

# 概述与资源

本文是 OneLogin HarmonyOS SDK 的部署文档，用于指导 OneLogin HarmonyOS SDK 的集成，读者需具有一定 HarmonyOS ArkTs 编程知识基础。

## 1、环境需求

条目	资源
开发目标	HarmonyOS Next
开发环境	DevEco Studio 6.0.1 Release
HarmonyOS SDK 版本	API Version 20 Release
包增量	1M
网络制式	移动 2G/3G/4G/5G，联通 3G/4G/5G，电信 4G/5G（2G/3G 网络下时延相对较高，成功率相对较低）
网络环境	打开蜂窝数据流量并且给予应用蜂窝数据权限

## 2、相关开发资料

条目	说明
SDK名称	-一键认证
开发者信息	北京亿美软通科技有限公司
最新版本号	<b>v1.1.6</b>
最新版本发布时间	2026/01/27
主要功能	直连三大运营商网关取号服务，帮助企业优化验证流程，助力运营拉新、留存、促活。
SDK隐私政策	<a href="#">一键认证-隐私政策</a>
SDK合规指南	<a href="#">一键认证-合规指南</a>
产品结构流程	<a href="#">交互流程</a> ， <a href="#">通讯流程</a>
常见问题	<a href="#">常见问题</a>
资源链接	<a href="#">登录后台获取 SDK</a> 登录后，点击右上角账号>SDK 下载中心>选择对应产品 SDK

# 准备工作

## 1、创建应用

登录后台创建应用获取 APPID 和 APPKEY，具体步骤可参照账号创建。

## 2、快速体验 Demo

HarmonyOS 压缩包附带的 demo 文件夹中是的示例工程，使用 DevEco Studio 打开示例工程，直接运行起来即可进行快速体验测试。

如果开发者需要将 SDK 集成到自己的项目进行体验，可完成以下配置步骤进行本地测试：

1. 将包名修改成对应的测试包名；
2. 将签名配置改成您的签名配置；
3. 将 APPID 换成您在管理后台创建生成的信息(需注意 APPID 和包名/包签名的一致性)；
4. 参照服务端接入文档完成服务端接口的对接，注意客户端 APPID 与服务端保持一致。如未完成该步骤则仅能体验APP端功能，不能获取真实手机号。

## 3、开发环境搭建

### 3.1、手动集成 SDK

导入 SDK 到项目工程并配置开发环境

1. 在 sdk 目录下，将获取的 geetest\_onelogin\_android\_vx.x.x.x\_xxxxxxx.har 文件拖拽到工程中的 libs 文件夹下。
2. 在拖入 .har 到 libs 文件夹后，还要检查 .har 是否被 install，要在项目的 oh-package.json5 下添加如下代码：

```
"dependencies": {  
  "@geetest/onelogin":  
  "file:./libs/geetest_onelogin_android_vx.x.x.x_xxxxxxx.har"  
}
```

3. module.json5 权限配置：

SDK 已默认配置必要的权限，正常情况下直接集成即可。

必要的权限包含：

```
"requestPermissions": [
  {
    "name": "ohos.permission.GET_NETWORK_INFO"
  },
  {
    "name": "ohos.permission.INTERNET"
  },
  {
    "name": "ohos.permission.SET_NETWORK_INFO"
  }
]
```

权限说明：

权限名称	权限说明	使用说明
ohos.permission.INTERNET	允许应用程序联网	用于访问、运营商网关和认证服务器
ohos.permission.GET_NETWORK_INFO	允许应用获取数据网络信息	允许 SDK 判断当前网络处于移动网络或WiFi网络
ohos.permission.SET_NETWORK_INFO	允许应用配置网络连接状态	设备在WiFi跟数据双开时，允许 SDK 内部请求（仅用于网关取号的请求） 强行走数据网络通道

模块配置：

请设置 `useNormalizedOHMUrl` 为true，查看工程级build-profile.json5：

```
{
  "app": {
    "products": [
      {
        "buildOption": {
          "strictMode": {
            "useNormalizedOHMUrl": true
          }
        }
      }
    ]
  }
}
```

5. 混淆配置：

SDK 已做混淆处理，集成时请带上混淆规则，勿再次混淆 SDK

# OneLogin（一键登录）

## 1.1、调用逻辑

预取号逻辑封装在 SDK 内部，开发者只需控制授权页拉起时机，无需关心预取号是否完成。SDK 内部处理预取号的逻辑，包括预取号超期后的重新预取号，以及弱网状态下的重试等。

