目录

01.集成概述

说明

本文档旨在帮助 Android 应用开发者在程序中快速植入 2345 云图广告平台的广告。作为应用开发者，您只需要进行简单配置，就可以在您的应用中显示定制的广告。关于 SDK 的具体使用方法，请仔细阅读下面的文档。

背景

开发环境

- **操作系统**：支持 Linux/Mac/Windows 系统，具体依赖开发者选择的 IDE
- **开发工具**：支持 Android studio、Eclipse、Intellij
- **支持设备**：运行了 Android 4.4 以及以上系统的 Android 设备
- **接入方式**：请参考 ：依赖配置

术语介绍

- **appId**：商业化媒体 ID，这个 ID 是我们在广告网络中识别您应用的唯一 ID。
- **pluginJarChannel**：商业化插件渠道 ID，这个 ID 是我们在广告 SDK 中唯一标识插件升级渠道的 ID。
- **appChannelId**：宿主渠道 ID，这个 ID 是我们在广告网络中识别相同包名宿主，但是安装渠道不同的 ID。
- **代码调用**：指宿主通过调用 API 方法，请求加载广告数据或者直接打开广告界面的一种方式

02. 依赖配置

一、依赖添加

build.gradle 文件下：

```
dependencies {

    implementation 'com.google.android.material:material:1.0.0'
    implementation 'androidx.appcompat:appcompat:1.0.0'
```

```
    implementation 'androidx.legacy:legacy-support-v4:1.0.0'
    implementation 'androidx.recyclerview:recyclerview:1.0.0'

    // 武林榜
    implementation(name:'wlbsdk-release-5.2.0',ext:'aar')
    // 环境切换
    implementation(name:'EnvSwitcher-1.0.2',ext:'aar')
    implementation "com.google.code.gson:gson:2.8.5"
    implementation "com.tencent:mmkv-static:1.0.19"
    // 插件化 SDK
    implementation(name:'host-2.1.2',ext:'aar')
    // 动态模板
    implementation(name:'dynamic-view-1.0.0',ext:'aar')
    // 广告相关
    implementation(name:'adware-3.6.0',ext:'aar')
    implementation(name:'adware-bd-1.0.3',ext:'aar')
    implementation(name:'adware-gdt-1.0.7',ext:'aar')
    implementation(name:'adware-csj-1.0.6',ext:'aar')
    implementation(name:'adware-ks-1.0.4',ext:'aar')
    implementation(name:'adware-sigmob-1.0.2',ext:'aar')
    implementation(name:'sdk-9.062',ext:'aar')
    implementation(name:'sdk-4.371.1241',ext:'aar')
    implementation(name:'sdk-3.6.1.7',ext:'aar')
    implementation(name:'sdk-3.3.14',ext:'aar')
    implementation(name:'sdk-3.0.1',ext:'aar')
}
```

SDK 可集成版本请查看：修订日志

注意：若宿主适配 AndroidX 后遇到依赖的云图 SDK 插件包加载失败，请在 project 的 gradle.properties 添加如下配置

```
# 多个 aar 用,号分割
android.jetifier.blacklist=adware-.*\\.aar
```

二、配件文件（必选）

云图广告 SDK 需要将如下格式的配置文件，添加到宿主 APP assets 目录下（配置文件请联系商业化产品运营提供）

以下文件为 DEMO 示例，宿主请勿直接使用

文件名：cloud_ad_config.json （配置文件名不可修改）

文件内容：

```
{
  "appId": "10000",
  "pluginJarChannel": "ceshizhuanyong",
  "thirdSdkConfig": {
    "csj": {
      "appId": "5001121"
    },
    "baidu": {
      "appId": "d03bfa72"
    },
    "gdt": {
      "appId": "1101152570"
    },
    "ks": {
      "appId": "90009"
    },
    "sigmob": {
      "appId": "2342",
      "appKey": "daee6486ef2e4324"
    }
  },
  "adSenseConfig": {
    "splash": {
      "adSenseId": "100090"
    },
    "interstitial": {
      "adSenseId": "100238"
    },
    "rewardVideo": {
      "adSenseId": "100104"
    }
  }
}
```

03. 初始化 SDK

一、初始化方法

接入代码示例

初始化之前配置（可选）

```
//是否使用插件加载优化（使用之后会缓存插件信息），默认使用，提高插件加载速度
CloudSdk.usePluginLoadOptimization(true);
//是否允许初始化 SDK 同时初始 UserAgent，默认允许，提高开屏配置接口更新成功率（因为 getUserAgent 方法耗时较长，但是允许之后会增加初始化 SDK 时长，大约 100~200ms 左右，请宿主可根据实际需求设置）
CloudSdk.setAllowInitUserAgent(true);
```

setAllowInitUserAgent 设置为 true 之后，大约会**增加初始 SDK 的时长 100-200ms（和手机系统性能有关）左右**，但是会**提高开屏接口更新的成功率（因为 getUserAgent 方法耗时较长，在性能较低的手机上很容易导致配置更新超时）**，进而提高首次安装开屏广告请求的成功率。**宿主设置为 true 或者 false 均不会影响 SDK 的正常使用，请宿主根据实际需要设置是否允许**

如果宿主考虑初始化时长问题，请考虑使用 setAllowInitUserAgent 设置为 false 的方式

因为实际测试 Application onCreate 方法里面立即切换子线程调用 getUserAgent 方法会耗时非常长（200-800ms 左右），这个时间内开屏流程已经开始执行了，切换子线程初始化 SDK 的方式并不能起到提高开屏配置请求成功率的作用

初始化方法调用（必选）

请在 Application 类的 onCreate() 回调中调用如下方法初始化 SDK（务必仅在主进程中调用初始化方法）

```
// 武林榜 SDK 初始化（必须在云图广告 SDK 前初始化）
WlbStatistic.init(this);
// 环境切换 SDK 初始化
EnvSwitcher.init(this);
// 云图广告初始化
CloudSdk.init(this, appChannelId, new CloudSdk.InitializeCallBack() {
        @Override
        public void onFailed(String msg) {
            //SDK 初始化失败，msg：错误信息
        }

        @Override
        public void onSuccess() {
            //SDK 初始化成功
        }
    });
```

说明：

1、appChannelId 为宿主渠道 ID，可不传，不传则默认读取 minifest 文件中的 UMENG_CHANNEL 值，配置示例如下：

```
<meta-data
        android:name="UMENG_CHANNEL"
        android:value="宿主渠道 ID" />
```

2、init 方法有四个重载方法，选择其中一个调用即可，具体方法说明请查看：CloudSDK API 说明

二、权限申请

1、广告 SDK 仅在 manifest 文字声明必要的申请，但是不会主动申请任何危险权限

目前广告 SDK 及其聚合的第三方广告 SDK 声明的所有权限如下（以下权限已在 SDK AAR Manifest 中声明，无需宿主单独声明）：

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.REQUEST_INSTALL_PACKAGES" />
<uses-permission android:name="android.permission.GET_TASKS" />
<uses-permission android:name="android.permission.DISABLE_KEYGUARD" />
<!-- Optional for location -->
<uses-permission
android:name="android.permission.ACCESS_LOCATION_EXTRA_COMMANDS" />
<!-- 如果有视频相关的广告且使用 textureView 播放，请务必添加，否则黑屏 -->
<uses-permission android:name="android.permission.WAKE_LOCK" />
<!-- 获取用户设备的蓝牙地址 -->
<uses-permission android:name="android.permission.BLUETOOTH" />
<!-- 开机启动，应用到达统计需要 （Rom 推广渠道） -->
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" />
<uses-permission android:name="android.permission.QUERY_ALL_PACKAGES" />
<!-- SDK 内需要的权限，与下载相关 -->
<permission
        android:name="${applicationId}.permission.KW_SDK_BROADCAST"
        android:protectionLevel="signature" />

<uses-permission android:name="${applicationId}.permission.KW_SDK_BROADCAST" />
<!-- To allow starting foreground services on Android P+ -
https://developer.android.com/preview/behavior-changes#fg-svc -->
<uses-permission android:name="android.permission.FOREGROUND_SERVICE" />
<uses-permission android:name="com.asus.msa.SupplementaryDID.ACCESS" />
<!-- <uses-permission android:name="android.permission.CAMERA" /> -->
```

SDK 不强制校验上述权限（即:无上述权限 sdk 也可正常工作），但建议您申请上述涉及的危险权限，以便广告 SDK 提供更精准优质的广告

2、百度 SDK 广告权限获取设置

