

NGAdSDK集成文档

版本发布记录

1. SDK 集成与工程配置

1.1 接入须知

1.2 SDK版本说明列表

1.3 开发环境建议

1.4 aar集成

1.5 快速集成

1.6 支持架构

1.7 详细集成

1.7.1 添加权限

1.7.2 AndroidManifest及资源文件配置

1.7.3 添加混淆

2. 隐私合规信息&初始化说明

2.1 隐私合规信息说明

2.2 SDK初始化

3. 开屏广告

3.1 简介

3.2 注意事项

3.3 接入开屏广告

3.4 接口说明

3.5 高级功能

3.5.1 与网游联运SDK集成时机

3.5.2 首次加载超时解决方案

4. 激励视频广告

4.1 简介

4.2 注意事项

4.3 接入激励视频广告

4.4 接口说明

- 4.5 示例代码
- 5. 插全屏广告
 - 5.1. 简介
 - 5.2 注意事项
 - 5.3 接入插全屏广告
 - 5.4 接口说明
 - 5.6 示例代码
- 6. Banner广告
 - 6.1 简介
 - 6.2 注意事项
 - 6.3 接入Banner广告
 - 6.4 接口说明
 - 6.5 示例代码
- 7. 融合SDK版本接入注意事项
 - 7.1 常见问题
- 8. SDK错误码问题定位总结思路及错误码解析
 - 8.1 广告SDK高频错误码解析
 - 8.2 ADN广告常见错误码
 - 8.2.1 GDT
 - 8.2.2 KS
 - 8.3 问题反馈模版
 - 8.4 抓取排查日志
 - 打开日志方式一
 - 打开日志方式二
 - 抓取日志

版本发布记录

文档版本	修订日期	修订内容
3.2.1.0	2025-05-29	1. 升级融合SDK版本。

3.2.0.1	2024-11-26	<ol style="list-style-type: none"> 1. 接入优化：简化接入方式，提升接入体验。 2. 工具优化：增加露出SDK版本方式，加固游戏支持SDK版本读取。
3.1.0.0	2024-09-06	<ol style="list-style-type: none"> 1. 功能增加：增加对快手ADN的支持 2. 接入优化：优化日志，增加本地日志排查手段
3.0.0.0	2024-07-23	<ol style="list-style-type: none"> 1. 接口变更：NGAdSdk.init(Activity activity, ...) 2. 功能增加：激励视频，增加无网络引导弹窗，默认开启 3. 优化缓存机制。 4. Demo 完善功能注释引导。
1.0.0.2	2024-07-03	<ol style="list-style-type: none"> 1. 修复Bug 2. 删除useMediation 接口 3. Demo 增加隐私示例 4. Demo 增加开屏与联运SDK初始化时机流程 5. 增加无网络过滤处理，详情查看错误码
1.0.0.0	2024-06-21	初发布

1. SDK 集成与工程配置

1.1 接入须知

- ① **接入注意事项**：详情查看【融合SDK版本接入注意事项】。
- ② **错误码问题定位及错误码解析**：详情查看【SDK错误码问题定位总结思路及错误码解析】。

1.2 SDK版本说明列表

ADN	SDK版本	adapter版本
穿山甲 SDK	v6.2.1.7	/
GDT SDK	4.570.1440	4.570.1440.1

快手 SDK	3.3.63	3.3.63.2
--------	--------	----------

1.3 开发环境建议

Gradle 版本无限制，游戏根据自身情况酌情配置，但建议低于 8.0，

推荐 gradle-7.3.3-bin & AGP 7.2.2。

推荐 targetSdkVersion 26 以上，游戏根据自身情况酌情配置。

推荐 JDK 版本 11，并配置 JavaCompatibility 1.8。

▼ Plain Text

```

1 compileOptions {
2     sourceCompatibility JavaVersion.VERSION_1_8
3     targetCompatibility JavaVersion.VERSION_1_8
4 }
```

1.4 aar集成

下载SDK的压缩包，解压后NGAD_SDKx.x.x.x的文件夹里面会有以下内容：

目录/文件	说明
aar	广告相关的aar文件
docs	广告集成文档
NGAdSDKDemo.zip	广告Demo工程

备注：融合SDK支持接入多家ADN，开发者可以按需接入对应的ADN的SDK和adapter

```
1 // 提供一种aar目录声明管理方式：将需要的ADN、adapter放入对应子目录就行
2 dependencies {
3     implementation fileTree(include: ['*.jar'], dir: 'libs')
4     // 增加 libs 目录下的aar
5     implementation fileTree(dir: 'libs', include: ['*.aar', '*.jar'], exclude: [])
6     // 增加 libs/adapter 目录下的aar
7     implementation fileTree(dir: 'libs/adapter', include: ['*.aar', '*.jar'], exclude: [])
8     // 增加 libs/adn 目录下的aar
9     implementation fileTree(dir: 'libs/adn', include: ['*.aar', '*.jar'], exclude: [])
10
11 }
```