1. `init()` 初始化 SDK 并配置 APPID
2. `register()` 注册（应用启动或进入登录页的前一个页面、用户登出时是调用该方法的时机）
3. `requestToken()` 拉起授权页面（调用该方法前可以调用 `isPreGetTokenResultValidate` 判断预取号是否成功, 失败时可先加载自定义 loading 等待对话框）
4. `dismissAuthPage()` 关闭授权页面

`register` 方法内会自动进行预取号，如果应用启动就调用，建议先判断或者请求网络权限之后再调用，否则可能因为网络不可达导致预取号失败。另外建议 `register` 与 `requestToken` 方法不要同时调用，预留预取号一定的时间以及内部初始化。

## 1.2、初始化

- 初始化 `init`

SDK 初始化接口 `init`

方法描述

```
public init(appId: string): OneLoginHelper
```

从 `v1.1.3` 版本开始，移除上下文入参，变更为 `requestToken()` 入参

参数说明

参数	类型	说明
appId	string	后台配置唯一产品 APPID，请在官网申请, 注意与服务端保持一致

- 注册 `register`

方法描述

注册接口，在登录页前一个页面初始化时调用；登录成功后 SDK 内部不再维护预取号的有效性，如果用户退出登录后，为了方便下次重新登录能快速拉起授权页，也可以重新调用 `register` 进行预取号。

```
public register(appId?: string, timeout?: number): void
```

参数说明

参数	类型	说明
app_id	string	后台配置唯一产品 APPID，请在官网申请, 如 init 接口传了 appld 此处可传空值
timeout	number	超时时间，单位: ms，取值范围: 1000~15000，默认 8000。传递该参数会统一设置预取号超时时间和取号超时时间为设定值。

代码示例

```
OneLoginHelper
    .with()
    //开启 SDK 日志打印功能
    .setLogEnable(true)
    .init(APPID)
    .register();
```

1.3、拉起授权页

- 拉起授权页 requestToken

方法描述

在需要登录的地方调用 requestToken 接口拉起一键登录授权页，待用户点一键登录授权后获取运营商 token，获取成功后即可请求服务端换取本机手机号码。

```
requestToken(context: UIContext, oneLoginThemeConfig: OneLoginThemeConfig,
oneLoginListener: AbstractOneLoginListener): void
```

从 v1.1.3 版本开始，UIContext入参移至requestToken()，以便init()调用可前置

参数说明

参数	类型	说明
context	UIContext	上下文
oneLoginThemeConfig	OneLoginThemeConfig	自定义全局配置接口，用来配置授权页面 UI 样式
oneLoginListener	AbstractOneLoginListener	回调监听器，需要开发者自己实现

注：弹窗模式实现的授权页暂不支持

代码示例

取号并获取免密登录的 token，通过接口进行校验，并获取登录信息。

```
OneLoginHelper
    .with()
    .requestToken(getUIContext(), new
OneLoginThemeConfig.Builder().build(), this.oneLoginListener);
```

```
private oneLoginListener: AbstractOneLoginListener = {
  onResult:(jsonObject: Record<string, Object>)=>{
    try {
      let status = jsonObject["status"];
      // status=200 为取号成功, 其他返回码请参考返回码章节
      if (status == 200) {
        const processId = jsonObject["process_id"] as string;
        const token = jsonObject["token"] as string;
        const authCode = jsonObject["authcode"] as string;
        let param: Record<string, Object> = {
          "process_id": processId,
          "token": token,
          "authcode": authCode,
          "id_2_sign": Constants.APP_ID
        }
        let result = HttpUtils.requestNetwork(<---获取用户登录信息的接口-->, param);

        /// TODO 判断 result 是否有效
        OneLoginHelper.with().stopLoading();
        OneLoginHelper.with().dismissAuthPage(this.getUIContext());
      }
    } catch (e) {
      e.message;
    }
  }
}
```

## 1.4、关闭授权页

- 关闭授权页 `dismissAuthPage`

### 方法描述

主动关闭授权页, SDK 除了返回按钮触发关闭以外, 默认是不 back 授权页的, 需要开发者在回调结束后自行实现关闭授权页。

```
public dismissAuthPage(uiContext: UIContext): void
```

从 **v1.1.3** 版本开始, 增加 `UIContext` 入参

### 代码示例

```
OneLoginHelper.with().dismissAuthPage(this.getUIContext());
```

## 授权页面 UI 修改

通过实例化 `OneLoginThemeConfig` 并进行自定义配置, 完成授权页面 UI 的个性化设计, 每次调用拉起授权页方法前必须先传入该实例, 否则授权界面会展示异常。授权页 UI 设计规范如下:



