Android 接入指南

1 集成指引

1.1 目录解构

1.1.1 资源包目录结构

----HXOneKeyLogin-android.zip
----|----HXLibrary.aar
----|----HXOneKeyLoginaar
----|----Resource.zip
----|----|----layout（授权页面布局文件需导入）
----|----|----l----activity_oauth_dialog_land.xml（弹框模式横屏）
----|----|----l----activity_oauth_dialog_port.xml（弹框模式竖屏）
----|----|----l----activity_oauth_land.xml(横屏沉浸式)
----|----|----l----activity_oauth_port.xml(竖屏沉浸式)
----|----|----l----oauth_privacy_web_activity.xml(协议布局文件)
----|----|----libs
----|----|----l----arm64-v8a
----|----|----l----|----libCtaApiLib.so
----|----|----l----armeabi
----|----|----l----|----ibCtaApiLib.so
----|----|----l----armeabi-v7a
----|----|----l----|----libCtaApiLib.so
----|----|----l---- x86
----|----|----l----|----libCtaApiLib.so
----|----|----l----x86_64
----|----|----l----|---- libCtaApiLib.so
----|----|----xml
----|----|----l----|---- network_security_config.xml（9.0 网络适配）
----|----|----demo.apk
----|----Demo.zip(demo 示例)

1.2 集成步骤

1.2.1 架包导入

1.在资源下载中下载一键通 HXOneKeyLogin-android.zip 压缩包，解压之后将其中的 HXLibrary.aar,HXOneKeyLogin.arr 拷贝到自己工程的 libs 目录下，如没有该目录需新建。

2.导入布局文件，请将解压后的文件夹->layout（可根据项目需求选择，或者全部拷贝）->对应的布局文件拷贝到项目 layout 中。

3.解压之后中 libs 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 libs 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。

4.aar 在项目中引用时 gradle 以及混淆配置请参考文档后面的混淆配置及 build.gradle 配置。

5.复制<color name="white">#ffffff</color> <color name="gray">#999999</color>到项目的 color.xml 中

6.具体资源文件存放路径可参考 DemoProject

1.2.2 权限配置

在 android manifest 中添加如下权限

```
<uses-permission android:name="android.permission.WRITE_SETTINGS"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<uses-permission android:name="android.permission.GET_TASKS" />
```

权限说明	权限列表
网络权限、必要	android.permission.INTERNET
网络权限、必要	android.permission.WRITE_SETTINGS
网络权限、必要	android.permission.ACCESS_NETWORK_STATE
手机状态权限（Android6.0及以上需app动态申请此权限）、必要	android.permission.READ_PHONE_STATE
获取信息有关当前或最近运行的任务、必要	android.permission.GET_TASKS
WiFi状态权限(需要运行时权限)、可选	android.permission.ACCESS_WIFI_STATE
WiFi状态权限(需要运行时权限)、可选	android.permission.CHANGE_WIFI_STATE

在 application 中添加 android:networkSecurityConfig = “@xml/network_security_config” 适配 9.0 网络权限。

在 xml 中创建 network_security_config.xml。如下：

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <base-config cleartextTrafficPermitted="true">
        <trust-anchors>
            <certificates src="system" overridePins="true" />
            <certificates src="user" overridePins="true" />
        </trust-anchors>
    </base-config>
```

```
9.      </network-security-config>
```

1.2.3 build.gradle 配置 SDK 包

```
1.      dependencies{
2.      implementation fileTree(include: ['*.aar'], dir: 'libs')
3.      }
```

1.2.4 build.gradle 配置 jni(so 库)

开发者按自己的需求，将需要支持的 cpu 架构 so 文件添加到工程。如果将 so 文件添加在 module 的 libs 文件夹下(建议全部 so 库都放进去)，然后配置 app 的 build.gradle 文件：

```
1.      android {
2.      sourceSets {
3.      main {
4.      jniLibs.srcDirs = ['libs','src/main/jniLibs']
5.      }
6.      }
7.      repositories {
8.      flatDir {
9.      dirs 'libs'
10.     }
11.     }
12.     }
```

1.2.5 混淆配置

- 在 proguard-rules 文件中添加如下代码：

