

# Android v2.2.1.4 免密&本机

---

## 一.准备工作

概述

创建应用

快速体验demo

开发环境搭建

## 二.SDK使用说明

1.初始化

2.预取号

3.拉起授权页

4.销毁授权页

5.协议点击监听

6.置换手机号

## 三.授权页界面修改

1.设计规范

2.授权页配置

3.添加自定义控件

4.设置弹窗样式

5.设置横竖屏

## 四.本机认证

1.初始化

2.本机号校验

3.校验手机号

## 五.返回码

# 一.准备工作

# 概述

本文是闪验SDK\_Android版本的接入文档，用于指导SDK的使用方法，默认读者已经熟悉 IDE（Eclipse 或者 Android Studio）的基本使用方法，以及具有一定的 Android 编程知识基础。

在对接之前您需要花5-10分钟阅读以下条目，可减少对接过程中的问题。

- HBuilder插件集成请移步至「[DCloud市场](#)」
- 常见问题请移步至「[对接问题](#)」
- 崩溃问题请移步至「[崩溃问题](#)」
- 升级版本请移步至「[升级指南](#)」
- 下载SDK资源请移步至「[资源下载](#)」

## 前置条件

- 闪验SDK支持minSdkVersion 16及以上版本
- 闪验SDK支持中国移动3/4G、联通3/4G、电信4G的取号能力，在3G网络下时延会更高
- 闪验SDK支持单数据网络/数据网络与WiFi网络双开，不支持单WiFi网络
- 对于双卡手机，闪验SDK只对当前流量卡取号，双卡均未开数据流量SDK将会返回错误码

# 创建应用

应用的创建流程及APPID/APPKEY的获取，请查看「[账号创建](#)」文档

**注意：如果应用有多个包名或签名不同的马甲包，须创建多个对应包名和签名的应用，否则马甲包将报包名或签名校验不通过。**

# 快速体验demo

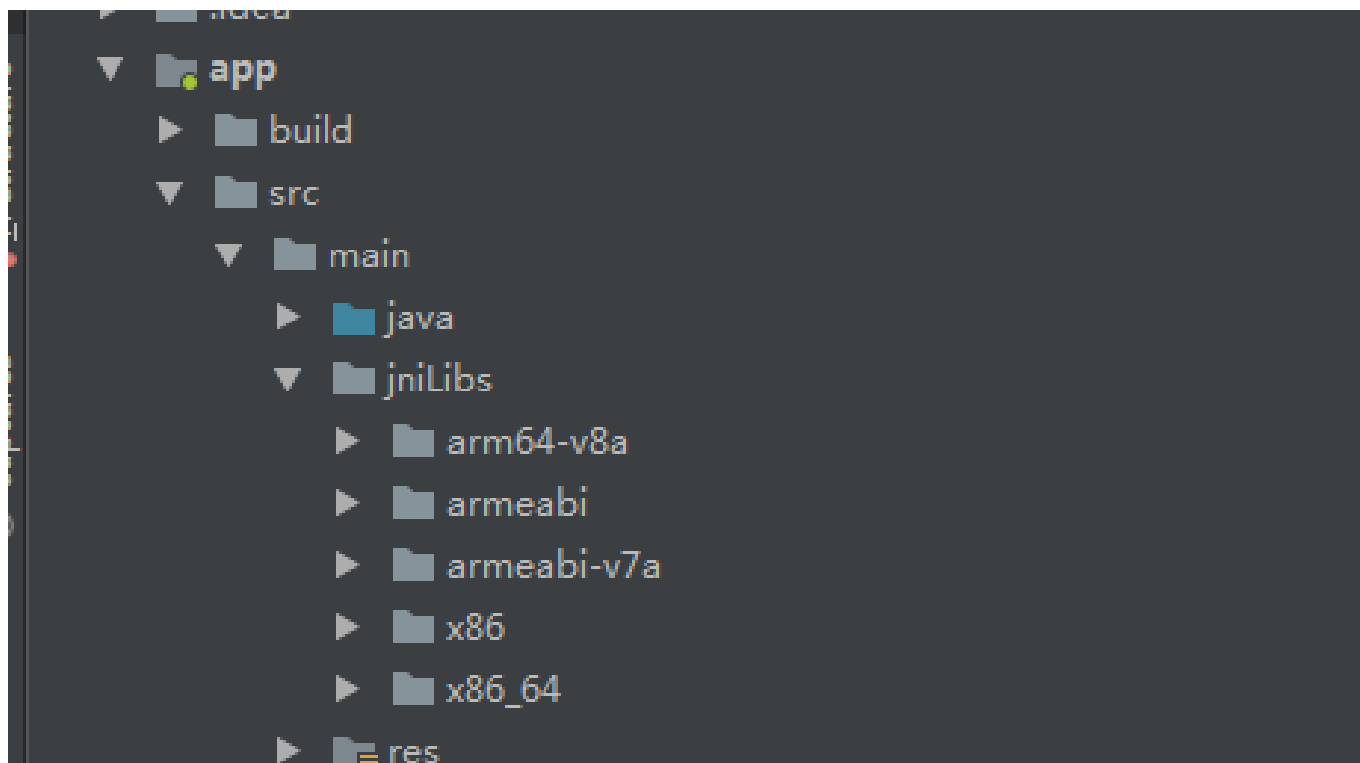
- Android压缩包附带的apk文件夹中是闪验demo的安装包，可以直接安装到Android手机上。并快速体验闪验在您的手机上的表现。
- Android压缩包附带的demo文件夹中是闪验的示例工程，使用Android studio打开示例工程，完成以下步骤配置，然后直接运行起来测试。
  - a.将build里面的applicationId换成对应的测试包名
  - b.将签名配置改成您的签名配置
  - c.将AppId和AppKey换成您在闪验平台创建应用后生成的信息