## 1、创建实例

- 创建实例

```
let oneLoginThemeConfig = new OneLoginThemeConfig();
```

## 1、添加自定义控件

- 添加自定义控件 `addOneLoginRegisterViewConfig`

### 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 addOneLoginRegisterViewConfig(authRegisterViewConfig: WrappedBuilder<[]>[]): OneLoginHelper
```

### 代码示例

```
OneLoginHelper.with().addOneLoginRegisterViewConfig([wrapBuilder(CustomView)])  
;  
  
@Builder  
function CustomView() {  
    Text('自定义组件')  
        .size({ width: 'auto', height: 'auto' })  
        .fontSize(12)  
        .fontColor(Color.Pink)  
}
```

```
        .backgroundColor(0x550055)
        .alignRules({
            top: { anchor: '__container__', align: VerticalAlign.Top },
            left: { anchor: '__container__', align: HorizontalAlign.Start },
            right: { anchor: '__container__', align: HorizontalAlign.End }
        })
    }
```

参数说明

参数	类型	说明
id	string	开发者自定义控件名称
authRegisterViewConfig	WrappedBuilder<[]>[]	配置开发者自定义控件的 <code>WrappedBuilder</code>

## 4、授权页面其他配置

- 设置授权页面背景 `setAuthBGImgPath`

方法描述

设置背景图片。

```
setAuthBGImgPath(authBGImgPath: ResourceStr)
```

参数说明

参数	参数类型	说明	默认值
authBGImgPath	string	设置背景图片。放在 <code>media</code> 目录下，以下背景图片路径与之保持一致。不需要加后缀，比如图片在 <code>media</code> 中的存放目录是 <code>resources/base/media/demo_bg.png</code> ，则传入参数为 <code>\$r('app.media.demo_bg')</code>	gt_one_login_bg

- 标题栏设置 `setAuthNavConfig`

方法描述

设置标题栏布局

```
public setAuthNavConfig(authNavParams: AuthNavParams)
```

AuthNavParams 参数说明



参数	参数类型	说明	默认值
navColor	ResourceColor	标题栏颜色	默认0xFFFFFFFF
navMarginTop	number	授权页标题栏上边距	0
authNavHeight	Length	标题栏高度	49
authNavGone	boolean	标题栏是否隐藏	false
navText	string	文字设置，默认为空	""
navTextColor	ResourceColor	字体颜色	默认0xFF000000
navTextSize	number	字体大小	17
navTextMargin	number	标题栏文本的左右间距，可选配	36
navTextFontWeight	FontWeight	标题栏的文字的字体	FontWeight.Normal
webNavTextFontWeight	FontWeight	条款页面标题栏的文字的字体	FontWeight.Normal
returnImgPath	string	返回按钮图片	gt_one_login_ic_chevron_left_black
returnImgWidth	Length	返回按钮图片宽度	24
returnImgHeight	Length	返回按钮图片高度	24
returnImgHidden	boolean	返回按钮是否隐藏	false
returnImgOffsetX	Length	返回按钮图片距离屏幕左边X轴偏移量	12
returnImgOffsetY	Length	返回按钮图片距离上方偏移量，可选配，不传该参数默认标题栏垂直居中	0
returnImgCenterInVertical	boolean	返回按钮图片是否在标题栏中垂直居中，若值为true将忽略returnImgOffsetY	true

- 服务条款页面标题栏设置 `setWebNavLayout`

### 方法描述

设置服务条款页面标题栏布局

```
public setWebNavConfig(webNavParams: WebNavParams)
```

### webNavParams参数说明

参数	参数类型	说明	默认值
webNavColor	ResourceColor	标题栏颜色	默认0xFFFFFFFF
webNavHeight	Length	标题栏高度	49
webNavTextNormal	boolean	设置是否标题栏中间文字使用默认值，true 为使用webNavViewText，false为使用默认隐私条款的名字	false
webNavText	string	标题栏中间文字	服务条款
webNavTextColor	ResourceColor	标题栏中间文字颜色	默认0xFF000000
webNavTextSize	number	标题栏中间文字大小	17
webNavMarginTop	number	标题栏上边距	0
webNavreturnImgPath	string	返回按钮图片	gt_one_login_ic_chevron_left_black
webNavreturnImgWidth	Length	返回按钮图片宽度	24
webNavreturnImgHeight	Length	返回按钮图片高度	24
webNavreturnImgHidden	boolean	返回按钮是否隐藏	false
webNavreturnImgOffsetX	Length	返回按钮图片距离屏幕左边X轴偏移量	12
webNavreturnImgOffsetY	Length	返回按钮图片距离上方偏移量，可选配，不传该参数默认标题栏垂直居中	0
webNavreturnImgCenterInVertical	boolean	返回按钮图片是否在标题栏中垂直居中，若值为true将忽略 returnImgOffsetY	true
webDomStorage	boolean	Web页面是否开启DOM STORAGE	false

- 返回事件屏蔽设置 `setBlockReturnEvent`

方法描述

设置是否屏蔽返回键与返回按钮返回事件，屏蔽后仍然会发生对应的错误码回调，只是 SDK 默认不关闭授权页。