```
// 设置 SDK 可以使用的权限，包含：设备信息、定位、存储、APP LIST
// 注意：建议授权 SDK 读取设备信息，SDK 会在应用获得系统权限后自行获取 IMEI 等设备信息
// 授权 SDK 获取设备信息会有助于提升 ECPM
MobadsPermissionSettings.setPermissionReadDeviceID(true);
MobadsPermissionSettings.setPermissionLocation(true);
MobadsPermissionSettings.setPermissionStorage(true);
MobadsPermissionSettings.setPermissionAppList(true);
```

三、混淆配置

云图广告 SDK 已配置 consumerProguardFiles，无需手动添加混淆。

1、如果使用 R8 编译器（Gradle 插件 3.4.0 及以上版本默认）则不需要配置，压缩和混淆规则已经打包 Jar 中，R8 能够自动解释运行。

2、如果指定了使用 proGuard 则需要单独配置混淆规则（因为集成版本混淆规则变动，请联系商业化开发单独提供）

四、登录和退出登录方法

接入代码示例

CloudSDK API 说明

1、请在合适的时机调用登录方法，传入用户 passId 和是否是游客标识

2、请在用户退出登录时，调用退出登录方法，清除 SDK 中缓存的用户 ID

登录和退出登录方法，均不依赖 SDK 是否初始化成功结果，合适的时机调用即可

```
/**
 * 登录方法
 *
 * @param passId    passId 用户 Id
 * @param isTourist 是否是游客
```

```
 */
CloudSdk.login(passId, boolean isTourist)


//退出登录方法
CloudSdk.logout()
```

五、落地页 UI 样式配置

接入代码示例

[CloudSdkSetting API 说明]

（可选配置）

```
CloudSdkSetting.configLandingPage(
                new LandingPageSetting.Builder()
                        // 标题栏标题颜色
                        .setTitleColor("#FFFFFF")
                        // 标题栏背景色
                        .setTitleBarBackgroudColor("#008B8B")
                        // webview loading 进度条颜色
                        .setLoadingProgressBarColor("#FF6A6A")
                        // 标题栏高度 单位：px
                        .setTitleBarHeight(DimenUtils.dp2px(this, 50))
                        .build());
```

六、敏感信息配置

接入代码示例

[CloudSdkSetting API 说明]

必须在初始化前调用

（可选配置）

```
CloudSdkSetting.configSensitiveSetting(
                new SensitiveSetting.Builder()
                            .setCanGetAppList(ISensitiveSetting.ENABLE)
```

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

04.开屏广告

一、代码调用

接入代码示例

```
//(详细内容请参考压缩包中的代码示例,demo 类：SplashActivity)
//广告请求参数，具体参数说明请查看主要 API 说明
/**
  * activity 当前请求的开屏广告依附页面（必填）
  * splashContainer 开屏 View 容器（必填）
  *skipView 自定义跳过按钮（可选,不填使用商业化默认跳过按钮。请传入真实的 View，并勿对此 View 设置点击事件）
  *screenWidth screenHeight 容器宽度和容器高度（可选，建议传递，否则可能会导致第三方广告请求响应时长增加）
  * doubleSplash 设置是否支持算开屏，true 支持（可选，v3.5.0 版本新增方法，设置为 true 仅表示宿主需要支持,还需要根据广告配置和物料返回结果判断实际是否展示双开）
  * timeout 广告请求超时时长（可选，不填默认 3000ms）
```

```
 */
mSupportCustomSkipButton = true;
SplashRequestParam param = new SplashRequestParam.Builder()
                .setActivity(activity)
                .setContainer(splashContainer)
                .setSkipView(skipView)
                .setContainerSize(screenWidth, screenHeight)
                .setTimeout(5000)
                .build();
//SplashLoadListener，广告请求状态回调
//loadSplash 请求开屏广告方法
CloudSdkManager.loadSplash(param, new SplashLoadListener() {

            @Override
            public void onClose() {
                //广告关闭回调（部分广告源如果点击跳转落地页后，还会继续倒计时，可能在广告落地页的时候回调关闭，请注意此种情况下的关闭回调逻辑处理）
            }

            @Override
            public void onError(CloudError adError) {
                                //请求失败，失败原因:adError.getMessage()

            }

            @Override
            public void onClick() {
                //广告点击回调
            }

            @Override
            public void onShow() {
                //广告有效曝光回调
            }

                        //3.3.0 版本新增回调，因为开屏存在覆盖的情况，此回调可能在曝光之前也可能在曝光之后，请宿主收到回调后如果判断不支持立即隐藏跳过按钮，同时在 onTick 回调和 onPresent 回调中（默认支持）判断如果不支持及时隐藏
                        @Override
            public void onCustomSkipButton(boolean isSupport) {
```

```
                                    //记录是否支持自定义跳过按钮，并隐藏自定义跳过按钮
                                    //因为开屏存在覆盖的情况，收到此回调并且不支持自定义时立即隐藏跳过按钮
                    mSupportCustomSkipButton = isSupport;
                    if (!mSupportCustomSkipButton) {
                        mSkipView.setVisibility(View.GONE);
                    }
                }

                @Override
                public void onTick(long millisUntilFinished) {
                    //倒计时 millisUntilFinished 剩余时长
                                    //设置倒计时提示文案
                    //if (mSkipConfig.isChecked() && mSupportCustomSkipButton) {
                    //    mSkipView.setVisibility(View.VISIBLE);
                    //    String text = millisUntilFinished / 1000 + "秒";
                    //    mSkipView.setText(text);
                    //}
                }