1.5 快速集成

提供 `ng-ad-sdk-quick-${版本号}.aar` 的快速集成aar，做了以下工作：

1. 权限声明
2. AndroidManifest组件声明
3. 资源文件配置
4. 混淆

以上会集成所有ADN所需的声明，快速集成方式CP可以引入，不用关心细节。

PS：如果需要定制集成，请移步最下面的[详细集成](#)，并排除对 `ng-ad-sdk-quick-${版本号}.aar` 的依赖

1.6 支持架构

SDK默认支持armeabi-v7a,arm64-v8a两种架构，如果有其他架构（**armeabi架构**）需求，请联系技术支持同学；

您可以在应用中的 `build.gradle` 中使用 `abiFilters` 选择支持的架构。如下所示：

```

1 ndk { // 设置支持的 SO 库构架，注意这里要根据你的实际情况来设置
2     abiFilters armeabi-v7a , arm64-v8a
3 }

```

1.7 详细集成

!!：推荐使用快速集成的方式，使用了快速集成的方式，《详细集成》这块内容可以忽略

!!：各个ADN要求都接入，能最大化广告收益和填充率

1.7.1 添加权限

1. 融合SDK所需权限

```

1 <!-- 所需权限 -->
2 <uses-permission android:name="android.permission.INTERNET" />
3 <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
4 <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
5 <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
6
7 <uses-permission android:name="android.permission.REQUEST_INSTALL_PACKAGE" />
8 <uses-permission android:name="android.permission.READ_PHONE_STATE" />

```

1. 申请以下权限用于防作弊功能以及有助于广告平台投放广告

```

1 <!--可选权限，申请后用于防作弊功能以及有助于广告平台投放广告-->
2 <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
3 <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
4
5 <!--建议添加“query_all_package”权限，SDK将通过此权限在Android R系统上判定广告对应的应用是否在用户的app上安装，避免投放错误的广告，以此提高用户的广告体验。若添加此权限，需要在您的用户隐私文档中声明！ -->
6 <uses-permission android:name="android.permission.QUERY_ALL_PACKAGES" />

```

1. 聚合三方ADN可选权限

```
1  <!-- 可选权限 -->
2  <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
3
4  <!--suppress DeprecatedClassUsageInspection -->
5  <uses-permission android:name="android.permission.GET_TASKS" />
6  <uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
7
8  <!-- 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 -->
9  <uses-permission android:name="android.permission.WAKE_LOCK" />
```

1.7.2 AndroidManifest及资源文件配置

CSJ广告

- [AndroidManifest配置](#)

```
1  <!-- 穿山甲 start===== -->
2  <provider
3      android:name="com.bytedance.sdk.openadsdk.TTFileProvider"
4      android:authorities="${applicationId}.TTFileProvider"
5      android:exported="false"
6      android:grantUriPermissions="true">
7      <meta-data
8          android:name="android.support.FILE_PROVIDER_PATHS"
9          android:resource="@xml/pangle_file_paths" />
10 </provider>
11
12 <provider
13     android:name="com.bytedance.sdk.openadsdk.multipro.TTMultiProvider"
14     android:authorities="${applicationId}.TTMultiProvider"
15     android:exported="false" />
16 <!-- 穿山甲 end===== -->
```

- 在res/xml目录下，添加文件pangle_file_paths.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <paths>
3      <external-path name="tt_external_root" path="." />
4      <external-path name="tt_external_download" path="Download" />
5      <external-files-path name="tt_external_files_download" path="Download"
6  />
7      <files-path name="tt_internal_file_download" path="Download" />
8      <cache-path name="tt_internal_cache_download" path="Download" />
9  </paths>
```

gdt广告

- [AndroidManifest配置](#)


```

1  <!-- GDT start===== -->
2  <!-- targetSDKVersion >= 24时才需要添加这个provider。provider的authorities属性
   的值为${applicationId}.fileprovider，请开发者根据自己的${applicationId}来设置这
   个值，例如本例中applicationId为"com.qq.e.union.demo"。 -->
3  <provider
4      android:name="com.qq.e.comm.GDTFileProvider"
5      android:authorities="${applicationId}.gdt.fileprovider"
6      android:exported="false"
7      android:grantUriPermissions="true">
8      <meta-data
9          android:name="android.support.FILE_PROVIDER_PATHS"
10         android:resource="@xml/gdt_file_path" />
11 </provider>
12
13 <activity
14     android:name="com.qq.e.ads.PortraitADActivity"
15     android:configChanges="keyboard|keyboardHidden|orientation|screenSize"
16     android:screenOrientation="portrait" />
17 <activity
18     android:name="com.qq.e.ads.LandscapeADActivity"
19     android:configChanges="keyboard|keyboardHidden|orientation|screenSize"
20     android:screenOrientation="landscape"
21     tools:replace="android:screenOrientation" />
22
23 <!-- 声明SDK所需要的组件 -->
24 <service
25     android:name="com.qq.e.comm.DownloadService"
26     android:exported="false" />
27 <!-- 请开发者注意字母的大小写，ADActivity，而不是AdActivity -->
28
29 <activity
30     android:name="com.qq.e.ads.ADActivity"
31     android:configChanges="keyboard|keyboardHidden|orientation|screenSiz
   e" />
32 <!-- GDT end===== -->

```

- 在res/xml目录下，添加文件 gdt_file_path.xml