# 开发环境搭建

### (1) 将开发包拷贝到工程

- a.将SDK中libs目录下的aar包拷贝到自己工程的libs目录下，如没有该目录需新建。
- b.SDK中jniLibs目录下包含多个so库，分别支持armeabi, armeabi-v7a, arm64-v8a, x86, x86\_64等cpu架构，请根据项目情况，选择相应的so库。如果您的项目包含某个abi目录，则复制对应的so文件到您的项目，例如，您的项目中只有armeabi-v7a目录，则只复制jniLibs中的armeabi-v7a文件到您的项目；如果您的项目没有abi目录，请自行创建并复制。

jniLibs所在目录结构如下图：



**注意：**如果您的项目中在build里面指定了 `jniLibs.srcDirs = ['libs']`，则必须将so库放到libs目录下的对应文件中，否则so库会引用不到，导致电信卡报：`"result":80102,"msg":"预登录异常"`。

在app文件夹下的build.gradle的dependencies中配置对应版本的aar依赖并添加repositories，详细代码如下：

```
1 repositories {
2     flatDir {
3         dirs '../app/libs'
4     }
5 }
```

```
6 dependencies {api(name: 'aar包名', ext: 'aar')}
```

## (2) 配置AndroidManifest.xml文件

在manifest标签内添加必要的权限支持

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

建议的权限：如果选用该权限，需要在预取号步骤前提前动态申请。

```
1 <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
```

建议开发者申请本权限，本权限只用于移动运营商在双卡情况下，更精准的获取数据流量卡的运营商类型，  
缺少该权限，存在取号失败概率上升的风险。

### 配置权限说明

权限名称	权限说明	使用说明
INTERNET	允许应用程序联网	用于访问网关和认证服务器
ACCESS_WIFI_STATE	允许访问WiFi网络状态信息	允许程序访问WiFi网络状态信息
ACCESS_NETWORK_STATE	允许访问网络状态	区分移动网络或WiFi网络
CHANGE_NETWORK_STATE	允许改变网络连接状态	设备在WiFi跟数据双开时，强行切换使用数据网络
READ_PHONE_STATE	允许读取手机状态	获取IMSI用于判断双卡和换卡

		(可选)
WRITE_SETTINGS	允许读写系统设置项	6.0以下添加，可增加电信成功率，6.0以上可去除
GET_TASKS	允许应用程序访问TASK	

在application标签内配置授权登录activity

```
1  <!-- *****联通授权页activity*****
   **-->
2  <activity
3      android:name="com.sdk.mobile.manager.login.cucc.OauthActivity"
4      android:launchMode="singleTop"
5      android:configChanges="keyboardHidden|orientation|screenSize"
6      android:screenOrientation="portrait" />
7  <!-- *****移动授权页activity*****
   **-->
8  <activity
9      android:name="com.cmic.sso.sdk.activity.LoginAuthActivity"
10     android:launchMode="singleTop"
11     android:configChanges="keyboardHidden|orientation|screenSize"
12     android:screenOrientation="portrait" />
13 <!-- *****电信授权页activity*****
   **-->
14 <activity
15     android:name="com.chuanglan.shanyan_sdk.view.ShanYanOneKeyActivi
16     ty"
17     android:launchMode="singleTop"
18     android:configChanges="keyboardHidden|orientation|screenSize"
19     android:screenOrientation="portrait" />
20 <!-- *****协议页activity*****--
   >
21 <activity
22     android:name="com.chuanglan.shanyan_sdk.view.CTCCPrivacyProtocol
23     Activity"
24     android:launchMode="singleTop"
25     android:configChanges="keyboardHidden|orientation|screenSize"
26     android:screenOrientation="portrait" />
```

配置Android9.0对http协议的支持两种方式：

方式一：

```
1 android:usesCleartextTraffic="true"
```

示例代码：

```
1 <application
2     android:name=".view.MyApplication"
3     android:allowBackup="true"
4     android:icon="@mipmap/shanyan_sdk_icon"
5     android:label="@string/app_name"
6     android:roundIcon="@mipmap/shanyan_sdk_icon"
7     android:supportsRtl="true"
8     android:theme="@style/AppTheme"
9     android:usesCleartextTraffic="true"
10 ></application>
```

方式二：

目前只有移动运营商个别接口为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：cmpassport.com

(3) 混淆规则：

```
1 -dontwarn com.cmic.sso.sdk.**
2 -dontwarn com.sdk.**
3 -keep class com.cmic.sso.sdk.**{*;}
4 -keep class com.sdk.** { *;}
5 -keep class cn.com.chinatelecom.account.api.**{*;}
```

通过上面的几个步骤，工程就配置完成了，接下来就可以在工程中使用闪验SDK进行开发了。

## 二.SDK使用说明

# 1.初始化

使用一键登录功能前，必须先进行初始化操作，**放在Application的onCreate()方法中（必须放在主线程）。**

方法原型

```
1 public void init(Context context, String appId, String appKey, InitListener initListener){}
```

参数描述

参数	类型	说明
context	Context	<b>必须传ApplicationContext对象</b>
appId	String	闪验平台获取到的appId
appKey	String	闪验平台获取到的appKey
initListener	InitListener	初始化回调监听，getInitStatus是该监听唯一的抽象方法，即void getInitStatus(int code, String result)