                @Override
                public void onPresent() {
                    //广告请求成功曝光回调,曝光仅代表广告请求成功,不代表广告有效曝光
                }
            });
```

二、主要 API 说明

CloudSdkManager

请查看 [CloudSdkManager 类 API 说明文档]

方法名	方法介绍
static void loadSplash(SplashRequestParam	请求开屏广告方法。参数说明：params（开屏广告请求参数：SplashRequestParam ），listener（开屏广告请求状态监听：SplashLoadListener ）

params, SplashLoadListener listener)	

SplashRequestParam

请使用 SplashRequestParam.Bulider 类构建 SplashRequestParam 对象

SplashRequestParam.Bulider

方法名	方法介绍
Builder setActivity(Activity activity)	（**必填项**）设置当前请求开屏广告的 Activity。参数说明：activity（当前请求的开屏广告依附页面）
Builder setContainer(ViewGroup container)	（**必填项**）设置开屏展示容器。参数说明：container（开屏展示容器）
Builder setSkipView(View skipView)	（**可选设置**）设置自定义跳过按钮。参数说明：skipView（自定义跳过按钮）。（**如需设置请传入真实的 View，View 设置条件：1、大小不能小于 3dp*3dp；2、可见状态不能是 gone；3、不能对此 View 设置点击事件**）
Builder setContainerSize(int containerWidth, int containerHeight)	（**可选设置**）设置开屏展示容器宽高。参数说明：containerWidth（容器宽度，单位 px），containerHeight（容器高度，单位 px）。（**建议传递，否则会因为及时容器的宽高导致第三方广告请求响应时长增加**）
Builder setAdSenseId(String adSenseId)	（**可选设置**）设置开屏广告位 ID。参数说明：adSenseId（开屏广告位 ID）。（**不设置，则使用配置文件中的默认广告位请求开屏**）
Builder setTimeout(int timeout)	（**可选设置**）设置开屏广告请求超时时长。参数说明：timeout，单位 ms（开屏广告请求超时时长）。（**不设置，则默认 3000ms**）
SplashRequestParam build()	构建 SplashRequestParam 对象

SplashLoadListener

方法名	方法介绍
void onTick(long millisUntilFinished)	开屏广告展示倒计时回调。参数说明：millisUntilFinished（剩余展示时长）。（如果宿主设置了自定义跳过按钮，可以此回调中根据剩余展示时长结果设置提示）
void onPresent()	开屏广告请求成功曝光回调。(此曝光不等于有效曝光)
void onError(CloudError adError)	开屏请求失败回调。参数说明：adError（错误信息）
void onShow()	开屏广告有效曝光回调。
void onClick()	开屏广告点击回调。
void onClose()	开屏广告关闭回调。（部分广告源如果点击跳转落地页后，还会继续倒计时，可能在广告落地页的时候回调关闭事件，请注意此种情况关闭回调下的逻辑处理）
void onCustomSkipButton(boolean isSupport)	当前广告是否支持自动跳过按钮回调。参数说明：isSupport（true 支持自定义（默认），false 不支持） （3.3.0 版本新增回调，因为开屏存在覆盖的情况，此回调可能在曝光之前也可能在曝光之后，请宿主收到回调后如果判断不支持立即隐藏跳过按钮，同时在 onTick 回调和 onPresent 回调中（宿主可设置默认支持，没有收到 onCustomSkipButton 回调前）判断如果不支持及时隐藏）

SplashLoadExpandListener

（继承 SplashLoadListener 接口）

方法名	方法介绍
void onLoad(ISplash splash)	开屏广告请求成功(请求成功不代表展示成功)。参数说明：splash（广告数据）； v3.3.0 版本新增方法，使用示例可参考开屏点睛接入示例

CloudSdkSetting

1、宿主只有需要使用广告 SDK 的热启动开屏广告时，才需要设置如下配置，其他情况请勿设置

2、设置之后，打开的热启动开屏页（Activity）为广告 SDK 的页面，和宿主的开屏页没有关系，请勿重复展示，以免造成用户体验问题

3、设置之后，不一定会展示广告 SDK 的热启动开屏页，还需要根据当前开屏广告位请求到配置规则支持才会展示

<table>
<tr><th>方法名</th><th>方法介绍</th></tr>
<tr><td>static void configSplashPage(

SplashPageSetting setting)</td><td>（如果宿主已经自定义了热启动开屏，请勿设置此配置）

设置广告 SDK 的热启动开屏配置。参数说明：setting（开屏配置）

SplashPageSetting（使用 SplashPageSetting.Builder 类构建 SplashPageSetting 对象）

SplashPageSetting.Builder
<table>
<tr><th>方法名</th><th>方法介绍</th></tr>
<tr><td>Builder setBackgroundResId(int backgroudResId)</td><td>（必填项）设置背景图片。参数说明：backgroundResId（背景图片资源 ID）</td></tr>
<tr><td>Builder setLogoResId(int logoResId)</td><td>（可选设置）设置 logo 图片。参数说明：logoResId（logo 图片资源 ID）</td></tr>
<tr><td>Builder setLogoHeight(int logoHeight)</td><td>（可选设置）设置 logo 展示高度。参数说明：logoHeight（logo 展示高度）单位：px</td></tr>
<tr><td>Builder setTimeoutMillis(int timeoutMillis)</td><td>（可选设置）设置广告请求超时时长。参数说明：timeoutMillis，单位 ms（请求超时时长，默认 3000ms）</td></tr>
<tr><td>Builder setAdSenseId(int adSenseId)</td><td>（可选设置）设置热启动开屏的广告位 ID，参数说明：adSenseId（开屏请求广告位 ID，默认使用配置文件的默认广告位）</td></tr>
</table></td></tr>
</table>

CloudSdkLoadListener

1、宿主只有通过开屏广告 Deeplink，打开广告 SDK 的开屏广告页面（Activity）,才有可能会调用下如下的注册和解注册开屏广告监听方法，其他情况请勿调用

2、具体 Deeplink 请查看：云图广告 Deeplink 文档

方法名	方法介绍
static void registerSplashLoadListener(SplashLoadListener listener)	注册开屏广告状态监听。参数说明：listener（开屏广告请求状态监听：SplashLoadListener ）。（此监听注册后，仅会收到 deeplink 方式打开的广告 SDK 的开屏页面广告请求回调，且监听变量仅保存一个，后一次注册会替换前一次注册）
static void registerSplashLoadListener(SplashLoadListener listener)	解注册开屏广告状态监听。参数说明：listener（开屏广告请求状态监听：SplashLoadListener ）。（请在合适的时机销毁，SDK 内部使用静态变量保存，反注册不及时会造成内存泄露）

三、注意事项

3.1、关于广告关闭回调

Q：为什么跳转落地页后，广告会继续回调关闭按钮

A：由于第三方广告 SDK 开屏广告回调逻辑的问题，部分广告 SDK 点击之后会逻辑回调关闭或者 跳转落地页之后继续倒计时直至回调关闭，这种情况，宿主如果是跳转新的首页需要做特殊逻辑处理。处理逻辑见如下红字部分

：判断页面如果 onPause 状态之后才返回的广告 onClose 回调，则不直接跳转首页（如果有这个逻辑的话），等待开屏页 onResume 之后在判断已经 close 了，再处理相关的跳转逻辑

3.2、关于广告跳过按钮

Q：为什么自定义广告跳过按钮必须设置可见

A：因为部分第三方 SDK 的限制，如果广告曝光之后，跳过按钮不可见，则会直接回调开屏展示失败。处理逻辑见如下红字部分：

：View 设置条件：1、大小不能小于 3dp*3dp；2、可见状态不能是 gone；3、不能对此 View 设置点击事件。（！！！强烈建议一直设置成 VISIBLE）

（！！！如果实在没有办法，需要先设置 VIEW 是 INVISIBLE 的，请务必！！！在曝光回调之后立即设置为 VISIBLE，否则会导致广告展现异常；此种情况因为曝光和倒计时回调之间有一定的时长间隔，请做好此种情况下的 UI 展示兼容）

四、开屏点睛

3.3.0 版本新增，如果宿主不需要展示开屏点睛样式可忽略

当前仅如下广告源支持开屏点睛功能：

1、穿山甲广告（支持版本 3.3.0，如果宿主自定义了跳过按钮，则任何情况下都不支持开屏点睛）
2、广点通广告（支持版本 3.5.0）

开屏点睛的交互方式：在开屏呈现的过程中用户点击右上角的“跳过”或曝光结束后开屏图片或者视频将收缩到 APP 内右下角的小视窗继续展示。

交互样式需要宿主自定义实现，SDK 仅提供当前广告是否支持开屏点睛样式回调，以及样式展示之后通知 SDK 动画播放完成方法，具体方法和示例如下

样式示例

代码实现示例

具体示例代码请查看 SplashActivity，SplashClickEyeManager（主要用于实现开屏点睛动画管理类），MainActivity

```
//(详细内容请参考压缩包中的代码示例,demo 类：SplashActivity)
//第一步：使用 SplashLoadExpandListener 注册开屏请求监听，新增 onLoad 回调开屏广告数据
        mIsSdkAdTimeFinish = false;
        CloudSdkManager.loadSplash(buildRequestParams(), new
SplashLoadExpandListener() {

            //...

            @Override
            public void onLoad(ISplash splash) {
                LogUtil.v("splash111111111", "广告请求成功，准备展示");
                initSplashClickEyeData(splash);
            }

            //...
        });
    }


  //第二步：调用 setSplashClickEyeListener 方法注册开屏点睛相关监听
  private void initSplashClickEyeData(ISplash splashAd) {
        if (splashAd == null) {
            return;
        }
        splashAd.setSplashClickEyeListener(new
SplashClickEyeListenerImpl(SplashActivity.this, splashAd, mIsSplashClickEye ||
mMainActivity, mMainActivity));
    }

    //第三步：isSupportSplashClickEye 回调中判断，如果支持则保存广告数据
        //第四步：如果宿主首页和开屏页是同一个 Activity，则在
onSplashClickEyeAnimationStart 回调中处理广告 View
        //第五步：如果宿主首页和开屏页不是一个 Activity，跳过第四步，直接再首页中处理广告 View 动画展现
```

```
    public static class SplashClickEyeListenerImpl implements
SplashClickEyeListener {
        private final SoftReference<Activity> mActivity;
        private final ISplash mSplashAd;
        private final boolean mIsFromSplashClickEye;
        private final boolean mJumpMainActivity;

        public SplashClickEyeListenerImpl(Activity activity, ISplash splashAd,
boolean isFromSplashClickEye, boolean jumpHomeActivity) {
            mActivity = new SoftReference<>(activity);
            mSplashAd = splashAd;
            mIsFromSplashClickEye = isFromSplashClickEye;
            mJumpMainActivity = jumpHomeActivity;
        }