```
1  <paths>
2      <!-- 这个下载路径也不可以修改，必须为com_qq_e_download -->
3      <external-cache-path
4          name="gdt_sdk_download_path1"
5          path="com_qq_e_download" />
6      <cache-path
7          name="gdt_sdk_download_path2"
8          path="com_qq_e_download" />
9  </paths>
```

1.7.3 添加混淆

- 如果您需要使用proguard混淆代码，需确保不要混淆SDK的代码。请在proguard-rules.pro文件(或其他混淆文件)尾部添加如下配置。

```

1  //聚合混淆
2  -keep class bykvm*.**
3  -keep class com.bytedance.msdk.adapter.**{ public *; }
4  -keep class com.bytedance.msdk.api.** {
5      public *;
6  }
7  -keep class com.bytedance.msdk.base.TTBaseAd{*;}
8  -keep class com.bytedance.msdk.adapter.TTAbsAdLoaderAdapter{
9      public *;
10     protected <fields>;
11 }
12
13 # baidu sdk 不接入baidu sdk可以不引入
14 -ignorewarnings
15 -dontwarn com.baidu.mobads.sdk.api.**
16 -keepclassmembers class * extends android.app.Activity {
17     public void *(android.view.View);
18 }
19
20 -keepclassmembers enum * {
21     public static **[] values();
22     public static ** valueOf(java.lang.String);
23 }
24
25 -keep class com.baidu.mobads.** { *; }
26 -keep class com.style.widget.** {*;}
27 -keep class com.component.** {*;}
28 -keep class com.baidu.ad.magic.flute.** {*;}
29 -keep class com.baidu.mobstat.forbes.** {*;}
30
31 #ks 不接入ks sdk可以不引入
32 -keep class org.chromium.** {*;}
33 -keep class org.chromium.** { *; }
34 -keep class aegon.chrome.** { *; }
35 -keep class com.kwai.**{ *; }
36 -dontwarn com.kwai.**
37 -dontwarn com.kwad.**
38 -dontwarn com.ksad.**
39 -dontwarn aegon.chrome.**
40
41 #oaid 不同的版本混淆代码不太一致，你注意你接入的oaid版本 ， 不接入oaid可以不添加
42 -dontwarn com.bun.**
43 -keep class com.bun.** {*;}
44 -keep class a.**{*;}
45 -keep class XI.CA.XI.**{*;}

```

```
46 -keep class XI.K0.XI.**{*;}
47 -keep class XI.XI.K0.**{*;}
48 -keep class XI.vs.K0.**{*;}
49 -keep class XI.xo.XI.XI.**{*;}
50 -keep class com.asus.msa.SupplementaryDID.**{*;}
51 -keep class com.asus.msa.sdid.**{*;}
52 -keep class com.huawei.hms.ads.identifier.**{*;}
53 -keep class com.samsung.android.deviceidservice.**{*;}
54 -keep class com.zui.opendeviceidlibrary.**{*;}
55 -keep class org.json.**{*;}
56 -keep public class com.netease.nis.sdkwrapper.Utils {public <methods>;}
```

2. 隐私合规信息&初始化说明

2.1 隐私合规信息说明

SDK隐私政策：

接入事例

```
1  NGAdSdk.init(this, new NGAdConfig.Builder()
2      .setAppId("xxxx")
3      .setCustomController(getNGCustomController()) // 隐私合规设置
4      .build(), new Callback(){...});
5
6  /**
7   * 函数返回值表示隐私开关开启状态, 未重写函数使用默认值
8   */
9  private static NGCustomController getNGCustomController() {
10     return new NGCustomController() {
11
12         @Override
13         public boolean isCanUseLocation() {
14             return super.isCanUseLocation();
15         }
16
17         @Override
18         public boolean alist() {
19             return super.alist();
20         }
21
22         @Override
23         public boolean isCanUsePhoneState() {
24             return super.isCanUsePhoneState();
25         }
26
27         @Nullable
28         @Override
29         public String getDevImei() {
30             return super.getDevImei();
31         }
32
33         @Override
34         public boolean isCanUseWifiState() {
35             return super.isCanUseWifiState();
36         }
37
38         @Nullable
39         @Override
40         public String getMacAddress() {
41             return super.getMacAddress();
42         }
43
44         @Override
45         public boolean isCanUseWriteExternal() {
```