示例代码

```
1 OneKeyLoginManager.getInstance().init(getApplicationContext(), appId, appKey, new InitListener() {
2                                     @Override
3                                     public void getInitStatus(int code, String result) {
4                                     }
5                                     });
```

getInitStatus(int code, String result)方法返回参数分为外层code和result，含义如下：

字段	类型	含义
code	Int	code为1022:成功；其他：失败
result	String	返回信息

注意：初始化不要放在申请权限方法内，否则用户拒绝权限会造成后续方法没有回调。

## 2.预取号

- 在初始化成功回调里调用，可缩短拉起授权页时间，该方法会进行网络流量请求。
- 建议在判断当前用户属于未登录状态时使用，已登录状态用户请不要调用该方法
- 建议在执行拉起授权登录页的方法前，提前一段时间调用预取号方法，中间最好有2-3秒的缓冲（因为预取号方法需要1~3s的时间取得临时凭证），比如放在启动页的onCreate（）方法中，或者app启动的application中的onCreate（）方法中去调用，不建议放在用户登录时和拉起授权登录页方法一起调用，会影响用户体验和成功率。
- 请勿频繁的多次调用、请勿与拉起授权登录页同时和之后调用。
- 避免大量资源下载时调用，例如游戏中加载资源或者更新补丁的时候要顺序执行

方法原型：

```
1 public void getPhoneInfo(GetPhoneInfoListener getPhoneInfoListener){}
```

参数描述

参数	类型	说明
getPhoneInfoListene r	GetPhoneInf oListener	预取号回调监听，getPhoneInfoStatus是该监听中唯一的抽象方法，即 void getPhoneInfoStatus(int code, String result)

示例代码：

```
1 OneKeyLoginManager.getInstance().getPhoneInfo(new GetPhoneInfoListene  
r() {  
2     @Override  
3     public void getPhoneInfoStatus(int code, String result) {  
4     }  
5 });
```

getPhoneInfoStatus(int code, String result) 方法返回参数分为外层code和result，含义如下：

字段	类型	含义
----	----	----



code	Int	code为1022：成功；其他：失败
result	String	返回信息

### 3.拉起授权页

- 调用拉起授权页方法后将会调起运营商授权页面。**已登录状态请勿调用。**
- 每次调用拉起授权页方法前均需先调用授权页配置方法，否则授权页可能会展示异常。**
- 3秒之内只能调用一次，而且必须保证上一次拉起的授权页已经销毁再调用，否则SDK会返回请求频繁。

方法原型：

```
1 public void openLoginAuth(boolean isFinish, OpenLoginAuthListener openLoginAuthListener, OneKeyLoginListener oneKeyLoginListener){}
```

参数描述

字段	类型	含义
isFinish	boolean	点击授权页一键登录按钮有回调时是否自动销毁授权页： true：自动销毁 false:不自动销毁，开发者需主动调用销毁授权页方法进行授权页销毁操作
openLoginAuthListener	OpenLoginAuthListener	拉起授权页监听，getOpenLoginAuthStatus是该接口中唯一的抽象方法，即 void getOpenLoginAuthStatus(int code,String result)
oneKeyLoginListener	OneKeyLoginListener	一键登录监听，getPhoneCode是该接口中唯一的抽象方法，即 void getOneKeyLoginStatus(int code,String result)

示例代码

```
1 OneKeyLoginManager.getInstance().openLoginAuth(false, new OpenLoginAuthListener() {
2     @Override
3     public void getOpenLoginAuthStatus(int code, String result) {
4     }
5 }, new OneKeyLoginListener() {
```

```

6      @Override
7      public void getOneKeyLoginStatus(int code, String result) {
8      }
9  });

```

getOpenLoginAuthStatus(int code,String result)方法返回参数分为外层code和result，含义如下：

字段	类型	含义
code	int	code为1000：授权页成功拉起 其他：失败
result	String	返回信息

getOneKeyLoginStatus(int code,String result)方法返回参数分为外层code和result，含义如下：

字段	类型	含义
code	int	code为1000：成功；其他：失败
result	String	返回信息

当外层code为1000时，result的返回为

```

1  {
2  "appId": "",
3  "accessToken": "",
4  "telecom": "",
5  "timestamp": "",
6  "randoms": "",
7  "version": "",
8  "sign": "",
9  "device": ""
10 }

```

含义如下：

字段	类型	含义
appId	String	当前项目的appid
accessToken	String	用来和后台置换手机号。一次有效，有效期3min
telecom	String	当前数据流量卡的运营商类型： CMCC 移动 CTCC 电信

		CUCC 联通
timestamp	String	网络时间戳 有效期3min
randoms	String	随机数
version	String	后台接口版本号
sign	String	签名
device	String	设备型号

注意：返回字段中的accessToken、sign、device字段值为base64字符串，其中包含特殊字符，在客户APP与客户后台交互时应避免使用URL拼接字符串的传值方式，会导致特殊字符丢失，最终导致请求闪验后台的置换手机号接口报“请求非法，签名校验不通过”。

## 4. 销毁授权页

### A. 授权页面自动销毁

1. 在授权登录页面，当用户主动点击左上角返回按钮时，返回码为1011，SDK将自动销毁授权页；
2. 安卓 SDK，当用户点击手机的硬件返回键（相当于取消登录），返回码为1011，SDK将自动销毁授权页
3. 当用户设置一键登录或者其他自定义控件为自动销毁时，得到回调后，授权页面会自动销毁

### B. 授权页手动销毁

1. 当设置一键登录为手动销毁时，点击授权页一键登录按钮成功获取token不会自动销毁授权页，**请务必在回调中处理完自己的逻辑后手动调用销毁授权页方法。**
2. 当设置自定义控件为手动销毁时，**请务必在回调中处理完自己的逻辑后手动调用销毁授权页方法。**

方法原型

```
1 public void finishAuthActivity() {}
```

示例代码

```
1 OneKeyLoginManager.getInstance().finishAuthActivity();
```

**注意：**点击授权页的返回按钮或者自定义控件时，请不要回调或者返回到的界面主线程中做耗时操作，避免堵塞授权页销毁，导致再次调用拉起授权页时返回请求频繁。

## 5. 协议点击监听

需要对协议点击事件监听的用户，可调用此方法监听协议点击事件，无此需求可以不写。

方法原型

```
1 public void setOnClickPrivacyListener(OnClickPrivacyListener onClickPrivacyListener) {}
```