```
setBlockReturnEvent(blockReturnKey: boolean, blockReturnBtn: boolean)
```

参数说明

参数	参数类型	说明	默认值
blockReturnKey	boolean	是否屏蔽返回键返回事件	false
blockReturnBtn	boolean	是否屏蔽返回按钮返回事件	false

- logo设置 `setLogoImgView`

方法描述

设置logo相关

```
setLogoImgView(logoParams: LogoParams)
```

LogoParams 参数说明

参数	参数类型	说明	默认值
logoImgPath	string	logo 图片	gt_one_login_logo
logoWidth	Length	logo 图片宽度	71
logoHeight	Length	logo 图片高度	71
logoHidden	boolean	logo 是否隐藏	false
logoOffsetY	Length	logo 相对于上方 y 偏移	125
logoOffsetX	Length	logo 相对于屏幕左边 x 轴偏移量，当为 0 时表示居中显示	0
logoAlignRules	AlignRuleOption	logo 组件对齐规则	-

- 号码显示设置 `setNumberView`

### 方法描述

设置号码相关

```
setNumberView(numberParams: NumberParams)
```

### NumberParams 参数说明

参数	参数类型	说明	默认值
numberColor	ResourceColor	号码栏字体颜色	默认0xFF3D424C
numberSize	number	号码栏字体大小	24
numberOffsetY	Length	号码栏相对于上方 y 偏移	200
numberOffsetX	Length	号码栏相对于屏幕左边 x 轴偏移量，当为 0 时表示居中显示	0
numberText	CharSequence	号码栏的富文本内容, 参考 TextView 的 <code>setText(CharSequence text)</code> 方法	null
numberFontWeight	FontWeight	号码栏的文字的字体	FontWeight.Normal
numberAlignRules	AlignRuleOption	号码栏组件对齐规则	-

- 切换账号视图设置 `setSwitchView`

### 方法描述

设置切换账号相关

```
setSwitchView(switchViewParams: SwitchViewParams)
```

### SwitchViewParams 参数说明

参数	参数类型	说明	默认值
switchText	string	切换账号文字	切换账号，默认情况下受背景大小限制，只能显示4个字，如遇显示不全，建议通过 <code>setSwitchViewLayout</code> 调整背景大小试试
switchColor	string	切换账号字体颜色	默认0xFF3973FF
switchSize	number	切换账号字体大小	14
switchHidden	boolean	切换账号是否隐藏	false
switchOffsetY	Length	切换账号相对于上方 y 偏移	249
switchOffsetX	Length	切换账号相对于屏幕左边 x 轴偏移量, 当为0时表示居中显示	0
switchImgPath	string	切换账号背景图片	默认无背景
switchWidth	Length	切换账号背景宽度	80
switchHeight	Length	切换账号背景高度	25
switchFontWeight	FontWeight	切换账号的文字的字体	FontWeight.Normal
switchAlignRules	AlignRuleOption	切换账号组件对齐规则	-

- 登录按钮布局设置 `setLogBtnLayout`

方法描述

设置登录按钮布局

```
setLogBtnLayout(logBtnParams: LogBtnParams)
```

LogBtnParams 参数说明

参数	参数类型	说明	默认值
logBtnImgPath	string	登录按钮背景图片	gt_one_login_btn
logBtnUncheckedImgPath	ResourceStr	未选中隐私栏勾选框时登录按钮的背景图片	gt_one_login_btn_unchecked
logBtnUncheckedImgColor	ResourceColor	未选中隐私栏勾选框时登录按钮的背景颜色	gt_one_login_btn_unchecked_normal_color
logBtnWidth	Length	登录按钮宽度	268
logBtnHeight	Length	登录按钮高度	36
logBtnRadius	Length	BorderR0adiuses	登录按钮圆角
logBtnStateEffect	boolean		登录按钮是否开启按压效果
logBtnOffsetY	Length	登录按钮相对于上方 y 偏移	324
logBtnOffsetX	Length	登录按钮相对于屏幕左边 x 轴偏移量, 当为0时表示居中显示	0
logBtnText	string	文字设置	一键登录
logBtnColor	ResourceColor	文字颜色	0xFFFFFFFF
logBtnTextSize	number	文字大小	15
logBtnTextFontWeight	FontWeight	登录按钮中间的文字的字体	FontWeight.Normal
disableBtnIfUnChecked	boolean	登录按钮是否在选择框未选择时自动禁用	false
logAlignRules	AlignRuleOption	登录按钮组件对齐规则	-

- 登录按钮可多次点击设置 `setAuthBtnMultipleClick`

方法描述

设置取号成功后点击登录按钮能否收到点击回调(指 `onLoginButtonClick(): void`)