```

46         return super.isCanUseWriteExternal();
47     }
48
49     @Nullable
50     @Override
51     public String getDev0aid() {
52         return super.getDev0aid();
53     }
54
55     @Override
56     public boolean isCanUseAndroidId() {
57         return super.isCanUseAndroidId();
58     }
59
60     @Nullable
61     @Override
62     public String getAndroidId() {
63         return super.getAndroidId();
64     }
65
66     @Override
67     public boolean isCanUsePermissionRecordAudio() {
68         return super.isCanUsePermissionRecordAudio();
69     }
70 };
71 }
72 }

```

NGCustomController说明

隐私设置类，通过NGAdConfig的customController(NGCustomControllercontroller)来设置

方法名	说明
boolean isCanUseLocation()	是否允许SDK主动使用地理位置信息
LocationProvider getLocation()	可传入地理位置信息
boolean alist()	是否允许sdk上报手机app安装列表
boolean isCanUsePhoneState()	是否允许SDK主动使用手机硬件参数
String getDevlmei()	当isCanUsePhoneState=false时，可传入IME信息
boolean isCanUseWifiState()	是否允许SDK主动使用ACCESS_WIFI_STATE权限

String getAddress()	当isCanUseWifiState=false时，可传入Mac地址信息
boolean isCanUseWriteExternal()	是否允许SDK主动使用WRITE_EXTERNAL_STORAGE权限
String getDevOaid()	开发者可以传入OAID
boolean isCanUseAndroidId()	是否能获取android ID
String getAndroidId()	开发者可以传入android ID
boolean isCanUsePermissionRecordAudio()	是否允许SDK在申明和授权了的情况下使用录音权限
MediationPrivacyConfig getMediationPrivacyConfig()	设置聚合隐私控制开关

MediationPrivacyConfig说明

聚合设置隐私类

方法名	说明
List getCustomAppList()	开发者可以传入app列表
List getCustomDevImeis()	开发者可以传入多个imei值
boolean isCanUseOaid()	是否可以使用oaid
boolean isLimitPersonalAds()	是否限制个性化推荐接口
boolean isProgrammaticRecommend()	是否启用程序化广告推荐 true启用 false不启用

2.2 SDK初始化

注意事项

- ① 默认仅支持初始化SDK一次，多次初始化SDK以第一次初始化为准。
- ② NGAdSdk中IMediationManager getMediationManager() 需确保SDK初始化完成后调用，否则可能为空。

- ③ SDK初始化成功回调转到子线程，请不要直接在回调里做UI操作。
- ④ appId为必填内容，若appId是通过服务端下发的，那么在初始化前需要做不为空的判断。
- ⑤ 开发者需确保在success回调之后再去请求广告。
- ⑥ SDK初始化API：该API必须在主线程中调用，SDK会将初始化操作放在子线程执行。

接入事例