参数描述

参数	类型	说明
onClickPrivacyListener	OnClickPrivacyListener	协议点击事件回调监听，getOnClickPrivacyStatus是该监听唯一的抽象方法，即void getOnClickPrivacyStatus(int code, String result,String operator)

示例代码

```
1 OneKeyLoginManager.getInstance().setOnClickPrivacyListener(new OnClickPrivacyListener() {
2     @Override
3     public void getOnClickPrivacyStatus(int code, String result, String operator) {
4
5     }
6 });
```

getOnClickPrivacyStatus(int code, String result,String operator)方法返回参数含义如下：







字段	类型	含义
code	Int	code为0： 点击运营商协议 code为1： 点击自定义协议1 code为2： 点击自定义协议2
result	String	协议名称
operator	String	对应授权页的运营商类型： CMCC: 移动 CUCC: 联通 CTCC: 电信

6.置换手机号

当一键登录外层code为1000时，您将获取到返回的参数，请将这些参数传递给后端开发人员，并参考「[服务端](#)」文档来实现获取手机号码的步骤。

三.授权页界面修改

1.设计规范

闪验样式	Andriod规范	iOS规范
<div><div></div><div></div><div></div><div></div></div>	<div><div><p>1.状态栏</p><ul style="list-style-type: none"><li>只可修改文字黑白显示</li></ul></div><div><p>2.导航栏</p><p>存在返回键，可换图片，可隐藏</p><p>可调整返回按钮左右显示位置 and 大小</p><p>可添加自定义控件</p><p>可修改自定义登录文字内容以及文字颜色，可隐藏</p><p>可隐藏，可透明，背景颜色可修改</p></div><div><p>3.LOGO/手机号</p><ul style="list-style-type: none"><li>可更换LOGO图片，尺寸可自调，默认70dp</li><li>LOGO可隐藏，可上下左右调整</li><li>可修改本机号码文字颜色、大小，可上下左右调整</li><li>可修改号码隐藏宽度，不允许隐藏手机号</li></ul></div><div><p>4.运营商slogan</p><ul style="list-style-type: none"><li>可修改文字颜色、可上下左右调整</li></ul></div><div><p>5.按钮</p><div><p>5.1 一键登录按钮</p></div><div><p>5.2 其他方式登录</p><ul style="list-style-type: none"><li>默认隐藏，可通过添加自定义控件方式添加</li></ul></div></div><div><p>6.背景</p><ul style="list-style-type: none"><li>默认白底，可通过图片方式修改；全屏显示</li></ul></div><div><p>7.底部协议/授权信息</p><p>同意《<a href="#">天翼服务及隐私协议</a>》和《<a href="#">用户自定义协议条款</a>》、《<a href="#">用户服务协议</a>》并授权闪验获取本机号码</p><ul style="list-style-type: none"><li>可添加两个协议，协议内容带书名号，无法隐藏</li><li>默认居中</li><li>可修改颜色，可上下左右调整</li><li>默认有复选框，可隐藏</li><li>可设置是否默认勾选，可设置选中或未选中时的图片</li><li>底部授权信息（并授权闪验获取本机号码）默认为获取应用Application配置的label字段，可修改</li><li>协议以外描述文字可自定义设置“登录即同意”“和”、“”“并授权XXX使用本机号码验证”</li><li>协议文字可设置对其方式</li><li>文字大小可修改</li></ul></div></div>	<div><div><p>1.状态栏</p><ul style="list-style-type: none"><li>可隐藏、修改文字颜色（黑或白），不可添加自定义控件</li></ul></div><div><p>2.导航栏</p><p>存在返回键，可换图片，可隐藏</p><p>可调整返回按钮左右显示位置 and 大小</p><p>可添加自定义控件在左右两侧</p><p>可修改自定义登录文字内容以及文字颜色，可隐藏</p><p>可隐藏，可透明，背景颜色可修改也可通过图片方式</p></div><div><p>3.LOGO/手机号</p><ul style="list-style-type: none"><li>可更换LOGO图片，尺寸可自调</li><li>LOGO可隐藏，可上下左右调整，可圆角</li><li>可修改本机号码字体、大小、颜色，可上下左右调整</li><li>可修改号码隐藏宽度，不允许隐藏手机号</li></ul></div><div><p>4.运营商slogan</p><ul style="list-style-type: none"><li>可修改字体、颜色、对齐方式，可上下左右调整</li><li>可修改slogan隐藏宽度</li></ul></div><div><p>5.按钮</p><div><p>5.1 一键登录按钮</p></div><div><p>5.2 其他方式登录</p><ul style="list-style-type: none"><li>默认隐藏，可通过添加自定义控件方式添加</li></ul></div></div><div><p>6.背景</p><ul style="list-style-type: none"><li>默认白底，可直接设置背景图片：全屏显示</li></ul></div><div><p>7.底部协议/授权信息</p><p>同意<a href="#">中国移动认证服务条款</a>和<a href="#">测试连接A、测试连接B</a>并授权闪验获得本机号码</p><ul style="list-style-type: none"><li>可添加两个协议，无法隐藏，默认居中，第二行可设置对齐方式</li><li>可修改字体、颜色、大小，可上下调整</li><li>默认有勾选的复选框，尺寸15dp，可隐藏</li><li>可设置是否默认勾选，可设置选中或未选中时的图片</li><li>底部授权信息（并授权闪验获取本机号码）默认为获取应用Application配置的label字段</li><li>协议以外描述文字可自定义设置“登录即同意”“和”、“”“并授权XXX使用本机号码验证”</li></ul></div></div>
	准则	
	1.不允许隐藏手机号	2.不允许隐藏登录按钮
	3.不允许隐藏运营商协议	4.支持竖屏全屏、横屏全屏、弹窗