```
setAuthBtnMultipleClick(enable: boolean)
```

参数说明

参数	参数类型	说明	默认值
enable	boolean	在取号成功后点击登录按钮能否收到点击回调	false

- 设置 `Slogan` 布局

方法描述

设置 `Slogan` 布局

```
setSloganLayout(sloganParams: SloganParams)
```

参数说明

参数	参数类型	说明	默认值
visible	boolean	slogan 是否显示	true
sloganText	string	slogan 文本	运营商标语，如 中国移动提供认证服务
sloganColor	ResourceColor	设置 Slogan 字体颜色	默认0xFFA8A8A8
sloganSize	number	设置 Slogan 字体大小	10
sloganOffsetY	Length	设置 Slogan 相对于上方 y 偏移	40
sloganOffsetX	Length	设置 Slogan 相对于屏幕左边 x 轴偏移量，当为0时表示居中显示	0
sloganFontWeight	FontWeight	Slogan 文字的字体	FontWeight.Normal
sloganWidth	Length	Slogan 的宽度，仅支持设置宽度自适应或者是固定值	'auto'
sloganHeight	Length	Slogan 的高度，仅支持设置高度自适应或者是固定值	'auto'
sloganAlignRules	AlignRuleOption	slogan 组件对齐规则	-

- 设置 loading 自定义组件

方法描述

设置 loading 自定义组件

```
setLogBtnLoadingComponent(component: WrappedBuilder<[]>, enableLoading?: boolean)
```

参数说明

参数	参数类型	说明	默认值
component	WrappedBuilder<[]>	@Builder修饰的function自定义组件	@Builder function LoadingProgress()
enableLoading	boolean	是否显示loading	true

- 隐私栏布局设置 setPrivacyLayout

方法描述

设置隐私条款布局

```
setPrivacyLayout(privacyParams: PrivacyParams)
```

参数说明

参数	参数类型	说明	默认值
privacyOffsetY	Length	设置隐私条款相对于上方 y 偏移	0
privacyOffsetX	Length	设置隐私条款对于屏幕左边 x 轴偏移量, 当为 0 时表示居中显示	0
privacyCheckBoxWidth	Length	选择框图片宽度	9
privacyCheckBoxHeight	Length	选择框图片高度	9
privacyCheckBoxOffsetY	Length	选择框图片Y轴偏移, 可选配	0
privacyCheckBoxMarginRight	number	选择框图片右边距, 可选配	5
privacyCheckedImgPath	ResourceStr	复选框选中图片	<code>\$r('app.media.gt_one_login_checked')</code>
privacyUncheckedImgPath	ResourceStr	复选框未选中图片	<code>\$r('app.media.gt_one_login_unchecked')</code>
baseClauseColor	ResourceColor	设置隐私条款基础文字颜色	默认0xFFA8A8A8
clauseColor	ResourceColor	设置隐私条款协议文字颜色	0xFF3973FF
privacyClauseTextSize	number	设置隐私条款字体大小	10
privacyClauseBaseFontWeight	FontWeight	隐私栏基础的文字的字体	FontWeight.Normal
privacyClauseFontWeight	FontWeight	隐私条款的文字的字体	FontWeight.Normal
privacyClauseTextStrings	OLPrivacyTermItem[]	设置隐私条款数组	null
privacyAddFrenchQuotes	boolean	是否设置隐私条款名称显示书名号	false
privacyCheckBoxAlignRules	AlignRuleOption	隐私条款选择框组件对齐规则	-
privacyContentAlignRules	AlignRuleOption	隐私条款内容组件对齐规则	-
privacyTextTopMargin	number	设置隐私条款文字外部上边距	1
enableToast	boolean	设置是否弹出 Toast 提示文字, 可选配	true
privacyUncheckedToastText	string	设置未同意隐私条款的文字提示相关	请同意服务条款

- 开发者隐私条款设置 `privacyClauseTextStrings`

**参数描述** `OLPrivacyTermItem`

设置多个开发者隐私条款相关。自定义多个隐私条款。可通过预留一组空字符串配置让 SDK 自动添加运营商协议。

参数	参数类型	说明	默认值
privacyPrefix	string	设置隐私条款前连接字符	
privacySuffix	string	设置隐私条款后连接字符	
privacyContent	string	设置隐私条款内容	
privacyLinkURL	string	设置隐私条款链接	
operatorPlaceHolder	boolean	是否为运营商占位	

**代码示例**