        @Override
        public void onSplashClickEyeAnimationStart() {
            //开始执行开屏点睛动画
            if (mIsFromSplashClickEye && !mJumpMainActivity) {
                startSplashAnimationStart();
            }
        }

        @Override
        public void onSplashClickEyeAnimationFinish() {
            //sdk 关闭了了点睛悬浮窗
            if (mIsFromSplashClickEye ) {
                finishActivity();
            }

        }

        @Override
        public boolean isSupportSplashClickEye(boolean isSupport) {
            LogUtil.v("splash111111111","物料是否支持开屏点睛：" + isSupport);
            if (!mIsFromSplashClickEye) {
                return false;
            }
            SplashClickEyeManager splashClickEyeManager =
SplashClickEyeManager.getInstance();
            Activity activity = mActivity.get();
```

```
            if (activity != null && !activity.isFinishing()) {
                splashClickEyeManager.setSplashInfo(mSplashAd,
mSplashAd.getSplashView(), activity.getWindow().getDecorView());
                splashClickEyeManager.setSupportSplashClickEye(isSupport);
            }
            return true;
        }

        private void finishActivity() {
            if (mActivity.get() == null) {
                return;
            }
            mActivity.get().finish();
        }

        private void startSplashAnimationStart() {
            if (mActivity.get() == null || mSplashAd == null) {
                return;
            }
            SplashClickEyeManager splashClickEyeManager =
SplashClickEyeManager.getInstance();
            ViewGroup content =
mActivity.get().findViewById(android.R.id.content);

splashClickEyeManager.startSplashClickEyeAnimation(mSplashAd.getSplashView(),
content, content, new SplashClickEyeManager.AnimationCallBack() {
                @Override
                public void animationStart(int animationTime) {
                }

                @Override
                public void animationEnd() {
                    mSplashAd.splashClickEyeAnimationFinish();
                }
            });
        }
    }
```

相关 API

ISplash

方法名	方法介绍
splashClickEyeAnimationFinish()	该方法需要开发者在点睛动画结束时，主动调用通知广告 SDK 去做点睛 view 的改变逻辑。
setSplashClickEyeListener(ISplashClickEyeListener listener)	通过将监听传递给广告 SDK，广告 SDK 在合适的时机触发 ISplashClickEyeListener 的各个接口方法，通知给开发者做不同操作。
int[] getSplashClickEyeSizeToDp()	传递给开发者的点睛 View 的推荐尺寸，建议开发者先使用该值去确定最终点睛 View 的大小。（可能为空，做好空判）
View getSplashView()	获取开屏展示 View（可能为空，做好空判）

SplashClickEyeListener

方法名	方法介绍
onSplashClickEyeAnimationStart()	3.3.0 版本以及之后版本，当满足开屏点睛条件时触发，开发者可以在该回调中开始展示点睛动画。
onSplashClickEyeAnimationFinish()	3.3.0 版本以及之后版本，当关闭点睛 view 时触发，开发者可以在该方法中做些资源回收，关闭开屏的操作。
isSupportSplashClickEye(boolean isSupport)	3.3.0 版本以及之后版本，当满足开屏点睛条件时触发，开发者可以在该方法中保持该 boolean 值，用于开屏两个 activity 判断是否需要展示开屏点睛时使用。

05. 插屏广告

一、代码调用

接入代码示例

```
//(详细内容请参考压缩包中的代码示例,demo 类：InterstitialActivity)

//广告请求参数，具体参数说明参考 API 说明
String adSenseId = "";

//请求插屏广告
CloudSdkManager.loadInterstitial(new
InterstitialRequestParam.Builder().setAdSenseId(adSenseId).build(),
        new InterstitialLoadListener() {
        @Override
        public void onLoaded(boolean isDownload) {
                showToast("onAdLoaded isDownload:" + isDownload);
        }

        @Override
        public void onError(CloudError adError) {
                if (adError != null) {
                showToast("onAdError code:" + adError.getCode() + ",msg:" +
adError.getMessage());
                } else {
                showToast("onAdError");
                }
        }

        @Override
        public void onShow() {
                showToast("onAdShow");
        }

        @Override
```

```
        public void onClick() {
                showToast("onAdClick");
        }

        @Override
        public void onClose() {
                showToast("onAdClose");
        }
});
```

二、主要 API 说明

CloudSdkManager

请查看 CloudSdkManager 类 API 说明文档

方法名	方法介绍
static void loadInterstitial(InterstitialRequestParam params, InterstitialLoadListener listener)	请求插屏广告方法。参数说明：params（广告请求参数：InterstitialRequestParam ），listener（广告请求状态监听：InterstitialLoadListener ）

InterstitialRequestParam

请使用 InterstitialRequestParam__.Bulider 类构建 InterstitialRequestParam 对象

InterstitialRequestParam.Bulider

方法名	方法介绍
Builder setAdSenseId(String adSenseId)	（**必填项**）设置插屏广告位 ID。参数说明：adSenseId（插屏广告位 ID）
InterstitialRequestParam build()	构建 InterstitialRequestParam 对象

InterstitialLoadListener

方法名	方法介绍
void onLoaded()	广告请求成功回调
void onError(CloudError adError)	请求失败回调 参数说明：adError（错误信息）
void onShow()	广告曝光回调
void onClick()	广告点击回调
void onClose()	广告关闭回调

06. 激励视频广告

一、代码调用

接入代码示例

```
//(详细内容请参考压缩包中的代码示例,demo 类：RewardVideoActivity)
//广告请求参数，具体参数说明参考 API 说明
/**
  * activity    当前请求的激励视频广告依附页面（必填）
  * playPolicy 播放策略（可选项。默认 0、缓存下载视频完成后回调成功；1、视频广告数据请求成功后（不等视频物料下载成功）即回调成功），
  * loadMode    广告请求加载模式（可选项。默认 0、请求广告之后立即播放视频；1、预加载，只请求广告数据不播放视频。）
  * adSenseId   激励视频广告位 ID（必填）
  */
```

```
int loadMode = mLoadModeCb.isChecked() ? RewardVideoRequestParam.LOAD_LAZY :
RewardVideoRequestParam.LOAD_NOW;
int playPolicy = mPlayPolicyCb.isChecked() ?
RewardVideoRequestParam.PLAY_WHEN_CACHED :
RewardVideoRequestParam.PLAY_AFTER_CACHED;
//请求激励视频广告
CloudSdkManager.loadRewardVideo(new RewardVideoRequestParam.Builder()
                .setActivity(this)
                .setLoadMode(loadMode)
                .setPlayPolicy(playPolicy)
                .setAdSenseId(adSenseId).build(), new RewardVideoLoadListener()
{
            @Override
            public void onSuccess(CloudRewardVideo cloudRewardVideo) {
                LogUtil.v("demo 测试", "激励视频请求成功");
                if (RewardVideoRequestParam.LOAD_LAZY == loadMode) {
                    mRewardVideo = cloudRewardVideo;
                    showToast("预加载请求成功，缓存数据到本地");
                }
            }

            @Override
            public void onError(CloudError adError) {
                LogUtil.v("demo 测试", "激励视频请求失败，失败原因：" +
adError.getMessage());
                if (RewardVideoRequestParam.LOAD_LAZY == loadMode) {
                    showToast("预加载请求失败");
                }
            }

            @Override
            public void onClick() {
                LogUtil.v("demo 测试", "激励视频点击回调");
            }

            @Override
            public void onClose() {
                LogUtil.v("demo 测试", "激励视频关闭");
            }

            @Override
```

```
            public void onReward() {
                LogUtil.v("demo 测试", "激励视频奖励回调");
            }

            @Override
            public void onVideoError(CloudError adError) {
                LogUtil.v("demo 测试", "失败播放错误，错误原因：" +
adError.getMessage());
            }

            @Override
            public void onVideoStart() {
                LogUtil.v("demo 测试", "激励视频开始播放");
            }

            @Override
            public void onVideoCompleted() {
                LogUtil.v("demo 测试", "激励视频播放完成");
            }