```
1  public class DemoApplication extends Application {
2      public static String PROCESS_NAME_XXXX = "process_name_XXXX";
3
4      @Override
5      public void onCreate() {
6          super.onCreate();
7          //强烈建议在应用对应的Application#onCreate()方法中调用，避免出现content为null
            的异常
8          NGAdSdk.init(context, buildConfig(context), new Callback(){...});
9
10         //setp1.2: 启动SDK，必须在NGAdSdk.init回调success之后再调用
11         NGAdSdk.start(new NGAdSdk.Callback() {
12             @Override
13             public void success() {
14
15                 Log.i(TAG, "success: " + NGAdSdk.isSdkReady());
16                 startActivity(context);
17             }
18
19             @Override
20             public void fail(int code, String msg) {
21                 Log.i(TAG, "fail:  code = " + code + " msg = " + msg);
22             }
23
24             private static NGAdConfig buildConfig(Context context) {
25                 return new NGAdConfig.Builder()
26                     .appId("xxxxxx")//应用ID
27                     // .supportMultiProcess(true)//开启多进程
28                     .build();
29             }
30
31         }
32     }
33 }
```


NGAdSdk.Callback说明

方法名	说明
void success()	SDK初始化成功回调
void fail(int errorcode, String errorMsg)	SDK初始化失败回调

• SDK初始化失败回调错误码说明

value	说明
4201	<ol style="list-style-type: none">需要依赖 appcompat-v7 库；<ol style="list-style-type: none">implementation "com.android.support:appcompat-v7:28.0.0"检查下项目的CPU架构，架构不匹配时会提示该错误，SDK默认支持armeabi-v7a,arm64-v8a两种架构；若为androidx项目，请检查androidx及gradle相关配置是否正确；<ol style="list-style-type: none">在 gradle.properties文件添加 android.enableJetifier=true如以上排查没有问题，需提供对应抓包信息。

NGAdConfig.Builder说明

初始化SDK配置类

方法名	说明
appld(String var1)	必选参数，设置应用ID
appName(String var1)	可选参数，设置应用名称
paid(boolean var1)	是否为计费用户
keywords(String var1)	用户画像的关键词列表
titleBarTheme(int var1)	落地页主题
allowShowNotify(boolean var1)	是否允许SDK弹出通知
debug(boolean var1)	debug模式

directDownloadNetworkType(int... var1)	允许直接下载的网络状态集合
useTextureView(boolean var1)	是否使用TextureView播放视频
supportMultiProcess(boolean var1)	支持多进程，注意：开启多进程开关时需要 对ADN进行设置 ，否则广告展示异常、影响收益。CSJ、gdt 无需额外设置
setAgeGroup(int var1)	设置年龄段，默认成年
customController(TTCustomController var1)	设置隐私信息控制开关
themeStatus(int var1)	主体模式设置，0是正常模式；1是夜间模式；

3. 开屏广告

3.1 简介

SDK为接入方提供了开屏广告，开屏广告建议为用户在进入App时展示的全屏广告。

开屏广告为一个View，为避免不同广告服务器对尺寸的要求，**宽高默认为match_parent（即全屏）**，同时不要在开屏广告上覆盖其他View（比如空白条、Logo之类，会影响广告尺寸大小的计算），否则会影响计费。

3.2 注意事项

- ① 图片尺寸传入与展示区域大小设置需保持一致，避免素材变形；
- ② 需要确保在SDK初始化成功后再进行广告请求，否则可能导致广告请求加载失败；
- ③ 聚合SDK是通过**广告位ID**发起广告请求的，切记不要使用混淆。
- ④ 在广告接入前需要明确**各ADN对应 NGAd 广告样式**情况，以确保正确完成广告接入，避免由于广告类型不匹配导致接入报错等情况的发生；
- ⑤ 由于各广告平台对于包名校验规则不同，需确保在NGAd平台填写的包名符合各ADN平台规范，避免由于包名校验不匹配导致的无广告返回情况的产生。
- ⑥ 与网游联运SDK的集成时机，请阅读**高级功能-与网游联运SDK集成时机**处理，否则也会影响计算

3.3 接入开屏广告

一、创建NGAdBase对象

```
1  NGAdBase adBaseLoader = NGAdapterManagerHolder.get().createAdNative(this);
```

二、创建广告请求AdPlacement

```
1  AdPlacement adPlacement = new AdPlacement.Builder()  
2      .setCodeId("聚合广告位ID")  
3      .setImageAcceptedSize(widthPx, heightPx)  
4      .build();
```

三、请求开屏广告

```
1  adBaseLoader.loadSplashAd(adPlacement, new NGAdBase.SplashAdListener() {  
2      @Override  
3      public void onSplashRenderSuccess(NGSplashAd ad) {  
4          /** 渲染成功后, 展示广告 */  
5          showSplashAd(ad);  
6      }  
7  
8      @Override  
9      public void onSplashLoadSuccess(NGSplashAd ad) {  
10     }  
11  
12     @Override  
13     public void onSplashLoadFail(NGAdError ngAdError) {  
14     }  
15  
16     @Override  
17     public void onSplashRenderFail(NGSplashAd ad, NGAdError ngAdError) {  
18     }  
19     }  
20 }, 3500);
```

四、展示开屏广告

```
1 public void showSplashAd(NGSplashAd ad) {
2     ad.setSplashAdListener(new NGSplashAd.SplashAdListener() {
3         @Override
4         public void onSplashAdShow(NGSplashAd ad) {
5         }
6
7         @Override
8         public void onSplashAdClick(NGSplashAd ad) {
9         }
10
11        @Override
12        public void onSplashAdClose(NGSplashAd ad, int closeType) {
13            finish();
14        }
15    });
16    View splashView = ad.getSplashView();
17    UIUtil.removeFromParent(splashView);
18    mSplashContainer.removeAllViews();
19    mSplashContainer.addView(splashView);
20 }
```

五、获取展示后广告信息事例