```
1.      -dontwarn com.sdk.**
2.      -keep class com.sdk.** {*;}
3.      -keep class cn.com.chinatelecom.account.api.** {*;}
4.      -dontwarn cn.com.chinatelecom.gateway.lib.**
5.      -keep class cn.com.chinatelecom.gateway.lib.** {*;}
6.      -keepclassmembers class cn.com.chinatelecom.gateway.lib.** {*;}
7.      -keep class com.xinyan.HXLogin.** {*;}
8.      -keep class com.xinyan.android.device.sdk.** {*;}
9.      -keep class com.huaxin.library.** {*;}
```

1.2.6 配置授权页面 Activity 主题样式(如需修改样式，请修改 style.xml 中内容，无需修改则使用 SDK 默认主题样式)

授权页面主题样式（主要是沉浸模式和弹框模式）需要在 values values-v19 values-21 values-23 设置 activity 样式

```
1.      //values 文件夹 styles.xml:
2.        <style name="oauthActivityParent" parent="android:Theme.Translucent.NoTitleBar">
3.            <item name="android:windowBackground">@android:color/transparent</item>
```

```
4.              <item name="android:fitsSystemWindows">true</item>
5.              <item name="android:backgroundDimEnabled">true</item>
6.              <item name="android:windowNoTitle">true</item>
7.              <item name="android:configChanges">orientation|keyboardHidden|screenSize</ite
    m>
8.              <item name="android:launchMode">singleTop</item>
9.          </style>
10.
11.         <style name="oauthDialogActivity">//弹框模式
12.             <!--设置 dialog 的背景-->
13.             <item name="android:windowBackground">@android:color/transparent</item>
14.             <!--设置 Dialog 的 windowFrame 框为无-->
15.             <item name="android:windowFrame">@null</item>
16.             <!--设置无标题-->
17.             <item name="android:windowNoTitle">true</item>
18.             <!--是否浮现在 activity 之上-->
19.             <item name="android:windowIsFloating">true</item>
20.             <!--是否半透明-->
21.             <item name="android:windowIsTranslucent">true</item>
22.             <!--设置窗口内容不覆盖-->
23.             <item name="android:windowContentOverlay">@null</item>
24.         </style>
25.
26.     //values-19 文件夹 styles.xml:
27.         <style name="oauthActivityParent" parent="android:Theme.Translucent.NoTitleBar">
28.             <item name="android:windowBackground">@android:color/transparent</item>
29.             <item name="android:fitsSystemWindows">true</item>
30.             <item name="android:backgroundDimEnabled">true</item>
31.             <item name="android:windowTranslucentStatus">false</item>
32.             <item name="android:windowTranslucentNavigation">true</item>
33.             <item name="android:windowNoTitle">true</item>
34.             <item name="android:configChanges">orientation|keyboardHidden|screenSize</ite
    m>
35.             <item name="android:launchMode">singleTop</item>
36.         </style>
37.
38.     //values-v21 文件夹 styles.xml:
39.         <style name="oauthActivityParent" parent="android:Theme.Translucent.NoTitleBar">
40.             <item name="android:windowBackground">@android:color/transparent</item>
41.             <item name="android:fitsSystemWindows">true</item>
42.             <item name="android:backgroundDimEnabled">true</item>
43.             <item name="android:windowTranslucentStatus">false</item>
44.             <item name="android:windowTranslucentNavigation">true</item>
45.             <item name="android:statusBarColor">@android:color/transparent</item>
46.             <item name="android:windowNoTitle">true</item>
47.             <item name="android:configChanges">orientation|keyboardHidden|screenSize</ite
    m>
48.             <item name="android:launchMode">singleTop</item>
49.         </style>
```