            @Override
            public void onSkipVideo() {
                LogUtil.v("demo 测试", "激励视频播放跳过");
            }
        });
```

二、主要 API 说明

CloudSdkManager

请查看 CloudSdkManager 类 API 说明文档

方法名	方法介绍
static void loadRewardVideo(RewardVideoRequestParam params, RewardVideoLoadListener listener)	请求激励视频广告方法。参数说明：params（广告请求参数：RewardVideoRequestParam ），listener（广告请求状态监听：RewardVideoLoadListener ）

RewardVideoRequestParam

请使用 RewardVideoRequestParam.Bulider 类构建 RewardVideoRequestParam 对象

RewardVideoRequestParam.Bulider

方法名	方法介绍
Builder setActivity(Activity activity)	（必填项）设置当前请求广告的 Activity。参数说明：activity（当前请求广告依附的页面）
Builder setLoadMode(int loadMode)	（可选设置）设置广告请求加载模式。参数说明：loadMode（加载模式，默认 0、请求广告之后立即播放视频；1、预加载，只请求广告数据不播放视频。）
Builder setPlayPolicy(int playPolicy)	（可选设置）设置播放策略。参数说明：playPolicy（播放策略，0、缓存下载视频完成后回调成功；1、广告数据请求成功后（不等视频物料下载成功）即回调成功）。（回调成功之后即可以调用 showRewardVideo 方法展示激励视频页面 Activity） （部分广告源（自有和 SIGMOB）不支持数据请求成功后回调成功，如果请求的是这些广告源的广告均按照缓存完成后再回调成功处理） （该字段控制的仅仅是广告请求成功回调的时机，如果设置是模式 1，请求广告数据成功后即回调成功，缓存一段时间之后再调用 show 方法展示，可能这个时候视频物料也已经下载完成了）
Builder setAdSenseId(String adSenseId)	（必填项）设置激励视频广告位 ID。参数说明：adSenseId（激励视频广告位 ID）
RewardVideoRequestParam build()	构建 RewardVideoRequestParam 对象

RewardVideoLoadListener

方法名	方法介绍
void onSuccess(	广告请求成功回调。参数说明：rewardVideo（回调成功之后即可以调用 showRewardVideo 方法展示激励视频页

CloudRewardVideo rewardVideo)	面 Activity）
void onError(CloudError adError)	请求失败回调。参数说明：adError（错误信息）
void onClick()	广告点击回调
void onClose()	广告关闭回调
void onReward()	广告奖励回调（观看视频大于一定时长或者视频播放完毕回调，不一定会回调）
void onVideoError(CloudError adError)	广告播放错误回调。参数说明：adError（错误信息）
void onVideoStart()	视频开始播放回调
void onVideoCompleted()	视频播放完成回调
void onSkipVideo()	视频播放跳过回调（部分广告源不会回调跳过事件）

CloudRewardVideo

激励视频广告数据

方法名	方法介绍
void showRewardVideo(Activity activity)	展示播放激励视频(**仅预加载模式下返回的数据可调用该方法，改方法只能调用一次，调用后该数据失效**) 注意:这个时候激励视频页面还是有可能会有**页面内部的loading**，第三方 SDK 可能会在初始化界面，播放器等情况下展示短时间的 loading，这个并不是广告还在请求的 loading，和广告请求快慢无关

07. draw 视频广告（外拓）

- DrawLoadListener

一、代码调用

接入代码示例

```
public class DrawActivity extends BaseActivity {

    ViewGroup mDrawContainer;
    int mAcceptedWidth;
    int mAcceptedHeight;
    ICloudDraw mCloudDraw;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_draw);
        setOnClickListener(R.id.draw_ad_display_ll);
    }

    public void requestDrawAd(View view) {
        String adSenseId = "";
        DrawRequestParam param = new DrawRequestParam.Builder()
                .setAcceptedExpressViewSize(mAcceptedWidth, mAcceptedHeight)
                .setAdSenseId(slotId).build();
        CloudSdkManager.loadDraw(param, new DrawLoadListener() {
            @Override
            public void onLoaded(List<? extends ICloudDraw> iCloudDraws) {
                onDrawAdLoaded(iCloudDraws);
            }

            @Override
            public void onError(CloudError adError) {
                int code = 0;
                String msg = "draw 视频请求失败";
                if (adError != null) {
                    code = adError.getCode();
                    msg = adError.getMessage();
                }
```

```
                showToast("draw 视频请求失败 msg:" + msg + ",code:" + code);
            }
        });
    }

    private void onDrawAdLoaded(List<? extends ICloudDraw> iCloudDraws) {
        if (iCloudDraws != null && iCloudDraws.size() > 0) {
            mCloudDraw = iCloudDraws.get(0);
            registerListener();
            if (mCloudDraw != null) {
                mCloudDraw.render();
            }
        } else {
            showToast("draw 视频请求成功 但无广告数据返回");
        }
    }

    private void registerListener() {
        if (mCloudDraw != null) {
            mCloudDraw.setDrawInteractionListener(new
ICloudDraw.CloudDrawInteractionListener() {
                @Override
                public void onRenderSuccess() {
                    onDrawAdRenderSuccess();
                }

                @Override
                public void onRenderFail(String msg) {
                    showToast("广告渲染失败 msg:" + msg);
                }

                @Override
                public void onClick(@Nullable View view) {
                    showToast("广告点击");
                }

                @Override
                public void onShow(@Nullable View view) {
                    showToast("广告曝光");
                }
            });
```

```
mCloudDraw.setVideoListener(new CloudVideoListener() {
    @Override
    public void onVideoLoad() {
        showToast("onVideoLoad");
    }

    @Override
    public void onVideoError(CloudError adError) {
        String msg = "";
        if (adError != null) {
            msg = adError.getMessage();
        }
        showToast("onVideoError msg:" + msg);
    }

    @Override
    public void onVideoStart() {
        showToast("onVideoStart");
    }

    @Override
    public void onVideoProgressUpdate(long current, long total) {
        LogUtil.d("current:" + current + ",total:" + total);
    }

    @Override
    public void onVideoPause() {
        showToast("onVideoPause");
    }

    @Override
    public void onVideoContinuePlay() {
        showToast("onVideoContinuePlay");
    }

    @Override
    public void onVideoCompleted() {
        showToast("onVideoCompleted");
    }
});
```

```
            mCloudDraw.setDownloadListener(new CloudAppDownloadListener() {
                @Override
                public void onIdle() {

                }

                @Override
                public void onDownloadActive(String packageName, int progress) {

                }

                @Override
                public void onDownloadPaused(String packageName, int progress) {

                }

                @Override
                public void onDownloadFailed(String packageName, int progress) {

                }

                @Override
                public void onDownloadFinished(String packageName) {

                }

                @Override
                public void onInstalled(String packageName) {

                }
            });
        }
    }

    private void onDrawAdRenderSuccess() {
        View view = mCloudDraw.getDrawView(this);
        if (view != null) {
            ViewGroup.LayoutParams params = new ViewGroup.LayoutParams(
                    ViewGroup.LayoutParams.MATCH_PARENT,
ViewGroup.LayoutParams.MATCH_PARENT);
```

```
            mDrawContainer.addView(view, params);
        } else {
            showToast("draw 视频 view 为空");
        }
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        if (mCloudDraw != null) {
            mCloudDraw.destroy();
        }
    }
}
```

二、主要 API 说明

CloudSdkManager

请查看 CloudSdkManager 类 API 说明文档

方法名	方法介绍
static void loadDraw(DrawRequestParam params, DrawLoadListener listener)	请求 draw 视频广告方法。参数说明：params（广告请求参数：DrawRequestParam ），listener（广告请求状态监听：DrawLoadListener ）

DrawRequestParam

请使用 DrawRequestParam.Bulider 类构建 DrawRequestParam 对象

DrawRequestParam.Bulider

方法名	方法介绍
Builder setAdSenseId(String adSenseId)	（必填项）设置激励视频广告位 ID。参数说明：adSenseId（激励视频广告位 ID）
Builder setAcceptedExpressViewSize(int	（可选配置）设置 draw 视频广告期望展示的尺寸。参数说明：acceptedExpressViewWidth 期望展示

acceptedExpressViewWidth, int acceptedExpressViewHeight)	的宽度 acceptedExpressViewHeight 期望展示的高度（**单位：dp ，sdk 内部默认使用屏幕的宽高**）
DrawRequestParam build()	构建 DrawRequestParam 对象

DrawLoadListener

方法名	方法介绍
void onLoaded(List<? extends ICloudDraw> list)	广告请求成功回调。参数说明：list（请求成功返回的广告数据，单次请求只返回一条广告数据）
void onError(CloudError adError)	请求失败回调。参数说明：adError（错误信息）

ICloudDraw

draw 视频广告数据

方法名	方法介绍
View getDrawView(Context context)	获取 draw 视频 view
void render()	开始渲染
void destroy()	释放资源(**务必保证合适的时机调用此方法以释放内存，destory 的广告不可再用**)
void setVideoListener(CloudVideoListener videoListener)	设置视频播放监听
void setDownloadListener(CloudAppDownloadListener downloadListener)	设置下载监听
void setDrawInteractionListener(CloudDrawInteractionListener listener)	注册 draw 视频广告交互监听

08.信息流自渲染广告

一、接入代码示例