准则

1.不允许隐藏手机号 2.不允许隐藏登录按钮 3.不允许隐藏运营商协议 4.支持竖屏全屏、横屏全屏、弹窗

注意：开发者不得通过任何技术手段，将授权页面的隐私栏、品牌露出内容隐藏、覆盖，对于接入闪验SDK并上线的应用，我方和运营商会做授权页面做审查，如果有出现未按要求设计授权页面，我方有权将应用的登录功能下线。

## 2.授权页配置

调用该方法可实现对三网运营商授权页面个性化设计，**每次调用拉起授权页方法前必须先调用该方法**，否则授权界面会展示异常。（**三网界面配置内部实现逻辑不同，请务必使用移动、联通、电信卡分别测试三网界面**）

方法原型

```
1 public void setAuthThemeConfig (ShanYanUIConfig shanYanUIConfig) {}
```

参数说明

参数	参数类型	说明
shanYanUIConfig	ShanYanUIConfig	主题配置对象，开发者在 ShanYanUIConfig.java类中调用对应的方法配置授权页中对应的元素

示例代码

```
1 //自定义运营商授权页界面
2 OneKeyLoginManager.getInstance().setAuthThemeConfig(ConfigUtils.getUIConfig(getApplicationContext()));
```

ShanYanUIConfig.java配置元素说明

方法	参数类型	说明
setAuthBGImgPath	Drawable	设置授权页背景图片

授权页导航栏

方法	参数类型	说明
setNavColor	int	设置导航栏背景颜色
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（单位sp）	设置导航栏标题文字大小
setNavReturnImgPath	Drawable	设置导航栏返回按钮图标

setNavReturnImgHidden	boolean	设置导航栏返回按钮是否隐藏（true：隐藏；false：不隐藏）
setNavReturnBtnWidth	int（单位dp）	设置导航栏返回按钮宽度
setNavReturnBtnHeight	int（单位dp）	设置导航栏返回按钮高度
setNavReturnBtnOffsetRightX	int（单位dp）	设置导航栏返回按钮距离屏幕右侧X偏移
setNavReturnBtnOffsetLeftX	int（单位dp）	设置导航栏返回按钮距离屏幕左侧X偏移
setNavReturnBtnOffsetY	int（单位dp）	设置导航栏返回按钮距离屏幕上侧Y偏移
setAuthNavHidden	boolean	设置导航栏是否隐藏（true：隐藏；false：不隐藏）
setAuthNavTransparent	boolean	设置导航栏是否透明（true：透明；false：不透明）

#### 授权页logo

方法	参数类型	说明
setLogoImagePath	Drawable	设置logo图片
setLogoWidth	int（单位dp）	设置logo宽度
setLogoHeight	int（单位dp）	设置logo高度
setLogoOffsetY	int（单位dp）	设置logo相对于标题栏下边缘y偏移
setLogoOffsetBottomY	int（单位dp）	设置logo相对于屏幕底部y偏移
setLogoHidden	boolean	设置logo是否隐藏（true：隐藏；false：不隐藏）
setLogoOffsetX	int（单位dp）	设置logo相对屏幕左侧X偏移

#### 授权页号码栏

方法	参数类型	说明
setNumberColor	int	设置号码栏字体颜色

setNumFieldOffsetY	int（单位dp）	设置号码栏相对于标题栏下边缘y偏移
setNumFieldOffsetBottomY	int（单位dp）	设置号码栏相对于屏幕底部y偏移
setNumFieldWidth	int（单位dp）	设置号码栏宽度
setNumFieldHeight	int（单位dp）	设置号码栏高度
setNumberSize	int(单位sp)	设置号码栏字体大小
setNumFieldOffsetX	int（单位dp）	设置号码栏相对屏幕左侧X偏移

#### 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
setLogBtnImgPath	Drawable	设置授权登录按钮图片
setLogBtnOffsetY	int（单位dp）	设置登录按钮相对于标题栏下边缘Y偏移
setLogBtnOffsetBottomY	int（单位dp）	设置登录按钮相对于屏幕底部Y偏移
setLogBtnTextSize	int（单位sp）	设置登录按钮字体大小
setLogBtnHeight	int（单位dp）	设置登录按钮高度
setLogBtnWidth	int（单位dp）	设置登录按钮宽度
setLogBtnOffsetX	int（单位dp）	设置登录按钮相对屏幕左侧X偏移

#### 授权页隐私栏

方法	参数类型	说明
setAppPrivacyOne	string（链接）	设置开发者隐私条款1名称和URL 参数1：名称 参数2：url
setAppPrivacyTwo	string（链接）	设置开发者隐私条款2名称和URL 参数1：名称 参数2：url
setPrivacyTextSize	int（单位sp）	设置隐私栏字体大小
setAppPrivacyColor	int	设置隐私条款名称颜色 参数1：基础文字颜色 参数2：协议文字颜色



setPrivacyOffsetBottomY	int（单位dp）	设置隐私条款相对于授权页面底部下边缘y偏移
setPrivacyOffsetY	int（单位dp）	设置隐私条款相对于授权页面标题栏下边缘y偏移
setPrivacyOffsetX	int（单位dp）	设置隐私条款相对屏幕左侧X偏移
setPrivacyOffsetGravityLeft	boolean	设置隐私条款文字左对齐（true：左对齐；false：居中）
setPrivacyState	boolean	设置隐私条款的CheckBox是否选中（true：选中；false：未选中）
setUncheckedImgPath	Drawable	设置隐私条款的CheckBox未选中时图片
setCheckedImgPath	Drawable	设置隐私条款的CheckBox选中时图片
setCheckBoxHidden	boolean	设置隐私条款的CheckBox是否隐藏（true：隐藏；false：不隐藏）
setPrivacyText	String	设置隐私条款名称外的文字，包含四个参数，参数示例（红色 字体为传入参数）： 同意《***》和《***》、《***》并授权闪验获取本机号码 参数1: 同意 参数2: 和 参数3: 、 参数4: 并授权闪验获取本机号码