```
50.
51.       //values-v23 文件夹 styles.xml:
52.          <style name="oauthActivityParent" parent="android:Theme.Translucent.NoTitleBar">
53.              <item name="android:windowBackground">@android:color/transparent</item>
54.              <item name="android:fitsSystemWindows">true</item>
55.              <item name="android:backgroundDimEnabled">true</item>
56.              <item name="android:windowTranslucentStatus">false</item>
57.              <item name="android:windowTranslucentNavigation">true</item>
58.              <item name="android:statusBarColor">@android:color/transparent</item>
59.              <item name="android:windowNoTitle">true</item>
60.              <item name="android:configChanges">orientation|keyboardHidden|screenSize</ite
   m>
61.              <item name="android:launchMode">singleTop</item>
62.           </style>
```

1.2.7 配置授权页面 Activity

在 APP 的 AndroidManifest.xml 配置运营商授权页面 activity 和协议页面 activity

```
1.        <?xml version="1.0" encoding="utf-8"?>
2.        <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3.            xmlns:tools="http://schemas.android.com/tools"
4.            package="com.huaxin.example"
5.            >
6.            <application
7.                android:networkSecurityConfig="@xml/network_security_config"
8.                >
9.                <!-- 联通授权页面 -->
10.               <activity
11.                   tools:replace="android:screenOrientation"
12.                   android:screenOrientation="behind"
13.                   android:theme="@style/oauthActivityParent"
14.                   android:name="com.sdk.mobile.manager.login.cucc.OauthActivity"
15.                   />
16.               <!-- 电信授权页面 -->
17.               <activity
18.                   tools:replace="android:screenOrientation"
19.                   android:screenOrientation="behind"
20.                   android:theme="@style/oauthActivityParent"
21.                   android:name="cn.com.chinatelecom.account.sdk.ui.AuthActivity"
22.                   android:exported="false" />
23.               <!-- 移动授权页面 -->
24.               <activity
25.                   tools:replace="android:screenOrientation"
26.                   android:screenOrientation="behind"
27.                   android:theme="@style/oauthActivityParent"
28.                   android:name="com.cmic.sso.sdk.activity.OAuthActivity"
29.                   >
30.               </activity>
```

```
        <activity
            tools:replace="android:screenOrientation"
            android:screenOrientation="behind"
            android:theme="@style/oauthActivityParent"
            android:name="com.cmic.sso.sdk.activity.LoginAuthActivity"
            >
        </activity>

        <!-- 协议 WebView 页面 -->
        <activity
            tools:replace="android:screenOrientation"
            android:screenOrientation="behind"
            android:theme="@style/oauthActivityParent"
            android:name="com.xinyan.XYLogin.oklUI.WebViewActicity"
            />
    </application>
</manifest>
```

1.2.8 SDK 代码集成步骤

1、初始化 SDK

建议在 application 的 onCreate()中初始化 SDK

```
    /**
        *@MethodName: initSDK
        *@Description: 方法描述
    *@Param: [context, appKey, appSecret,MERCHANTNO] appKey 应用唯一标示、appSecret 集成 SDK 的鉴权码，用于验证集成 SDK 的合法性,MERCHANTNO,用户 id
        *@return: void
        */
```

```
HXLogin.getInstance().initSDK(this, APPKEY, APPSECRET, MERCHANTNO);;
```

参数说明

参数名	是否必填	参数类型	参数描述
context	必填	Context	上下文信息
appKey	必填	String	应用唯一标示

参数名	是否必填	参数类型	参数描述
appSecret	必填	String	集成华信 SDK 的鉴权码，用于验证集成 SDK
MERCHANTNO	必填	String	用户 id

2、设置调试模式

```
开启 debug 调试模式 默认为 false 不开启
正式发布请将代码屏蔽或设置 false
```

```
HXLogin.getInstance().isDebug(true);
```

3、设置超时时间

```
//设置获取授权时的超时时长 单位 ms 默认 15000ms
```

```
HXUIConfig.getInstance().setOauthOutTime(15000);//不设置使用 sdk 默认超时时长 15000ms
```

4、设置横竖屏

```
HXUIConfig.getInstance().setOrientation(HXUIConfig.PORTRAIT);//竖屏 默认沉浸式
HXUIConfig.getInstance().setOrientation(HXUIConfig.LANDSCAPE);//横屏 默认沉浸式
HXUIConfig.getInstance().setBackgroundImgName("授权页面背景图片名称");//设置沉浸式的背景图片
```

5、设置、关闭弹框模式