```
//(详细内容请参考压缩包中的代码示例,demo 类：NativeFlowAdActivity)
//广告请求参数，具体参数说明请查看主要 API 说明
/**
  * adSenseId 信息流自渲染广告位 ID（必填）
  */
//--------请求示例
CloudSdkManager.loadNative(new
NativeRequestParam.Builder().setAdSenseId(adSenseId).build(),
                new NativeLoadListener() {
                    @Override
                    public void onLoaded(List<? extends ICloudNative>
cloudNativeAds) {
                        //请求成功，cloudNativeAds 广告数据集合,宿主可以使用
ICloudNative 对象获取广告图片，标题，描述等物料信息
                                                //ICloudNative 类说明请查看主要
API 说明
                    }

                    @Override
                    public void onError(CloudError adError) {
                                                //请求失败
```

```
                showToast(adError.getMessage());
            }
        });
```

二、主要 API 说明

CloudSdkManager

请查看 CloudSdkManager 类 API 说明文档

方法名	方法介绍
static void loadNative(NativeRequestParam params, NativeLoadListener listener)	请求信息流自渲染广告方法。参数说明：params（广告请求参数：NativeRequestParam ），listener（广告请求状态监听：NativeLoadListener ）

NativeRequestParam

请使用 NativeRequestParam .Bulider 类构建 NativeRequestParam 对象

NativeRequestParam.Bulider

方法名	方法介绍
Builder setAdSenseId(String adSenseId)	（**必填项**）设置当前请求广告的 Activity。参数说明：activity（当前请求广告依附的页面）
Builder setPlayPolicy(int playPolicy)	（**可选设置**）设置广告请求播放政策。参数说明：playPolicy（播放政策，**默认 0、WIFI 下自动播放；1、始终自动播放；2、不自动播放。改设置仅对视频广告有效**）
NativeRequestParam build()	构建 NativeRequestParam 对象

NativeLoadListener

方法名	方法介绍
void onLoad(List<? extends	广告请求成功回调。参数说明：

方法名	方法介绍
ICloudNative> cloudNativeAds)	cloudNativeAds（请求成功返回的广告数集合）
void onError(CloudError adError)	请求失败回调。参数说明：adError（错误信息）

ICloudNative

广告数据对象

方法名	方法介绍
int getSdkChannelId()	获取第三方广告源 id
String getTitle()	获取广告标题
String getDescription()	获取广告副标题
String getAppName()	获取广告 appName(仅下载广告有)
String getAppIcon()	获取广告 app Icon(仅下载广告有)
String getAdLogo()	获取广告源 logo（仅自有广告源有）
List<String> getImageList()	获取广告图片物料
View getVideoView(Context context)	获取视频广告播放器 view(须在主线程调用，**请务必在注册监听 registerViewForInteraction 方法之前调用**)
void resume()	宿主 Activity onResume 时调用（**广告展示时必须调用，如果一个页面有多条广告，则所有的广告均需调用，如果不调用会导致广告状态错乱**）
void destroy()	当开发者不在使用该条广告时调用，释放广告资源，且广告不能再次使用（**如果一个页面有多条广告，则所有的广告均需调用**）
boolean isDownload()	是否下载类广告。true 下载类广告

方法名	方法介绍
boolean isDial()	是否拨号类广告。true　拨号类广告
boolean isVideo()	是否视频广告。true 视频广告
int getInteractionType()	获取交互类型。（-1=未知类型；2=浏览器打开；3=落地页打开；4=下载广告；5=拨号类广告。常量定义见：CloudIntertionType 接口）
int getImageMode()	获取图片素材类型（1：小图一文；2=三图一文；3=大图一文；其他未知）
void setDownloadListener(CloudAppDownloadListener downloadListener)	设置下载监听。参数说明：downloadListener(下载状态监听：CloudAppDownloadListener)（**仅下载类广告，才可能需要调用该方法**）
void setVideoListener(CloudVideoListener videoListener)	设置视频播放监听。参数说明：videoListener （视频播放状态监听：CloudVideoListener ）
void registerViewForInteraction(ViewGroup viewGroup, List<View> creativeViews, FrameLayout.LayoutParams logoParam, CloudNativeInteractionListener listener)	注册信息流自渲染广告交互监听。 参数说明：viewGroup （广告展示容器，**必填**）， creativeViews （创意点击 view 列表，可为空）， logoParam （设置 LOGO 位置，可为空）， listener （点击曝光状态回调监听：CloudNativeInteractionListener ）
int getClientAdLogoResId()	获取非自有广告 LOGO 资源 ID，自有广告返回-1

三、注意事项

1、广告展示的父容器必须是 com.qq.e.ads.nativ.widget.NativeAdContainer（FrameLayout 的子类），否则广点通广告展示曝光会有问题。

- 可以通过 CloudSdkManager.createGdtContainer 方法（**如果宿主选择排除广点通 SDK，可能会返回 null**）动态获取 NativeAdContainer 容器
- 或者直接将 NativeAdContainer（FrameLayout 的子类）加到布局中

2、如果是视频广告，getVideoView 方法调用必须在 registerViewForInteraction 方法调用之前，否则广告展示会有问题

四、附录

CloudNativeInteractionListener

```
public interface CloudNativeInteractionListener{

    /**
     * 广告点击
     *
     * @param view 点击 view
     */
    void onClick(@Nullable View view);

    /**
     * 广告有效曝光
     *
     * @param view 曝光 view
     */
    void onShow(@Nullable View view);
}
```

CloudAppDownloadListener

类说明

CloudAppDownloadListener 是下载广告下载状态回调接口（仅部分广告业务支持，具体调用可查看各个广告业务的集成说明）

```
public interface CloudAppDownloadListener {
    /**
     * 未开始下载
     *
     */
    void onIdle();

    /**
     * 下载中回调
     *  @param packageName 应用包名
     * @param progress    下载进度
     */
    void onDownloadActive(String packageName, int progress);

    /**
     * 下载暂停回调
     *  @param packageName 应用包名
     * @param progress    下载进度
     */
    void onDownloadPaused(String packageName, int progress);

    /**
     * 下载失败回调
     *  @param packageName 应用包名
     * @param progress    下载进度
     */
    void onDownloadFailed(String packageName, int progress);

    /**
     * 下载完成回调
     *  @param packageName 应用包名
     *
     */
    void onDownloadFinished(String packageName);

    /**
     * 安装完成回调
     *  @param packageName 应用包名
     *
     */
    void onInstalled(String packageName);
```

```
}
```

CloudVideoListener

类说明

CloudVideoListener 是视频广告播放状态回调接口（仅部分广告业务支持，具体调用可查看各个广告业务的集成说明）

```
public interface CloudVideoListener {

    /**
     * 视频加载完成
     */
    void onVideoLoad();

    /**
     * 视频加载失败
     *
     * @param adError 错误信息
     */
    void onVideoError(CloudError adError);

    /**
     * 视频开始播放
     */
    void onVideoStart();

    /**
     * 视频播放进度回调
     *
     * @param current 当前播放时长
     * @param total   总时长
     */
    void onVideoProgressUpdate(long current, long total);

    /**
     * 视频暂停播放
     */
    void onVideoPause();
```

```
    /**
     * 视频继续播放
     */
    void onVideoContinuePlay();

    /**
     * 视频播放完成
     */
    void onVideoCompleted();
}
```

ICloudNative

```
public interface ICloudNative {

    /**
     * 获取广告标题
     *
     * @return 主标题
     */
    String getTitle();

    /**
     * 获取广告副标题
     *
     * @return 副标题
     */
    String getDescription();