```

1  MediationBaseManager mediationManager = mSplashAd.getMediationManager();
2
3  printShowEcpmInfo(mediationManager)
4
5  /**
6   * 打印展示后Ecpm信息
7   */
8  public static void printShowEcpmInfo(MediationBaseManager mediationManager) {
9      if (mediationManager != null) {
10         MediationAdEcpmInfo showEcpm = mediationManager.getShowEcpm();
11         if (showEcpm != null) {
12             logEcpmInfo(showEcpm);
13         }
14     }
15 }
16
17 /**
18 * MediationAdEcpmInfo 字段参数如下:
19 */
20 public static void logEcpmInfo(MediationAdEcpmInfo item){
21     Log.d(Const.TAG, "EcpmInfo: \n" +
22         "adn名称 SdkName: " + item.getSdkName() + ",\n" +
23         "自定义adn名称 CustomSdkName: " + item.getCustomSdkName() + ",\n" +
24         "代码位Id PlacementID: " + item.getPlacementId() + ",\n" +
25         "广告价格 Ecpm: " + item.getEcpm() + ",\n" +
26         "广告竞价类型 ReqBiddingType: " + item.getReqBiddingType() + ",\n" +
27         "多阶底价标签 LevelTag: " + item.getLevelTag() + ",\n" +
28         "多阶底价标签解析失败原因 ErrorMsg: " + item.getErrorMsg() + ",\n" +
29         "adn请求Id RequestId: " + item.getRequestId() + ",\n" +
30         "广告类型 RitType: " + item.getRitType() + ",\n" +
31         "AB实验Id AbTestId: " + item.getAbTestId() + ",\n" +
32         "场景Id ScenarioId: " + item.getScenarioId() + ",\n" +
33         "流量分组Id SegmentId: " + item.getSegmentId() + ",\n" +
34         "流量分组渠道 Channel: " + item.getChannel() + ",\n" +
35         "流量分组子渠道 SubChannel: " + item.getSubChannel() + ",\n" +
36         "开发者传入的自定义数据 customData: " + item.getCustomData()
37     );
38 }

```

六、销毁广告

```

1  protected void onDestroy() {
2      super.onDestroy();
3      if (mSplashAd != null && mSplashAd.getMediationManager() != null) {
4          mSplashAd.getMediationManager().destroy();
5      }
6  }

```

3.4 接口说明

AdPlacement.Builder说明

方法名	说明
setCodeId(String codeId)	聚合广告位ID
setImageAcceptedSize(int width, int height)	设置宽高，单位px
setUserID(String userID)	设置userid
setMediationAdPlacement(MediationAdPlacement mediationAdPlacement)	设置聚合广告请求参数
supportRenderControl()	

MediationAdPlacement.Builder 说明

方法名	说明
setMuted(boolean var1)	设置静音
setVolume(float var1)	设置音量范围0~1；静音设置为0
setUseSurfaceView(boolean var1)	是否使用SurfaceView
setExtraObject(String var1, Object var2)	设置额外参数
setBidNotify(boolean var1)	bidding类型广告，竞价成功或者失败后是否通知对应的ADN
setScenarioId(String var1)	广告场景ID

setSplashShakeButton(boolean var1)	开屏摇一摇开关
setSplashPreLoad(boolean var1)	是否开启预加载广告
setMediationSplashRequestInfo(MediationSplashRequestInfo var1)	开屏兜底配置，开启聚合设置条件下，在开屏广告下传入此字段，如果本地没有配置会优先初始化设置的ADN，请求设置的开屏代码位。 需要注意设置的ADN必须与服务端设置的一致，否则会导致广告加载失败

MediationSplashRequestInfo 说明

开屏自定义兜底配置

方法名	说明
public MediationSplashRequestInfo(String adnName, String adnPlacementId, String appld, String appkey)	adnName：请参考 MediationConstant.ADN_MINTEGRAL； adnPlacementId：ADN平台上申请的代码位ID； appld：应用ID； appkey：如果没有可以传空；

NGAdBase.SplashAdListener 说明

广告加载回调

方法名	说明
void onSplashLoadSuccess()	广告物料、素材加载成功
void onSplashLoadFail(NGAdError ngAdError)	广告物料、素材加载失败或超时回调
void onSplashRenderSuccess(NGSplashAd ad)	广告渲染成功
void onSplashRenderFail(NGSplashAd ad, NGAdError ngAdError)	广告渲染失败

NGSplashAd 说明

方法名	说明
-----	----

View getSplashView()	获取开屏广告
View getSplashClickEyeView()	获取点睛View
View getSplashCardView()	获取卡片View
Map<String, Object> getMediaExtraInfo()	广告额外信息
int[] getSplashClickEyeSizeToDp()	获取点睛View尺寸
void startClickEye()	通知SDK开始点睛动画
void setSplashAdListener(SplashAdListener splashAdListener)	广告交互
void setSplashClickEyeListener(SplashClickEyeListener clickEyeListener)	注册点睛回调
void setSplashCardListener(SplashCardListener cardListener)	注册卡片阶段回调
void showSplashView(ViewGroup viewGroup)	告知SDK展示开屏
void showSplashClickEyeView(ViewGroup viewGroup)	告知SDK展示点睛
void showSplashCardView(ViewGroup viewGroup, Activity activity)	告知SDK展示卡片
MediationSplashManager getMediationManager()	获取聚合广告信息

SplashAdListener 说明

方法名	说明
void onSplashAdShow(NGSplashAd ad)	开屏展示
void onSplashAdClick(NGSplashAd ad)	开屏点击