```
HXUIConfig.getInstance().setDialogMode(HXUIUtils.dip2px(this, 450),
                        HXUIUtils.dip2px(this, 300),Gravity.CENTER,0.8f);
//设置弹框模式（设置弹框模式前，请先设置屏幕方向），参数：下上文 宽度 、高度、位置、透明度

HXUIConfig.getInstance().disableDialogMode();//关闭弹框模式
```

6、设置自定义 UI

1、修改布局文件名称时，需将布局文件名字传给 SDK
2、不修改布局文件名称时，无需设置布局文件名称，在原有布局文件上进行 UI 拓展即可

Tips:无论是否修改布局文件名称，布局中固有 id 不可修改,具体 id 参考下面表格中组件 ID

```
//设置授权页面使用的布局文件
```

```
2.      HXUIConfig.getInstance().setOauthLayoutIdName("支持用户修改布局文件名字,授权页面布局文件
                                                    名称");//不设置默认使用 SDK 中布局文件
3.      参考代码：
4.      HXUIConfig.getInstance().setOauthLayoutIdName("activity_oauth_port_custom");
```

组件 ID	组件类型
oauth_back_img	ImageView
oauth_title_txt	TextView
oauth_logo_img	ImageView
oauth_mobile_txt	TextView
brand_txt	TextView
service_protocol_box	CheckBox
service_protocol_txt	TextView

7、设置协议

```
1.          CustomProtocol customProtocol = new CustomProtocol();
2.          //设置默认协议
3.          customProtocol.setNoBookMark(false);//默认协议是否无书名号（默认 false 即带书名号） 设
    为 true 时无书名号
4.          customProtocol.setPrefix("同意即登录");//设置协议开始文字
5.          customProtocol.setSeparator("、");//设置默认协议间隔符
6.          customProtocol.setProtocolTextColor(R.color.red);//设置默认协议文字颜色
7.          customProtocol.setProtocolTextSize(20);//设置默认协议字体大小
8.          customProtocol.setBold(false);//设置默认协议粗体  true 为粗体 false 为正常
9.          customProtocol.setUnderline(false);//设置协议下划线 true 为带下划线 false 不带
10.         customProtocol.setSuffix("并授权 appName 使用");//设置协议末尾文本
11.
12.         //设置用户自定义协议
13.         ProtocolEntity protocolEntity = ProtocolEntity.createNewCustom("自定义协议 1","协议
    链接");//创建自定义协议 协议名称 协议链接地址
14.         protocolEntity.setPrefix("和");//自定义协议间隔符
```

```
        protocolEntity.setTextColor(R.color.colorAccent);//自定义协议颜色
        protocolEntity.setTextSize(40);//自定义协议字体大小
        protocolEntity.setBold(false);//设置自定协议粗体 true 为粗体 false 为正常字体
        protocolEntity.setUnderline(false);//设置协议下划线 true 为带下划线 false 为不带下划线
        customProtocol.addProtocolEntity(protocolEntity);//把自定义协议添加到协议控制类中
        HXUIConfig.getInstance().setCustomProtocol(customProtocol);//设置协议
        //设置协议勾选框未勾选时的回调（当用户有特殊需求时）
        HXUIConfig.getInstance().setHXCheckBoxUnCheckedCallBackallBack(new HXCheckBoxUnCheckedCallBackallBack() {
            @Override
            public void callBack(Activity activity) {
                Toast.makeText(activity,"请勾选并同意服务条款",Toast.LENGTH_SHORT).show();
            }
        });
```

8、设置 Loading 加载框

```
    HXUIConfig.getInstance().setLoadingMode(new HXUIConfig.LoadingMode() {
        @Override
        public void show() {
            //显示加载框,用户实现
        }

        @Override
        public void dismiss() {
            //关闭加载框，用户实现
        }

        @Override
        public void setLoadingContext(Context context) {
            //创建 Loading 加载框实例，用户实现
            //Tips：(由于上下文原因，需要用户使用授权页面的上下文创建加载框)
        }
    });
```

9、设置自定义点击事件