```
/**
 * 一个 OLPrivacyTermItem 为一组隐私协议条款
```

```

* 任意一组 operatorPlaceholder 传 true, SDK 会自动添加运营商隐私条款,
operatorPlaceholder 传 true的位置可前后调整。
* 不添加operatorPlaceholder为true的条目时, SDK自动添加运营商条款到末尾。
*/
let privacyTerms: OLPrivacyTermItem[] = []
privacyTerms.push(new OLPrivacyTermItem({
    privacyPrefix: '登录且同意',
    privacySuffix: '',
    privacyContent: '自定义条款',
    privacyLinkURL: 'www.geetest.com',
}))
privacyTerms.push(new OLPrivacyTermItem({
    privacyPrefix: '和',
    privacySuffix: '并使用本机号码登录',
    operatorPlaceholder: true
}))
OneLoginHelper.with().requestToken(getUIContext(),
    new OneLoginThemeConfig()
        .setPrivacyLayout({privacyClauseTextStrings:privacyTerms})

```

#### 注:

运营商隐私协议必须显示, 开发者按需保留一组条款参数的operatorPlaceholder为true, 不添加operatorPlaceholder为true的条目时, SDK自动添加运营商条款到末尾。

- 设置授权按钮监听事件 `setOneLoginAuthListener`

#### 方法描述

`OneLoginAuthListener` 接口方法 `onOLAuthListener: (context: UIContext, oneLoginAuthCallback: OneLoginAuthCallback) => void;`, 通过 `OneLoginAuthCallback` 的回调 `onOLAuthCallback(keepOn: boolean)` 决定是否继续登录流程。可以实现二次弹窗确认的功能

```
setOneLoginAuthListener(oneLoginAuthListener: OneLoginAuthListener)
```

#### 参数说明

参数	参数类型	说明	默认值
oneLoginAuthListener	OneLoginAuthListener	授权按钮监听	null

#### 代码示例

```

let oneLoginThemeConfig = new OneLoginThemeConfig()
oneLoginThemeConfig.setOneLoginAuthListener({
    onOLAuthListener: (_context: UIContext, oneLoginAuthCallback:
OneLoginAuthCallback): void => {
        promptAction.showDialog({
            message: '是否登录授权',
            buttons: [
                {
                    text: '确定',

```



```
        color: '#000000'
    },
    {
        text: '取消',
        color: '#000000'
    }
],
}))
.then(data => {
    if (data.index == 0) {
        oneLoginAuthCallback.onOLAuthCallback(true)
    } else {
        oneLoginAuthCallback.onOLAuthCallback(false)
    }
})
})
})
```

# 其他接口说明

## 1、回调接口实现

通过实现 `AbstractOneLoginListener` 接口，通过它的多个回调可获取预取号、拉起授权页取号时的各种返回状态与结果，每次调用拉起授权页方法前必须先传入该接口的实现，否则不能正常拉起授权页。

### 1.1、结果返回

#### 方法描述

当前流程的结果返回。取号成功、取号失败、切换账号、点返回或者按返回键从授权页返回取消登录等都从该回调返回结果，具体请参考 `OneLogin` 返回码。

```
onResult: (jsonObject: Record<string, Object>) => void;
```

#### 参数说明

参数	类型	说明
jsonObject	Record<string, Object>	回调信息

#### 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(有效期10分钟)
app_id	是	string	后台配置唯一 id
operator	是	string	客户端获取的运营商
clienttype	是	string	客户端，1 表示 HarmonyOS
sdk	是	string	SDK 的版本号
status	是	number	状态码，状态码为 200
token	是	string	运营商返回的 accessToken
authcode	否	string	电信运营商返回的 authcode

- 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(有效期10分钟)
app_id	是	string	后台配置唯一 id
metadata	是	Record<string, Object>	具体的错误原因
operator	是	string	客户端获取的运营商
clienttype	是	string	客户端，2 表示 HarmonyOS
sdk	是	string	SDK 的版本号
status	是	number	状态码，状态码为 500

## 2.2、获取脱敏手机号

### 方法描述

获取用于界面展示的脱敏手机号

```
onRequestTokenSecurityPhone?: (phone: string) => void
```

### 参数说明

参数	类型	说明
phone	string	脱敏手机号

## 2.3、授权页面拉起

### 方法描述

授权页面拉起回调

