setWebNavTextColor	int	设置协议页顶部导航栏标题颜色 不设置则与授权页设置一致	
setWebNavTextSize	int	设置协议页顶部导航栏文字大小	

		不设置则与授权页设置一致	
webNavReturnImgPath	String	设置协议页导航栏返回按钮图片路径 不设置则与授权页设置一致	
setBottomNavColor	int	设置底部虚拟按键背景色（系统版本 5.0 以上可设置）	

## 2. 授权页Logo

方法	参数类型	说明
setLogoHidden	boolean	隐藏logo
setLogoImagePath	String	设置logo 图片
setLogoWidth	int	设置logo 控件宽度
setLogoHeight	int	设置logo 控件高度
setLogoOffsetY	int	设置logo 控件相对导航栏顶部的位移，单位dp
setLogoOffsetY_B	int	设置logo 控件相对底部的位移，单位dp
setLogoScaleType	ImageView.ScaleType	设置logo图片缩放模式

## 3. 授权页Slogan

方法	参数类型	说明
setSloganText	String	设置slogan 文字内容
setSloganTextColor	int	设置slogan 文字颜色
setSloganTextSize	int	设置slogan 文字大小
setSloganOffsetY	int	设置slogan 相对导航栏顶部的位移，单位dp
setSloganOffsetY_B	int	设置slogan 相对底部的位移，单位dp

#### 4. 授权页号码栏

方法	参数类型	说明
setNumberColor	int	设置手机号码字体颜色
setNumberSize	int	设置手机号码字体大小
setNumFieldOffsetY	int	设置号码栏控件相对导航栏顶部的位移，单位 dp
setNumFieldOffsetY_B	int	设置号码栏控件相对底部的位移，单位 dp
setNumberFieldOffsetX	int	设置号码栏相对于默认位置的X轴偏移量，单位dp
setNumberLayoutGravity	int	设置手机号掩码的布局对齐方式，只支持Gravity.CENTER_HORIZONTAL、Gravity.LEFT、Gravity.RIGHT三种对齐方式

#### 5. 授权页登录按钮

方法	参数类型	说明
setLogBtnText	String	设置登录按钮文字
setLogBtnTextColor	int	设置登录按钮文字颜色
setLogBtnTextSize	int	设置登录按钮文字大小
setLogBtnWidth	int	设置登录按钮宽度，单位 dp
setLogBtnHeight	int	设置登录按钮高度，单位dp
setLogBtnMarginLeftAndRight	int	设置登录按钮相对于屏幕左右边缘边距
setLogBtnBackgroundPath	String	设置登录按钮背景图片路径
setLogBtnOffsetY	int	设置登录按钮相对导航栏顶部的位移，单位 dp
setLogBtnOffsetY_B	int	设置登录按钮相对底部的位移，单位 dp
setLoadingImgPath	String	设置登录loading dialog 背景图片路径

setLogBtnOffsetX	int	设置登陆按钮X轴偏移量，如果设置了 setLogBtnMarginLeftAndRight，并且布局对齐方式为左对齐或者右对齐,则会在margin的基础上再增加offsetX的偏移量，如果是居中对齐，则仅仅会在居中的基础上再做offsetX的偏移。
setLogBtnLayoutGravity	int	设置登陆按钮布局对齐方式，只支持 Gravity.CENTER_HORIZONTAL、Gravity.LEFT、Gravity.RIGHT三种对齐方式

## 6. 授权页隐私栏

方法	参数类型	说明
setAppPrivacyOne	String,String	设置开发者隐私条款 1 名称和URL(名称, url)
setAppPrivacyTwo	String,String	设置开发者隐私条款 2 名称和URL(名称, url)
setAppPrivacyColor	int,int	设置隐私条款名称颜色(基础文字颜色, 协议文字颜色)
setPrivacyOffsetY	int	设置隐私条款相对导航栏顶部的位移, 单位dp
setPrivacyOffsetY_B	int	设置隐私条款相对底部的位移, 单位dp
setPrivacyState	boolean	设置隐私条款是否默认勾选
setProtocolGravity	int	设置隐私条款文字对齐方式, 单位Gravity.xxx
setPrivacyTextSize	int	设置隐私条款文字大小, 单位sp
setPrivacyMargin	int	设置隐私条款距离手机左右边缘的边距, 单位dp
setPrivacyBefore	String	设置开发者隐私条款前置自定义