#### 授权页slogan

方法	参数类型	说明
setSloganTextColor	int	设置slogan文字颜色
setSloganTextSize	int（单位sp）	设置slogan文字字体大小
setSloganOffsetY	int（单位dp）	设置slogan相对于标题栏下边缘y偏移
setSloganHidden	boolean	设置slogan是否隐藏（true：隐藏；false：不隐藏）
setSloganOffsetBottomY	int（单位dp）	设置slogan相对屏幕底部Y偏移
setSloganOffsetX	int（单位dp）	设置slogan相对屏幕左侧X偏移

#### 授权页loading

方法	参数类型	说明
setLoadingView	View	设置自定义loading

注意：

a.控件X偏移如果不设置默认居中。

### 3.添加自定义控件

调用该方法可实现在授权页添加自定义控件。

方法原型

```
1 public ShanYanUIConfig.Builder addCustomView(View view, boolean isFinish, boolean type, ShanYanCustomInterface shanYanCustomInterface) {}
```

参数说明

参数	参数类型	说明
view（必填）	View	自定义控件对象
isFinish（必填）	boolean	是否需要销毁授权页：true销毁 false不销毁
type（必填）	boolean	设置自定义控件的位置： true为授权页导航栏 false为授权页空白处
shanYanCustomInterface	ShanYanCustomInterface	自定义控件监听

注意：如果添加布局为自定义控件，监听实现请参考demo示例。

示例代码：

```
1 /**
2     * 闪验三网运营商授权页配置类
3     *
4     * @param context
5     * @return
6     */
7 public static ShanYanUIConfig getUiConfig(Context context) {
8
9     /*****自定义控件**
10    *****/
11 }
```

```

11         Button close = new Button(context);
12         close.setBackgroundResource(context.getResources().getIdent
            ifier("close_black", "drawable", context.getPackageName()));
13         RelativeLayout.LayoutParams mLayoutParamsClose = new Relati
            veLayoutParams(RelativeLayout.LayoutParams.WRAP_CONTENT, Rel
            ativeLayout.LayoutParams.WRAP_CONTENT);
14         mLayoutParamsClose.setMargins(0, AbScreenUtils.dp2px(context
            t, 15), AbScreenUtils.dp2px(context, 10), 0);
15         mLayoutParamsClose.width = AbScreenUtils.dp2px(context, 20)
            ;
16         mLayoutParamsClose.height = AbScreenUtils.dp2px(context, 20
            );
17         mLayoutParamsClose.addRule(RelativeLayout.ALIGN_PARENT_RIGH
            T);
18         close.setLayoutParams(mLayoutParamsClose);
19         //其他方式登录
20         TextView otherTV = new TextView(context);
21         otherTV.setText("其他方式登录");
22         otherTV.setTextColor(0xff3a404c);
23         otherTV.setTextSize(TypedValue.COMPLEX_UNIT_SP, 13);
24         RelativeLayout.LayoutParams mLayoutParams1 = new RelativeLa
            yout.LayoutParams(RelativeLayout.LayoutParams.WRAP_CONTENT, Relativ
            eLayoutParams.WRAP_CONTENT);
25         mLayoutParams1.setMargins(0, AbScreenUtils.dp2px(context, 1
            75), 0, 0);
26         mLayoutParams1.addRule(RelativeLayout.CENTER_HORIZONTAL);
27         otherTV.setLayoutParams(mLayoutParams1);
28
29         View view = new View(context);
30         view.setBackgroundColor(0xffe8e8e8);
31         RelativeLayout.LayoutParams mLayoutParams3 = new RelativeLa
            yout.LayoutParams(RelativeLayout.LayoutParams.MATCH_PARENT, AbScree
            nUtils.dp2px(context, 1));
32         mLayoutParams3.setMargins(0, AbScreenUtils.dp2px(context, 4
            4), 0, 0);
33         view.setLayoutParams(mLayoutParams3);
34
35         TextView tip = new TextView(context);
36         tip.setText("使用该手机号登录");
37         tip.setTextColor(0xff1E3179);

```

```

38         tip.setTextSize(TypedValue.COMPLEX_UNIT_SP, 15);
39         RelativeLayout.LayoutParams mLayoutParamsTip = new Relative
Layout.LayoutParams(RelativeLayout.LayoutParams.WRAP_CONTENT, Relat
iveLayout.LayoutParams.WRAP_CONTENT);
40         mLayoutParamsTip.setMargins(AbScreenUtils.dp2px(context, 15
), AbScreenUtils.dp2px(context, 65), 0, 0);
41         mLayoutParamsTip.addRule(RelativeLayout.ALIGN_PARENT_LEFT);
42         tip.setLayoutParams(mLayoutParamsTip);
43
44         LayoutInflater inflater1 = LayoutInflater.from(context);
45         RelativeLayout relativeLayout = (RelativeLayout) inflater1.
inflate(R.layout.other_login_item_view, null);
46         RelativeLayout.LayoutParams layoutParamsOther = new Relativ
eLayout.LayoutParams(RelativeLayout.LayoutParams.WRAP_CONTENT, Rela
tiveLayout.LayoutParams.WRAP_CONTENT);
47         layoutParamsOther.addRule(RelativeLayout.CENTER_HORIZONTAL)
;
48         layoutParamsOther.addRule(RelativeLayout.ALIGN_PARENT_BOTTO
M);
49         relativeLayout.setLayoutParams(layoutParamsOther);
50         ImageView weixin = relativeLayout.findViewById(R.id.weixin)
;
51         ImageView qq = relativeLayout.findViewById(R.id.qq);
52         ImageView weibo = relativeLayout.findViewById(R.id.weibo);
53         weixin.setOnClickListener(new View.OnClickListener() {
54             @Override
55             public void onClick(View v) {
56                 Toast.makeText(context, "微信登录", Toast.LENGTH_SHOR
T).show();
57             }
58         });
59         qq.setOnClickListener(new View.OnClickListener() {
60             @Override
61             public void onClick(View v) {
62                 Toast.makeText(context, "qq登录", Toast.LENGTH_SHORT
).show();
63
64             }
65         });
66         weibo.setOnClickListener(new View.OnClickListener() {