```
void onSplashAdClose(NGSplashAd ad,  
int closeType)
```

开屏关闭，有些ADN会调用多次close回调需要开发者特殊处理

closeType枚举说明

```
1 public interface CSJSplashCloseType {  
2     int CLICK_SKIP = 1; // 跳过  
3     int COUNT_DOWN_OVER = 2; // 点击  
    倒计时结束  
4     int CLICK_JUMP = 3; // 点击跳转  
5     int VIDEO_PLAYER_COMPLETE = 4;  
    // 视频类广告播放完成  
6 }
```

SplashClickListener 说明

方法名	说明
void onSplashClickEyeReadyToShow(NGSplashAd bean)	通知媒体可以展示点睛
void onSplashClickEyeClick()	媒体点睛点击回调
void onSplashClickEyeClose()	点睛关闭回调

SplashCardListener 说明

方法名	说明
void onSplashCardReadyToShow(NGSplashAd bean)	通知媒体可以展示开屏卡片
void onSplashCardClick()	开屏卡片点击回调
void onSplashCardClose()	开屏卡片关闭回调

MediationSplashManager 说明

方法名	说明
boolean isReady()	当前广告是否可以展示
MediationAdEcpmInfo getShowEcpm()	为了确保数据的准确性，强烈建议在展示后获取展示广告的详细信息，包括广告位类型–getRitType、流量分组ID–getSegmentId、AB实验分组ID–getABTestId、渠道名称–getChannel、子渠道名称–getSubChannel、场景ID–getScenarioId、价格–getEcpm、ADN平台–getSdkName
List<MediationAdLoadInfo> getAdLoadInfo()	load回调后获取广告加载信息
List<MediationAdEcpmInfo> getMultiBiddingEcpm()	load回调之后获取获取多阶底价，clientBidding serverBidding的广告信息
MediationAdEcpmInfo getBestEcpm()	load回调之后获取最优广告价格信息
List<MediationAdEcpmInfo> getCacheList()	load回调之后，获取当前缓存池的全部信息
void destroy()	广告销毁

MediationAdEcpmInfo说明

方法名	说明
Map<String, String> getCustomData()	自定义参数
String getSdkName()	获取SDK名称
String getCustomSdkName()	获取自定义ADN名称
String getSlotId()	获取代码位ID
String getLevelTag()	获取多阶底价标签
String getEcpm()	获取价格
int getReqBiddingType()	获取bidding类型
String getErrorMsg()	获取错误信息
String getRequestId()	获取请求唯一标识

String getRitType()	获取代码位类型
String getSegmentId()	获取流量分组ID
String getChannel()	获取渠道
String getSubChannel()	获取子渠道
String getAbTestId()	获取AB测试ID
String getScenarioId()	获取场景ID

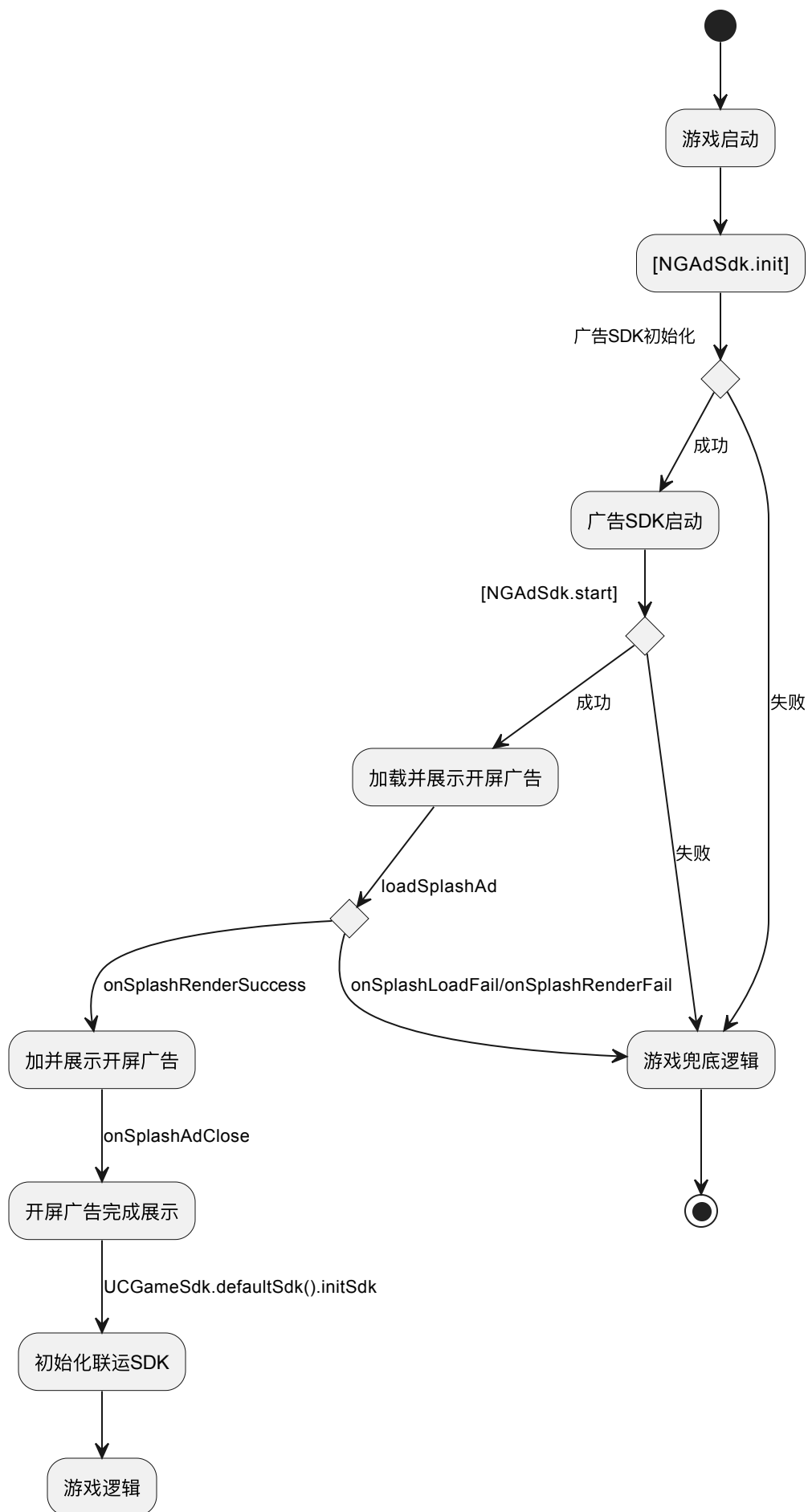
3.5 高级功能

3.5.1 与网游联运SDK集成时机

注意：

- ① 为保证开屏广告展示成功，请在开屏**展示完成**后再初始化网游联运SDK，防止被网游联动SDK初始化动画遮罩，影响计费。
- ② **时机流程**：广告SDK初始化 --> 开屏广告加载与展示 --> 开屏广告展示完成 --> 网游联运SDK初始化
- ③ **处理开屏广告的点击逻辑**：点击触发时，建议暂缓下一步逻辑（比如初始联动SDK），让广告跳转到点击结果页面，提高epcm等。

时机流程图