		文案
setPrivacyEnd	String	设置开发者隐私条款尾部自定义文案
setCheckboxHidden	boolean	设置复选框是否隐藏
setUncheckedImgPath	String	设置复选框未选中时图片
setCheckedImgPath	String	设置复选框选中时图片
setVendorPrivacyPrefix	String	设置运营商协议前缀符号，只能设置一个字符，且只能设置<>()《》【】『』[]()中的一个
setVendorPrivacySuffix	String	设置运营商协议后缀符号，只能设置一个字符，且只能设置<>()《》【】『』[]()中的一个
setProtocolLayoutGravity	int	设置隐私栏的布局对齐方式，该接口控制了整个隐私栏（包含checkbox）在其父布局中的对齐方式，而setProtocolGravity控制的是隐私协议文字内容在文本框中的对齐方式
setPrivacyOffsetX	int	设置隐私栏X轴偏移量，单位dp
setLogBtnToastHidden	boolean	设置checkbox未勾选时，点击登录按钮toast是否显示

## 7. 切换方式控件

方法	参数类型	说明
setSwitchAccHidden	boolean	设置切换按钮点是否可见
setSwitchAccText	String	设置切换按钮文字内容
setSwitchAccTextColor	int	设置切换按钮文字颜色
setSwitchAccTextSize	int	设置切换按钮文字大小
setSwitchOffsetY	int	设置切换按钮相对导航栏顶部的位移，单位 dp
setSwitchOffsetY_B	int	设置切换按钮相对底部的位移，单位 dp



8. 页面相关函数

方法	参数类型	说明
setAuthPageActIn	String	设置授权页进场动画
setAuthPageActOut	String	设置授权页退出动画
setScreenOrientation	int	设置屏幕方向，取值为ActivityInfo中的屏幕方向常数值，比如ActivityInfo.SCREEN_ORIENTATION_LANDSCAPE
setPageBackgroundPath	String	设置授权页背景图drawable资源的目录，不需要加后缀，比如图片在drawable中的存放目录是res/drawable-xxhdpi/loading.png,则传入参数为"loading",setPageBackgroundPath("loading")。
setDialogWidth	int	设置弹窗模式授权页宽度，单位dp,设置大于0即为弹窗模式
setDialogHeight	int	设置弹窗模式授权页高度，单位dp，设置大于0即为弹窗模式
setDialogOffsetX	int	设置弹窗模式授权页X轴偏移，单位dp
setDialogOffsetY	int	设置弹窗模式授权页Y轴偏移，单位dp
setDialogBottom	boolean	设置授权页是否居于底部

6.7 常见问题

1. 首次取号时，为什么 APP 网络通信正常，号码认证一直失败

- 首先检测sim 卡有没有欠费，能不能通过移动网络上网。

- Android 国内某些厂商的系统，wifi 网络权限与移动网络权限是分开管理的，检测下APP是否仅仅只有WLAN 网络权限，而移动网络权限缺失。

## 2. checkEnvAvailable函数返回false

- 检测是否有插入sim卡
- 检测蜂窝网络开关是否开启

## 3. 获取token失败，一般有哪些原因

- 手机设置检测网络制式，中国移动支持2G/3G/4G、中国联通支持3G/4G、中国电信支持4G，但各大运营商网络在4G网络成功率较高。
- 接口超时时间是否过短，建议3000~5000ms。
- SIM是否欠费，是否可以蜂窝网络上网。
- 切换卡的过程中，需要等网络稳定后，再使用认证登录功能。

## 4. Android双卡手机一键登登录过程中，一键登录逻辑是怎么样的

使用默认上网卡进行一键登录认证。

## 5. 权限问题

- 若出现权限相关问题，请检查 APP的权限是否申请正常。正常引用 aar，权限会自动merge。若权限没有 merge，需要添加如下权限。

```
1 <uses-permission
2     android:name="android.permission.INTERNET" />
3 <uses-permission
4     android:name="android.permission.ACCESS_WIFI_STATE" />
5 <uses-permission
6     android:name="android.permission.ACCESS_NETWORK_STATE" />
7 <uses-permission
8     android:name="android.permission.CHANGE_NETWORK_STATE" />
```