```

```

67         @Override
68         public void onClick(View v) {
69             Toast.makeText(context, "微博登录", Toast.LENGTH_SHOR
T).show();
70
71         }
72     });
73
74     /*****设置授权
页*****/
75     Drawable authNavHidden = context.getResources().getDrawable
(R.drawable.sysdk_login_bg);
76     Drawable navReturnImgPath = context.getResources().getDrawa
ble(R.drawable.sy_sdk_left);
77     Drawable logoImgPath = context.getResources().getDrawable(R
.drawable.sy_logo);
78     Drawable logBtnImgPath = context.getResources().getDrawable
(R.drawable.login_btn_bg);
79     Drawable uncheckedImgPath = context.getResources().getDrawa
ble(R.drawable.umcsdk_uncheck_image);
80     Drawable checkedImgPath = context.getResources().getDrawabl
e(R.drawable.umcsdk_check_image);
81     ShanYanUIConfig uiConfig = new ShanYanUIConfig.Builder()
82         .setDialogTheme(true, 460, 240, 0, 0, false)
83         //授权页导航栏：
84         .setNavColor(Color.parseColor("#ffffff")) //设置导航
栏颜色
85         .setNavText("") //设置导航栏标题文字
86         .setNavTextColor(0xff080808) //设置标题栏文字颜色
87         .setNavReturnImgPath(navReturnImgPath) //
88         .setNavReturnImgHidden(false)
89         .setAuthBGImgPath(authNavHidden)
90         .setAuthNavHidden(true)
91
92         //授权页logo（logo的层级在次底层，仅次于自定义控件）
93         .setLogoImgPath(logoImgPath) //设置logo图片
94         .setLogoWidth(108) //设置logo宽度
95         .setLogoHeight(36) //设置logo高度
96         .setLogoOffsetY(5) //设置logo相对于标题栏下边缘y偏移
97         .setLogoOffsetX(15)

```

```

98         .setLogoHidden(false)    //是否隐藏logo
99
100        //授权页号码栏：
101        .setNumberColor(0xff333333)    //设置手机号码字体颜色
102        .setNumFieldOffsetY(100)    //设置号码栏相对于标题栏下边
    缘y偏移
103        .setNumberSize(18)
104        .setNumFieldWidth(110)
105        .setNumFieldOffsetX(15)
106
107
108        //授权页登录按钮：
109        .setLogBtnText("本机号码一键登录")    //设置登录按钮文字
110        .setLogBtnTextColor(0xffffffff)    //设置登录按钮文字颜
    色
111        .setLogBtnImgPath(logBtnImgPath)    //设置登录按钮图片
112        .setLogBtnOffsetY(93)    //设置登录按钮相对于标题栏下边缘y
    偏移
113        .setLogBtnTextSize(15)
114        .setLogBtnWidth(150)
115        .setLogBtnHeight(45)
116
117        //授权页隐私栏：
118        .setAppPrivacyOne("用户自定义协议条款", "https://www.25
    3.com")    //设置开发者隐私条款1名称和URL(名称, url)
119        .setAppPrivacyTwo("用户服务条款", "https://www.253.co
    m")    //设置开发者隐私条款2名称和URL(名称, url)
120        //.setAppPrivacyColor(0xff666666, 0xff0085d0)    //
    设置隐私条款名称颜色(基础文字颜色, 协议文字颜色)
121        .setPrivacyOffsetY(156)
122        .setUncheckedImgPath(uncheckedImgPath)
123        .setCheckedImgPath(checkedImgPath)
124
125        //授权页slogan：
126        .setSloganTextColor(0xff999999)    //设置slogan文字颜色
127        .setSloganOffsetY(130)    //设置slogan相对于标题栏下边缘y
    偏移
128        .setSloganTextSize(9)
129        .setSloganOffsetX(15)
130

```

```

131          // 添加自定义控件：
132          //其他方式登录及监听，可以不写
133          .addCustomView(close, true, false, new ShanYanCustomInterface() {
134              @Override
135              public void onClick(Context context, View view)
136              {
137                  Toast.makeText(context, "点击关闭", Toast.LENGTH_SHORT).show();
138              }
139          })
140          //标题栏下划线，可以不写
141          .addCustomView(view, false, false, null)
142          .addCustomView(tip, false, false, null)
143          .addCustomView(relativeLayout, false, false, null)
144          .build();
145          return uiConfig;
146      }

```

## 4.设置弹窗样式

调用该方法可实现将授权页设置成弹窗样式。

方法原型

```

1 public ShanYanUIConfig.Builder setDialogTheme(boolean isdialogTheme,
2         int dialogWidth, int dialogHeight, int dialogX, int dialogY ,boolean
3         isDialogBottom) {}

```

参数说明

参数	参数类型	说明
isdialogTheme（必填）	boolean	是否用弹窗样式：true 弹窗样式 false 非弹窗样式
dialogWidth（必填）	int（单位dp）	授权页弹窗宽度
dialogHeight（必填）	int（单位dp）	授权页弹窗高度
dialogX	int（单位dp）	授权页弹窗X偏移量（以屏幕中心为原点）

dialogY	int (单位dp)	授权页弹窗Y偏移量（以屏幕中心为原点）
isDialogBottom	boolean	授权页弹窗是否贴于屏幕底部： true：显示到屏幕底部，dialogY参数设置将无效 false：不显示到屏幕底部，以dialogY参数为准