    /**
     * 获取广告 appName
     *
     * @return appName
     */
    String getAppName();

    /**
     * 获取广告 app Icon
     *
     * @return 广告 Icon
     */
```

```
String getAppIcon();

/**
 * 获取广告源 logo
 *
 * @return 广告源 logo
 */
String getAdLogo();

/**
 * 获取广告图片列表
 *
 * @return 广告图片列表
 */
List<String> getImageList();

/**
 * 须在主线程调用
 * 获取视频广告 view
 *
 * @param context context
 * @return 视频广告 view
 */
View getVideoView(Context context);

/**
 * 广告进入前台
 */
void resume();

/**
 * 释放广告资源
 */
void destroy();

/**
 * 是否下载类广告
 *
 * @return true 下载类广告  false 非下载类广告
 */
boolean isDownload();
```

```
/**
 * 是否拨号类广告
 *
 * @return true 拨号类广告  false 非拨号类广告
 */
boolean isDial();

/**
 * 是否视频广告
 *
 * @return true 视频广告  false 非视频广告
 */
boolean isVideo();

/**
 * 获取交互类型
 *
 * @return 交互类型
 */
int getInteractionType();

/**
 * 获取图片素材类型
 *
 * @return 图片素材类型
 */
int getImageMode();

/**
 * 获取实时价格
 *
 * @return 实时价格
 */
String getECPMLevel();

/**
 * 设置下载监听
 *
 * @param downloadListener
 */
```

```
    void setDownloadListener(CloudAppDownloadListener downloadListener);

    /**
     * 设置视频播放监听
     *
     * @param videoListener 视频播放监听
     */
    void setVideoListener(CloudVideoListener videoListener);

    /**
     * 注册激励视频广告交互监听
     *
     * @param viewGroup     广告展示根容器
     * @param creativeViews 创意点击 view 列表
     * @param logoParam     设置 LOGO 位置，可传空
     * @param listener      监听
     */
    void registerViewForInteraction(ViewGroup viewGroup, List<View>
creativeViews,
                                    FrameLayout.LayoutParams logoParam,
CloudNativeInteractionListener listener);

    /**
     * 获取客户端广告 LOGO 资源 ID
     * 其他返回-1
     *
     * @return 资源 ID
     */
    int getClientAdLogoResId();

    interface CloudNativeInteractionListener extends CloudInteractionListener {
    }
}
```

09.信息流模板广告

一、代码调用

接入代码示例

1、非信息流场景模板广告请求

```
NativeExpressSetting setting =
                new NativeExpressSetting.Builder()
                        .setAdSenseId(adSenseId)
                        .setExpressViewAcceptedSize(mAcceptedWidth,
mAcceptedHeight)
                        .setShowClose(true)
                        .setTitleTextColor(mTitleColor)
                        .setTitleTextSize(18)
                        .setSubTitleTextSize(12)
                        .setSubTitleTextColor(mSubTitleColor)
                        .setFlipTextSize(15)
                        .setFlipTextColor(mFlipTitleColor)
                        .setAutoRequest(true)
                        .build();

CloudSdkManager.loadNativeExpress(activity, adSetting, new
NativeExpressListener() {
                @Override
                public void onLoaded(View view) {
                        showToast("onAdLoaded");
                if (view == null) {
                return;
                }
                ViewGroup.MarginLayoutParams params = new
ViewGroup.MarginLayoutParams(
```

```
                        ViewGroup.LayoutParams.MATCH_PARENT,
ViewGroup.LayoutParams.WRAP_CONTENT);
                mExpressAdContainer.addView(view, params);
                }

                @Override
                public void onShow() {
                        showToast("onAdShow");
                }

                @Override
                public void onError(CloudError adError) {
                        if (adError != null) {
                                showToast("onAdError code:" + adError.getCode() +
" msg:" + adError.getMessage());
                        } else {
                                showToast("onAdError");
                        }
                }

                @Override
                public void onClick(boolean isDownload) {
                        showToast("onAdClick isDownload:" + isDownload);
                }

                @Override
                public void onClose() {
                        showToast("onAdClose");
                }
});
```

2、信息流场景模板广告请求

1）模板广告配置发生变化时（如夜间模式和无图模式切换），需重新调用模板广告全局配置接口更新配置；

2）广告内部监听了 activity 生命周期，当 activity 再次进入前台时刷新的配置即可生效（包含已请求展示的广告样式）。

```
NativeExpressSetting setting =
                new NativeExpressSetting.Builder()
```

```
                        .setShowClose(true)
                        .setTitleTextColor(mTitleColor)
                        .setTitleTextStyle(NativeExpressSetting.TEXT_BOLD)
                        .setTitleTextSize(18)
                        .setSubTitleTextSize(12)
                        .setSubTitleTextColor(mSubTitleColor)
                        .setDarkMode(mDarkMode)
                        .setNoPicMode(mNoPicMode)
                        .build();

// 信息流场景模板广告全局配置
CloudSdkSetting.configNativeExpress(adSetting);

CloudSdkManager.loadNativeExpress(
                        activity,
                        new NativeExpressSetting.Builder()
                                .setAdSenseId(adSenseId)
                                .setExpressViewAcceptedSize(mAcceptedWidth,
mAcceptedHeight)
                                .build(),
                        new NativeExpressListener() {
                                @Override
                                public void onLoaded(View view) {
                                        showToast("onAdLoaded");
                                if (view == null) {
                                return;
                                }
                                ViewGroup.MarginLayoutParams params = new
ViewGroup.MarginLayoutParams(
                                        ViewGroup.LayoutParams.MATCH_PARENT,
ViewGroup.LayoutParams.WRAP_CONTENT);
                                mExpressAdContainer.addView(view, params);
                                }

                                @Override
                                public void onShow() {
                                        showToast("onAdShow");
                                }

                                @Override
                                public void onError(CloudError adError) {
```

```
                                        if (adError != null) {
                                                showToast("onAdError code:" +
adError.getCode() + " msg:" + adError.getMessage());
                                        } else {
                                                showToast("onAdError");
                                        }
                                }

                                @Override
                                public void onClick(boolean isDownload) {
                                        showToast("onAdClick isDownload:" +
isDownload);
                                }

                                @Override
                                public void onClose() {
                                        showToast("onAdClose");
                                }
});
```

二、主要 API 说明

CloudSdkManager

请查看 CloudSdkManager 类 API 说明文档

方法名	方法介绍
static void loadNativeExpress(Activity activity, NativeExpressSetting adSetting, NativeExpressListener listener)	请求信息流模板广告方法。参数说明：adSetting（广告请求参数：NativeExpressSetting），listener（广告请求状态监听：NativeExpressListener ）

NativeExpressSetting

请使用 NativeExpressSetting__.Bulider 类构建 NativeExpressSetting 对象

NativeExpressSetting.Bulider

方法名	方法介绍
Builder setAdSenseId(String adSenseId)	（必填项）设置信息流模板广告位 ID。参数说明：adSenseId（信息流模板广告位 ID）
Builder setExpressViewAcceptedSize(int width, int height)	（可选设置）设置期望的模板 view 展示尺寸。参数说明：width 期望展示的宽度 height 期望展示的高度（宽高单位均为 dp，设置的宽度需为模板 view 展示真实的可用宽度，减掉 padding 和 margin，高度建议设置自适应高度，宽度和高度不足均有可能导致模板广告展示异常）
Builder setTitleTextColor(String titleTextColor)	（可选设置）设置标题文字颜色
Builder setTitleTextSize(int titleTextSize)	（可选设置）设置标题文字大小 单位：sp
Builder setTitleTextStyle(int titleTextStyle)	（可选设置）设置标题字体样式 。参数说明：titleTextStyle 标题字体样式 （正常字体：INativeExpressSetting.TEXT_NORMAL 加粗字体：INativeExpressSetting.TEXT_BOLD），仅适用信息流场景模板广告
Builder setSubTitleTextColor(String subTitleTextColor)	（可选设置）设置副标题文字颜色
Builder setSubTitleTextSize(int subTitleTextSize)	（可选设置）设置副标题文字大小 单位：sp
Builder setFlipTextColor(String flipTextColor)	（可选设置）设置单行文字链模板文字颜色， 仅适用非信息流场景的模板广告
Builder setFlipTextSize(int flipTextSize)	（可选设置）设置单行文字链模板文字大小 单位：sp ，仅适用非信息流场景的模板广告
Builder setAutoRequest(boolean autoRequest)	（可选设置）设置是否自动请求广告 true 自动请求广告，仅适用非信息流场景的模板广告
Builder setShowClose(boolean showClose)	（可选设置）设置是否显示关闭按钮 true 显示关闭按钮
Builder setDarkMode(boolean darkMode)	（可选设置）设置是否为夜间模式 true 夜间模式，仅适用信息流场景模板广告

Builder setNoPicMode(boolean noPicMode)	（可选设置）设置是否是否无图模式 true 无图模式 ，仅适用信息流场景模板广告
NativeExpressSetting build()	构建 NativeExpressSetting 对象

NativeExpressListener

方法名	方法介绍
void onLoaded(View view)	广告请求成功回调
void onError(CloudError adError)	请求失败回调 参数说明：adError（错误信息）
void onShow()	广告曝光回调
void onClick()	广告点击回调
void onClose()	广告关闭回调

10. 原生信息流广告

一、代码调用

接入代码示例