```

    //自定义点击事件时，设置是否关闭页面
    boolean close = true;
    HXUIConfig.getInstance().addCustomView(HXViewEntity.createCustomView("自定义 view 的 id 名",close, new HXUICallBack() {
        @Override
        public void clickCallBack(Context context, @Nullable View view) {
            //todo 处理自定义点击事件
        }
    }));
    //设置配置项
```

10、 一键登录

一键登录预取号

```
HXLogin.getInstance().preLogin(this, new HXCallBack() {
    @Override
    public void success(HXResultData resultData) {
        // TODO: 预取号成功，获取预取号返回的令牌 accessCode
        String accessCode = resultData.getAccessCode();
    }

     @Override
    public void failed(HXErrorEntity errorEntity) {
        // TODO: 预取号失败
    }
});
```

拉起一键登录授权页面

一键登录拉起授权页,使用登录成功回调的 oclToken 与服务端接口交互，获得明文手机号。参考服务端 REST API 对接指南，一键登录获取手机号 API 部分。

```
/**
 * 拉起授权页面
 * accessCode 预取号成功返回的令牌 accessCode
 */
 HXLogin.getInstance().openOauthActivity(accessCode,new HXCallBack() {
    @Override
    public void success(HXResultData resultData) {
        // TODO: 拉起授权页点击一键登录成功返回
        //授权令牌 String  resultData.getOclToken()
       //运营商类型 String  resultData.getOperatorType()
    }
    @Override
    public void failed(HXErrorEntity errorEntity) {
        // TODO: 发起授权页点击一键登录失败返回
    }
});
```

11、关闭授权页面

//关闭授权页方法

```
HXLogin.getInstance().finishActivity();
```

12、 号码认证

号码认证预取号

```
HXLogin.getInstance().preNumberValidate(this, new HXCallBack() {
    @Override
    public void success(HXResultData resultData) {
        //号码认证预取号成功，获取预取号返回的令牌 accessCode
        String accessCode = resultData.getAccessCode();
    }

    @Override
    public void failed(HXErrorEntity errorEntity) {
        //号码认证预取号失败回调
    }
});
```

号码认证获取授权令牌

号码认证获取授权令牌,获得 oclToken 和运营商类型，使用 oclToken 与服务端接口交互，验证号码是否一致。参考服务端 REST API 对接指南，本机号码认证 API 部分。

```
/**
  * @param accessCode
  * @param callBack
  */
HXLogin.getInstance().numberValidate(accessCode, this, new HXCallBack() {
    @Override
    public void success(HXResultData resultData) {
        //号码认证获取授权令牌成功成功回调
        //授权令牌 String  resultData.getOclToken()
        //运营商类型 String  resultData.getOperatorType()
    }

    @Override
    public void failed(HXErrorEntity errorEntity) {
        //号码认证获取授权令牌失败回调
    }
});
```

参数名称	参数类型
this	Context
HXCallBack	oclcallback

2. SDK 集成 demo 示例代码如下：

```
1.          //页面设置
2.          HXUIConfig HXUIConfig = HXUIConfig.getInstance();
3.          //设置竖屏
4.          HXUIConfig.setOrientation(HXUIConfig.PORTRAIT);
5.          //设置使用的布局 ID
6.          HXUIConfig.setOauthLayoutIdName("activity_oauth_dialog_land");
7.          //设置使用弹框模式
8.          HXUIConfig.setDialogMode(HXUIUtils.dip2px(OclActivity.this, 450),
9.                                   HXUIUtils.dip2px(OclActivity.this, 300),Gravity.CENTER,0.8f);
10.         //设置关闭弹框模式,默认为关闭状态
11.         HXUIConfig.disableDialogMode();
12.         //设置背景图片名称
13.         HXUIConfig.setBackgroundImgName("授权页面背景图片名称");
14.
15.         //设置加载框
16.         HXUIConfig.setLoadingMode(new HXUIConfig.LoadingMode() {
17.                 @Override
18.                 public void show() {
19.                     showProgressBar();
20.                 }
21.
22.                 @Override
23.                 public void dismiss() {
24.                     hideProgressBar();
25.                 }
26.
27.                 @Override
28.                 public void setLoadingContext(Context context) {
29.                     loadingDialog = new LoadingDialog((Activity) context);
30.                 }
31.             });
32.         //添加自定义点击事件 其中的 false 和 true 为是否需要关闭页面 false 不关闭授权页 true 关闭授权
    页面
33.         HXUIConfig.addCustomView(HXViewEntity.createCustomView("wx_img",false, new HXUICallB
    ack() {
34.                 @Override
35.                 public void clickCallBack(Context context, @Nullable View view) {
36.                     Toast.makeText(OclActivity.this,"微信账号登录被点击了",Toast.LENGTH_SHO
    RT).show();
37.                 }
38.             }));
39.             //true 关闭授权页面
40.             HXUIConfig.addCustomView(HXViewEntity.createCustomView("qq_img",true, new HXU
    ICallBack() {
41.                 @Override
42.                 public void clickCallBack(Context context, @Nullable View view) {
```