```
onAuthPageCreate?: (context: UIContext) => void
```

### 参数说明

参数	类型	说明
context	UIContext	授权页面上下文

## 2.4、登录按钮点击

### 方法描述

登录按钮点击回调

```
onLoginButtonClick?: () => void
```

## 2.5、返回按钮点击

### 方法描述

点授权页面标题栏返回按钮或者手机返回键时的回调。

注: 该回调与带 -20301/-20302 返回码的 `onResult` 回调共存，两者都会返回

```
onBackButtonClick?: () => void
```

## 2.6、切换账号点击

### 方法描述

点授权页面切换账号栏的回调。

注: 该回调与带 -20303 返回码的 `onResult` 回调共存，两者都会返回

```
onSwitchButtonClick?: () => void
```

## 2.7、隐私条款 CheckBox 点击

### 方法描述

CheckBox点击回调

```
onPrivacyCheckBoxClick?: (isChecked: boolean) => void;
```

#### 参数说明

参数	类型	说明
isChecked	boolean	CheckBox 是否选择

## 3、日志打印

- 日志打印控制 `setLogEnable`

#### 方法描述

设置是否开启 SDK 日志打印功能，默认是开启的，开启后 SDK 运行过程中会打印以 Geetest\_OneLogin 为 TAG 的 logcat 日志。正式上线应用可以调用该接口关闭日志。

```
public setLogEnable(openDebug: boolean): OneLoginHelper
```

#### 参数说明

参数	类型	说明
logEnable	boolean	是否开启日志

#### 代码示例

```
OneLoginHelper.with().setLogEnable(false);
```

注：开启日志后，Release 模式也会打印 SDK 内日志

## 4、预取号状态判断

- 取号是否有效 `isPreGetTokenResultValidate`

#### 方法描述

判断是否预取号结果是否有效，预取号失败、预取号结果超期、预取号结果已使用情况均返回预取号结果失效。

```
public isPreGetTokenResultValidate: boolean()
```

#### 代码示例

```
OneLoginHelper.with().isPreGetTokenResultValidate();
```

## 5、SDK 版本号

- 获取 SDK 版本号 `sdkVersion`

### 方法描述

获取 SDK 版本号

```
public sdkVersion(): string
```

### 代码示例

```
OneLoginHelper.with().sdkVersion()
```

## 6、初始化状态判断

- 初始化是否成功 `isInitSuccess`

### 方法描述

判断是否初始化成功，cancel 之后必须重新初始化。

```
public isInitSuccess: boolean()
```

### 代码示例

```
OneLoginHelper.with().isInitSuccess();
```

## 7、隐私条款是否勾选

- 隐私条款是否勾选 `isPrivacyChecked`

### 方法描述

判断是否勾选同意隐私条款声明。

```
public isPrivacyChecked: boolean()
```

### 代码示例

```
OneLoginHelper.with().isPrivacyChecked();
```

## 8、SDK 内存清理

- 释放 SDK 所有内部引用 `cancel`

### 方法描述

一键登录结束后，如果不需要继续使用一键登录相关功能，可以调用 cancel 方法释放 SDK 内部的引用，减少内存泄漏。**调用该方法后，如需再次登录，需要重新预取号。**

```
public cancel(): void
```

注: 该方法会清理掉 SDK 内部核心单例并置空, 导致部分缓存失效, 部分接口耗时增加, 建议酌情使用。

#### 代码示例

```
OneLoginHelper.with().cancel();
```

- 释放 SDK 持有的 listener 引用 `removeOneLoginListener`

#### 方法描述

一键登录结束后, 可以调用 `removeOneLoginListener` 释放 `preGetToken` 与 `requestToken` 传入的 listener 参数, 防止 **listener** 持有外部类的引用导致内存泄漏。

```
public removeOneLoginListener(): void
```

#### 代码示例

```
OneLoginHelper.with().removeOneLoginListener();
```

## 9、删除预取号的缓存

#### 方法描述

在预取号处于成功态时改为失败态, 并删除内存中预取号获得的相关数据

```
public deletePreResultCache(): void
```

#### 代码示例

```
OneLoginHelper.with().deletePreResultCache();
```

## OnePass（本机号码认证）

### 1、调用逻辑

- `init()` 初始化 SDK
- `getToken()` 获取本机号码对应的Token

### 2、接口配置

#### 2.1、初始化

##### 方法描述

SDK 初始化接口, 在项目的具体页面的生命周期方法里进行初始化。

```
public init(appId: string, timeout: number): OnePassHelper
```

#### 参数说明

参数	类型	说明
appId	string	后台配置唯一产品 APPID，请在官网申请，OneLogin 与 OnePass 属于不同的产品，注意产品 APPID 不可混用
timeout	number	超时时间，单位: ms，取值范围: 1000~15000，默认 8000

#### 代码示例