## 6. 返回错误码600005，手机终端不安全有哪些原因

1. 手机网络是否连接了代理
2. 手机是否处于hook状态，或者安装了相关的hook框架
3. APP是否处于attached状态
4. 手机是否被root
5. 是否在模拟器环境中运行
6. APP是否处于调试状态，使用 `getReporter.setLoggerEnable(true)` 可以关闭此项检测。

## 7. 非法手机号

针对一些客户在实际使用场景中出现非法手机号的事情，目前和运营商沟通反馈的结果是"试点纯流量卡，未实名制"。

## 8. VPN报错

用户开启VPN后使用sdk做一键登陆的时候联通出现源IP错误，或者电信出现800008错误，移动报103111错误。关闭vpn后再尝试，如果还不行建议打开飞行模式然后关闭。

## 9. 页面非法修改

添加悬浮窗控件遮挡隐私协议、一键登录按钮以及掩码，或者将字体颜色设置为透明，sdk回调600005页面非法修改。

## 10. setSDKAuthSDKInfo的密钥如何获取？

登录阿里云控制台，进入认证方案管理，点击”密钥“进行复制，建议维护在APP服务端。此密钥不是阿里云的AccessKey,AccessSecret！

## 11. 当使用移动卡请求一键登录不成功，出现以下报错信息时。

```
ontokenfaild{"code":"600011","msg":"vendorCode:200025, msg:获得的手机授权码失败: {\\"resultCode\\":\\"200025\\",\\"authType\\":\\"1\\",\\"authTypeDes\\":\\"WIFI下网  
关鉴权\\",\\"resultDesc\\":\\"发生未知错误  
\\"}","requestCode":0,"vendorName":"CMCC"}
```

请检查依赖v4包com.android.support:support-v4版本是否高于25.4.0或者v7包  
com.android.support:appcompat-v7版本是否高于25.4.0

## 12. 当使用移动卡请求一键登录不成功，出现以下报错信息时

```
ontokenfaild{"code":"600011","msg":"vendorCode:200028, msg:获得的手机授权码失败: {\\"resultCode\\":\\"200028\\",\\"authType\\":\\"1\\",\\"authTypeDes\\":\\"网络鉴权\\",\\"resultCode\\":\\"网络异常\\"}","requestCode":0,"vendorName":"CMCC"}
```

- 请检查清单文件application标签下配置了 `android:usesCleartextTraffic="true"` 还配置 `android:networkSecurityConfig` 文件
- 如果设置了 `networkSecurityConfig` 文件，请在 `networkSecurityConfig` 文件里面配置

```
1 <application
2     android:name=".DemoApplication"
3     android:icon="@drawable/ic_launcher"
4     android:label="@string/app_name"
5     android:supportsRtl="true"
6     android:theme="@style/AppTheme"
7     android:usesCleartextTraffic="true"
8     android:networkSecurityConfig="@xml/config"
9     android:requestLegacyExternalStorage="true">
10 </application>
```

```
1 <network-security-config>
2     <domain-config cleartextTrafficPermitted="true">
3         <domain includeSubdomains="true">cmpassport.com</domain>
4     </domain-config>
5 </network-security-config>
```

## 13. 内存泄漏

- Toast内存泄漏
  - a. 协议没勾选点击一键登录按钮显示Toast，退出授权页页出现内存泄漏。解决办法是 `AuthUIConfig.setLogBtnToastHidden(true)` 隐藏默认Toast，根据点击事件的code自己显示Toast即可。
  - b. 页面非法修改

当授权页号码栏、一键登录Button、协议栏出现重叠或者遮挡时点击一键登录按钮，显示Toast页面非法修改。解决办法，查看图层解决重叠即可。

- TokenResultListener内存泄漏

sdk内部会持有外部设置进来的TokenResultListener，在一键登录功能使用完毕之后通过PhoneNumberAuthHelper.setAuthListener(null)将回调置空即可。

## 14. AAR介绍

所有的AAR都是必须！

crashshiled-xx-release.aar：SDK的内部的crash防护和收集库，可以减少SDK的崩溃率，在SDK发生时不会影响接入者的APP。

logger-xx-release.aar：SDK内部用于收集日志信息和监报告警的。

main-xx-release.aar：SDK内部的一些核心工具类，比如线程池，json解析等等。