```
//(详细内容请参考压缩包中的代码示例,demo 类：NativePageActivity)

//广告请求参数，具体参数说明参考 API 说明
NativePageRequestParam param = new NativePageRequestParam.Builder()
```

```
        .setAdSenseId(slotId)
        .setUiStyle(Integer.parseInt(uiStyle))
        .build();

//请求原生信息流广告
CloudSdkManager.loadNativePage(param, new NativePageLoadListener () {
        @Override
        public void onLoaded(boolean isDownload) {
                showToast("onAdLoaded isDownload:" + isDownload);
        }

        @Override
        public void onError(CloudError adError) {
                if (adError != null) {
                showToast("onAdError code:" + adError.getCode() + ",msg:" +
adError.getMessage());
                } else {
                showToast("onAdError");
                }
        }

        @Override
        public void onShow() {
                showToast("onAdShow");
        }

        @Override
        public void onClick() {
                showToast("onAdClick");
        }

        @Override
        public void onClose() {
                showToast("onAdClose");
        }
});
```

二、主要 API 说明

CloudSdkManager

请查看 CloudSdkManager 类 API 说明文档

方法名	方法介绍
static void loadNativePage(NativePageRequestParam params, NativePageLoadListener listener)	请求原生信息流广告方法。参数说明：params（广告请求参数：NativePageRequestParam），listener（广告请求状态监听：NativePageLoadListener）

NativePageRequestParam

请使用 NativePageRequestParam.Bulider 类构建 NativePageRequestParam 对象

NativePageRequestParam.Bulider

方法名	方法介绍
Builder setAdSenseId(String adSenseId)	（必填项）设置原生信息流广告位 ID。参数说明：adSenseId（原生信息流广告位 ID）
Builder setUiStyle(int uiStyle)	（可选配置）原生信息流广告样式　　普通样式：INativePageParam.UI_STYLE_NORMAL　　高斯模糊样式：　INativePageParam.UI_STYLE_BLUR　　默认高斯模糊样式
NativePageRequestParam build()	构建 NativePageRequestParam 对象

NativePageLoadListener

方法名	方法介绍
void onLoaded()	请求成功回调
void onError(CloudError adError)	请求失败回调。参数说明：adError（错误信息）
void onShow()	广告曝光回调
void onClick()	广告点击回调
void onClose()	广告关闭回调

11. 互动式广告

一、代码调用

注：新版本 SDK 内部不再处理广告关闭操作，关闭按钮以及相关操作由宿主 App 自己实现；

接入代码示例

```
//(详细内容请参考压缩包中的代码示例,demo 类：InteractionAdActivity)

//广告请求参数，具体参数说明参考 API 说明
InteractionRequestParam param = new InteractionRequestParam.Builder()
        .setAdSenseId(adSenseId)
        .setContainer(mContainer)
        .setAcceptedViewSize(acceptedWidth, acceptedHeight)
        .setFlipInterval(interval)
        .build();
//请求互动式广告
CloudSdkManager.loadInteraction(param, new InterstitialLoadListener() {
        @Override
        public void onLoaded() {
                showToast("onLoaded");
        }

        @Override
        public void onError(CloudError adError) {
                if (adError != null) {
```

```
                        showToast("onAdError code:" + adError.getCode() + ",msg:"
+ adError.getMessage());
                } else {
                        showToast("onAdError");
                }
        }

        @Override
        public void onShow() {
                showToast("onShow");
        }

        @Override
        public void onClick() {
                showToast("onClick");
        }
});
```

二、主要 API 说明

CloudSdkManager

请查看 CloudSdkManager 类 API 说明文档

方法名	方法介绍
static void loadInteraction(InteractionRequestParam params, InteractionLoadListener listener)	请求互动式广告方法。参数说明：params（广告请求参数：InteractionRequestParam ），listener（广告请求状态监听：InteractionLoadListener ）

InteractionRequestParam

请使用 InteractionRequestParam .Bulider 类构建 InteractionRequestParam 对象

InteractionRequestParam.Bulider

方法名	方法介绍

Builder setAdSenseId(String adSenseId)	（**必填项**）设置互动式广告位 ID。参数说明：adSenseId（互动式广告位 ID）
Builder setContainer(ViewGroup container)	（**必填项**）互动式广告展示容器（**重复请求时，请务必保证先 remove 容器中已添加的互动式广告 view**）
Builder setAcceptedViewSize(int acceptedViewWidth, int acceptedViewHeight)	（**可选设置**）设置互动式广告 view 展示宽高，**单位：dp（建议宽高比 1：1）**
Builder setFlipInterval(long flipInterval)	（**可选设置**）设置广告轮播间隔 **单位：ms**
InteractionRequestParam.build()	构建 InteractionRequestParam 对象

InteractionLoadListener

方法名	方法介绍
void onLoaded()	广告请求成功回调
void onError(CloudError adError)	请求失败回调　参数说明：adError（错误信息）
void onShow()	广告曝光
void onClick()	广告点击回调
void onClose()	广告关闭回调

12. SDK 错误码（外拓）

- 附录

日志过滤 TAG：LoadManager，若无日志输出，请参考 debug 开关设置，开启日志

错误码	说明
2345000	服务端广告源返回有误，

	当前 sdk 版本该广告类型三方 SDK 未接入，不支持请求
2345001	初始化出错
2345002	媒体 id 被封
2345003	广告位 id 被封
2345100	媒体 id 不合法（未设置媒体 id 媒体 id 未收录 媒体 id 未生效）
2345101	广告位 id 不合法（未设置广告位 id 广告位 id 未收录 广告位 id 未生效）
2345102	媒体 id 与应用包名不匹配
2345103	媒体 id 和应用签名不匹配
2345104	媒体 id 和广告位 id 不匹配
2345105	广告位 id 和请求的广告类型不匹配
2345106	广告请求尺寸设置不对
2345107	广告请求数量不对
2345201	网络连接失败
2345202	请求超时
2345203	无广告填充
2345204	广告请求频次过高
2345205	广告位请求量超限
2345206	SDK 版本过低不返回广告
2345300	物料缓存失败

2345301	图片加载失败
2345302	视频缓存失败
2345303	模板 id 未匹配到对应模板
2345304	渲染失败

附录

- 百度 SDK 错误码
- 广点通 SDK 错误码
- 穿山甲 SDK 错误码
- SigMobSDK 错误码
- 快手 SDK 错误码

code	说明
40001	没有网络
40002	数据解析失败
40003	广告数据为空
40004	缓存视频资源失
100001	参数有误
100002	服务器错误
100003	不允许的操作
100004	服务不可用
310001	appId 未注册
310002	appId无效
310003	appId 已封禁
310004	packageName 与注册的 packageName 不一致
310005	操作系统与注册的不一致

320002	appId 对应账号无效
320003	appId 对应账号已封禁
330001	posId 未注册
330002	posId无效
330003	posId 已封禁
330004	posid 与注册的 appId 信息不一致

12. debug

日志开启命令（开启后重启应用）

```
adb shell setprop log.tag.Business V

adb shell setprop log.tag.CloudAdSdk V

adb shell setprop log.tag.Host V
```

过滤 Tag Business|biz2345-ad|HOST