**注意：**设置弹窗效果背景的透明度需要在Manifest.xml里面配置授权界面样式

样式示例：

#### 1.为授权界面的activity设置弹窗theme主题

```

1      <activity
2          android:name="com.sdk.mobile.manager.login.cucc.OauthActi
        vity"
3          android:launchMode="singleTop"
4          android:theme="@style/Theme.ActivityDialogStyle"
5          android:configChanges="keyboardHidden|orientation|screenS
        ize"
6          android:screenOrientation="behind" />

```

#### 2.设置theme主题的style样式

```

1 <style name="Theme.ActivityDialogStyle" parent="Theme.AppCompat.Ligh
    t.NoActionBar">
2     <!--背景透明-->
3     <item name="android:windowBackground">@android:color/transparent
        </item>
4     <item name="android:windowIsTranslucent">true</item>
5     <!--dialog的整个屏幕的背景是否有遮障层-->
6     <item name="android:backgroundDimEnabled">true</item>
7 </style>

```

## 5.设置横竖屏

全屏模式：



在manifest.xml文件中配置授权页的activity方向，例如：“android:screenOrientation="portrait"”。全屏模式下，授权页只能指定方向，不支持横竖屏切换。

弹窗模式：

targetSdkVersion 26以上：  
必须在manifest.xml文件中配置调用拉起授权页方法的activity方向，例如：“android:screenOrientation="portrait"”。弹窗模式下，必须指定前一个页面的方向，不支持横竖屏切换。不能配置授权页的方向，否则在8.0系统版本上会报“Only fullscreen opaque activities can request orientation”异常。  
targetSdkVersion 26及以下：  
在manifest.xml文件中配置授权页的activity方向，例如：“android:screenOrientation="portrait"”。授权页只能指定方向，不支持横竖屏切换。

注意：当targetSdkVersion>26时，Android8.0版本如果是透明的Activity，不能指定方向，否则会报“Only fullscreen opaque activities can request orientation”异常。

## 四.本机认证

注：本机认证同免密登录，需要初始化，本机认证、免密登录可共用初始化，两个功能同时使用时，只需调用一次初始化即可。

### 1.初始化

同SDK使用说明-->初始化

### 2.本机号校验

在本地初始化和网络初始化执行之后调用，本机号校验界面需自行实现，可以在多个需要校验的页面中调用。  
注意：该步骤为获取的token，需要将返回的参数原封不动的传到后台接口进行校验，通过接口返回确定是否为本机号码，具体请看第三节校验手机号。

方法原型

```
1 public void startAuthentication(String phone, AuthenticationExecuteListener authenticationExecuteListener) {}
```

参数描述

参数	类型	说明
		需要校验的手机号

phone (必填)	String	
authenticationExecuteListener (必填)	AuthenticationExecuteListener	本机号校验回调监听器，需要调用者自己实现； AuthenticationExecuteListener 是接口中的本机号校验参数回调接口，authenticationRespond是该接口中唯一的抽象方法，即  void authenticationRespond(int code,String result)

示例代码

```
1  OneKeyLoginManager.getInstance().startAuthentication(phone, new AuthenticationExecuteListener() {
2
3      @Override
4      public void authenticationRespond(int code, String result) {
5
6      }
7  });
```

authenticationRespond(int code,String result)方法返回参数分为外层code和result，含义如下：

字段	类型	含义
code	int	code为2000：成功 其他：失败
result	String	返回信息

当外层code为2000时，result的返回为

```
1  {
2      "appId": "",
3      "accessCode": "",
4      "mobile": "",
5      "telecom": "",
6      "timestamp": "",
7      "randoms": "",
8      "sign": "",
9      "version": "",
10     "device": ""
11 }
```

含义如下：

字段	类型	含义
appId	String	当前项目的appid
accessToken	String	用来和后台校验手机号。一次有效，有效期3min
mobile	String	输入的手机号
telecom	String	当前数据流量卡的运营商类型： CMCC 移动 CTCC 电信 CUCC 联通
timestamp	String	网络时间戳 有效期3min
randoms	String	随机数
sign	String	签名
version	String	后台接口版本号
device	String	设备型号

注意：返回字段中的accessToken、sign字段值为base64字符串，其中包含特殊字符，在客户APP与客户后台交互时应避免使用URL拼接字符串的传值方式，会导致特殊字符丢失，最终导致请求闪验后台的校验手机号接口报“请求非法，签名校验不通过”。

### 3.校验手机号

当本机号校验外层code为2000时，您将获取到返回的参数，请将这些参数传递给后端开发人员，并参考「[服务端](#)」文档来实现校验本机号的步骤

## 五.返回码

该返回码为闪验SDK自身的返回码，请注意1003及1023错误内均含有运营商返回码，具体错误在碰到之后查阅「[返回码](#)」

返回码	返回码描述
1000	一键登录成功，解析result，可得到网络请求参数
1001	运营商通道关闭
1002	运营商信息获取失败
1003	一键登录获取token失败

1007	网络请求失败
1011	点击返回，用户取消免密登录
1014	SDK内部异常
1016	APPID为空
1017	APPKEY为空
1019	其他错误
1020	非三大运营商，无法使用一键登录功能
1021	运营商信息获取失败（accessToken失效）
1022	网络初始化、预取号成功
1023	预初始化失败
1025	非联通号段（目前联通号段 46001 46006 46009）
1031	请求过于频繁
2000	本机号校验返回成功，解析result，可得到网络请求参数
2001	传入的手机号phone为空
2003	本机号校验返回失败
2020	非三大运营商