```
OnePassHelper.with().init(appId, 8000);
```

## 2.2、本机号校验接口

#### 方法描述

在点击登录时调用此接口获取认证传入的手机号码是否为本机号码、是否与本机 SIM 卡一致（对于双卡手机，则认证是否为当前流量对应的本机号码）需要的 token，具体认证需调用服务端接口，传入该接口获取到的 token 进行验证。

```
public getToken(phone: string, onePassListener: OnePassListener): void
```

#### 参数说明

参数	类型	说明
phone	string	需进行认证的手机号码
onePassListener	OnePassListener	回调监听器，需要开发者自己实现

#### 示例代码

```
OnePassHelper.with().getToken(this.opContent, {
    onTokenFail: (jsonObject: Record<string, Object>) => {
        Log.i(Constants.TAG, "onTokenFail:" + jsonObject.toString());
        AlertDialog.show({ message: "本机认证失败:" + JSON.stringify(jsonObject)
    });
    },

    onTokenSuccess: async (jsonObject: Record<string, Object>) => {
        console.log(Constants.TAG, "onTokenSuccess:" +
JSON.stringify(jsonObject));
        jsonObject.id_2_sign = Constants.APP_ID_OP;
        jsonObject.phone = this.opContent;
        let result = await this.CheckGatewayTask(jsonObject); // 请求服务端接口校验
手机号码
        console.log(Constants.TAG, "CheckGateway 请求结束");
    }
});
```

```

        console.log(Constants.TAG, "CheckGateway 请求成功:" + result);
    }
    try {
        let jsonObject = JSON.parse(result) as Record<string, Object>;
        let status = jsonObject.status as number;
        if (status == Constants.SUCCESS_CODE && "0" == (jsonObject.result)) {
            AlertDialog.show({ message: "校验成功:" + JSON.stringify(result)
        });
        } else {
            console.log(Constants.TAG, "CheckGateway 校验失败:" +
JSON.stringify(result));
        }
    } catch (e) {
        console.log(Constants.TAG, "CheckGateway 校验失败:" +
JSON.stringify(result));
    }
}
}
}

```

## 2.3、SDK 内存清理

### 方法描述

在页面关闭的时候执行此方法。执行此方法后，会释放 SDK 所有内部引用。**调用该方法后，如需再次校验，需要重新初始化。**

```
public cancel(): void;
```

### 代码示例

```
OnePassHelper.with().cancel();
```

## 3、OnePassListener 回调接口实现

### 3.1 校验成功

### 方法描述

```
onTokenSuccess: (jsonObject: Record<string, Object>) => void
```

### 参数说明

参数	类型	说明
jsonObject	Record<string, Object>	回调信息

jsonObject 的参数说明



参数名	必须	类型	说明
process_id	是	string	流水号(有效期10分钟)
accesscode	是	string	运营商返回的 accesscode
phone	是	string	待认证的手机号码

返回数据示例：

```
{
  "process_id" : "9dca9ca39a4ec4b404f2f042b3c25d1b", // 流水号
  "accesscode" :
  "CM__1__3996159873d7ccc36f46803b88dda97a__2.5.3.1__STsid0000001631534638077Nnr
  wIDzIvuCswl0wviUEVjnFGU4skzvX", // accesscode
  "phone" : "15012345678" // 待认证的手机号码
}
```

### 3.2 校验失败

方法描述

```
onTokenFail: (jsonObject: Record<string, Object>) => void;
```

参数说明

参数	类型	说明
jsonObject	Record<string, Object>	回调信息

jsonObject 的参数说明

参数名	必须	类型	说明
process_id	是	string	流水号(有效期10分钟)
code	是	string	错误码
custom_id	是	string	appId
metadata	是	string	错误描述信息的 json 字符串
real_op	是	string	sdk检测到的运营商类型
op	是	string	本次校验使用的运营商类型
clienttype	是	string	客户端类型，1表示Android
sdk	是	string	sdk版本号

返回数据示例：

```
{
  "process_id" : "d705fb57473e4d7b0e9911243a06fae4", //流水号
  "code" : "-20203", //错误码
  "custom_id" : "3996159873d7cdc36f25803b88dea95a", //appid
  "metadata" : {
    "error_data" : "Currently getting operators error:unknown" //错误描述
  },
  "real_op": "unknown", //sdk检测到的运营商类型
  "op": "unknown", //本次校验使用的运营商类型
  "clienttype": "1", //客户端类型, 1表示Android
  "sdk": "2.5.3.1" //sdk版本号
}
```