```
                Toast.makeText(OclActivity.this,"QQ 账号登录被点击了",Toast.LENGTH_SHORT).show();
            }
        }));
        HXUIConfig.addCustomView(HXViewEntity.createCustomView("wb_img",false, new HXUICallBack() {
            @Override
            public void clickCallBack(Context context, @Nullable View view) {
                Toast.makeText(OclActivity.this,"微博账号登录被点击了",Toast.LENGTH_SHORT).show();
            }
        }));

        CustomProtocol customProtocol = new CustomProtocol();
        //设置默认协议部分
        customProtocol.setNoBookMark(false);//不添加书名号（默认 false） 设为 true 时无书名号
        customProtocol.setPrefix("同意即登录");//设置协议开始文字
        customProtocol.setSeparator("、");//设置协议间隔符
        customProtocol.setProtocolTextColor(R.color.red);//设置协议文字颜色
        customProtocol.setSuffix("并授权 appName 使用");//设置
        customProtocol.setProtocolTextSize(20);//运营商协议字体大小
        customProtocol.setBold(false);//设置默认协议粗体  true 为粗体 false 为正常
        customProtocol.setUnderline(false);//设置协议下划线 true 为带下划线 false 不带

        //设置用户自定义协议
        ProtocolEntity protocolEntity = ProtocolEntity.createNewCustom("自定义协议 1","协议链接");//创建自定义协议 协议名称 协议链接地址
        protocolEntity.setPrefix("和");//自定义协议间隔符
        protocolEntity.setTextColor(R.color.colorAccent);//自定义协议颜色
        protocolEntity.setTextSize(40);//自定义协议字体大小
        protocolEntity.setBold(false);//设置自定协议粗体 true 为粗体 false 为正常字体
        protocolEntity.setUnderline(false);//设置协议下划线 true 为带下划线 false 为不带下划线
        customProtocol.addProtocolEntity(protocolEntity);//把自定义协议添加到协议控制类中
        HXUIConfig.setCustomProtocol(customProtocol);//设置协议
        //设置协议勾选框未勾选时的回调（当用户有特殊需求时）
        HXUIConfig.setCheckBoxUnCheckedCallBackallBack(new HXCheckBoxUnCheckedCallBackallBack() {
            @Override
            public void callBack(Activity activity) {
                Toast.makeText(activity,"请勾选并同意服务条款",Toast.LENGTH_SHORT).show();
            }
        });

        String accessCode;
        HXLogin.getInstance().preLogin(this, new HXCallBack() {
         @Override
         public void success(HXResultData resultData) {
             // TODO: 预取号成功，获取预取号返回的令牌 accessCode
             String accessCode = resultData.getAccessCode();
```

```
    }

     @Override
    public void failed(HXErrorEntity errorEntity) {
        // TODO: 预取号失败
    }
});

/**
 * 拉起授权页面
 * accessCode 预取号成功返回的令牌 accessCode
 */
HXLogin.getInstance().openOauthActivity(accessCode,new HXCallBack() {
    @Override
    public void success(HXResultData resultData) {
        // TODO: 拉起授权页点击一键登录成功返回
        //授权令牌 String  resultData.getOclToken()
        //运营商类型 String  resultData.getOperatorType()
    }
    @Override
    public void failed(HXErrorEntity errorEntity) {
        // TODO: 发起授权页点击一键登录失败返回
    }
});
```

3. 参数说明

成功回调

HXResultData 数据结构

参数	类型	描述
oclToken	String	华信平台授权 token
accessCode	String	预取号令牌
operatorType	String	运营商类别 CU 联通 CT 电信 CM 移动

失败回调

HXErrorEntity 数据结构

参数	类型
errorCode	String
errorMsg	String
errorDetailMsg	String
traceNo	String

4. 响应码

响应码 code	描述	描述详情
GET_CONFIGURATION_ERROR	获取华信 SDK 配置信息失败	对接可能存在错误，请检查配置或联系我们支持
NO_GETPRENUMBER_ERROR	请先调用相应业务的预取号方法	请先调用相应业务的预取号方法
GET_ACCESSCODE_ERROR	运营商预取号失败	用户网络不稳定，可切换纯 4G 操作
LOGIN_FAILURE	华信预取号失败	预下单接口失败，联系我们给予支持
CANCEL	登录取消	预取号成功后，用户点击返回
OPERATOR_INIT_FAILED	请求参数有误	对接可能存在错误，请检查配置或联 2019-09-02 14:28:42 星期一系我们支持

响应码 code	描述	描述详情
INTERNET_UNAVAULABLE	运营商网络异常	未检测到用户数据流量网络，需要用户检查手机网络情况
REQUEST_OPERATOR_TIMEOUT	请求运营商超时	网络异常/手机上网权限没有给 APP/手机 wifi 设置代理/预取号的时候走 http，但是客户 app 设置了只能走 https
PARAM_APP_ID_VALID_NOT_PASS	参数 appKey 不能为空	对接可能存在错误，请检查配置或联系我们支持
PARAM_APP_SECRET_VALID_NOT_PASS	请求参数 appSecret 不能为空	对接可能存在错误，请检查配置或联系我们支持
PARAM_CALL_BACK_VALID_NOT_PASS	回调函数 callBack 不能为空	对接可能存在错误，请检查配置或联系我们支持
PARAM_CONTEXT_VALID_NOT_PASS	上下文内容 context 不能为空	对接可能存在错误，请检查配置或联系我们支持
CHECK_PERMISSION_IS_OPEN	请使用运营商网络	手机上网权限没有给 APP
SERVICE_UNAVAILABLE	亲，服务不可用，请稍后重试！	系统内部异常

响应码 code	描述	描述详情
GET_ACCESSTOKEN_ERROR	运营商登录失败	获取运营商 token 失败，可能是网络不稳定 或者未开启数量流量网络
PACKAGE_NAME_ERROR	包名错误	对接可能存在错误，请检查配置或联系我们支持
APP_SIGN_ERROR	签名错误	对接可能存在错误，请检查配置或联系我们支持
PARAM_DECRYPT_ERROR	请求报文解析失败	对接可能存在错误，请检查配置或联系我们支持
OPERATOR_OTHER_ERROR	运营商内部错误	用户网络异常（包含网络不稳定，未连接， 切换异常，域名解析异常等）
OPERATOR_RESTRICT_REQUESTS	运营商限制请求	操作频繁，请稍后再试
APP_INFO_FAILED	应用信息错误	应用信息错误（对接可能存在错误， 请检查配置或联系我们支持）
APP_NOT_EXIST	应用不存在	应用不存在（对接可能存在错误， 请检查配置或联系我们支持）
OPERATOR_NUMBER_FAILED	运营商取号失败	取号失败可能性较多，如：无权限访问网络、ip 受限

响应码 code	描述	描述详情
OTHER_ERROR	其它异常、SDK 内部异常	—

5. 注意事项

Q:如何自定义授权页面 UI

A:用户可以自行编写授权页面 XML，但需将保留必要的组件 ID，详细请参考文档 SDK 代码集成-设置自定义 U

Q:如何设置协议

A:用户可自定义协议头部，尾部内容，可设置默认协议字体大小，颜色，是否带书名号，对于用户自己的协议，书名号由用户传入：例如”《用户协议》”，用户可自定义自定义协议的字体大小，内容，颜色，字体样式以及间隔符等，具体可参考文档-设置协议