```

1  public class GameActivity extends Activity {
2      private static final String TAG = "GameActivity";
3
4      @Override
5      public void onCreate(Bundle b) {
6          super.onCreate(b);
7          this setContentView(R.layout.splashscreen);
8
9          mSplashContainer = findViewById(R.id.fl_content);
10
11         // 1. 初始化广告SDK
12         NGAdSdk.init(GameActivity2.this, new NGAdConfig.Builder()
13             .setAppId("xxxx")
14             .setAppName("APP测试媒体")
15             //.setDebug(true)
16             .build(), new NGAdSdk.Callback() {
17             @Override
18             public void success() {
19                 // 2. 广告SDK 初始化成功, 开始展示广告
20                 NGAdSdk.start(new NGAdSdk.Callback() {
21                     @Override
22                     public void success() {
23                         // 3. 加载并展示开屏广告
24                         loadAndShowSplashAd(); // 3.1 进这个方法内部监听#mNG
25                         SplashAdListener, 在开屏展示完成后, 初始化UCGameSDK
26                     }
27
28                     @Override
29                     public void fail(int code, String msg) {
30                         // 广告SDK load 失败, 游戏执行兜底逻辑: 比如初始化UCGam
31                         eSDK
32                     }
33                 });
34             }
35
36             @Override
37             public void fail(int code, String msg) {
38                 // 广告SDK 初始化失败, 游戏执行兜底逻辑: 比如初始化UCGameSDK
39             }
40         });
41
42         private FrameLayout mSplashContainer;
43

```

```

44     private NGSplashAd mSplashAd;
45
46     private NGAdBase.SplashAdListener mNGSplashAdListener;
47
48     private NGSplashAd.SplashAdListener mNGSplashInteractionListener;
49
50     private void loadAndShowSplashAd() {
51         AdPlacement adPlacement = new AdPlacement.Builder()
52             .setCodeId("xxxx")
53             .setImageAcceptedSize(UIUtil.getScreenWidthInPx(this), UI
54 Util.getScreenHeightInPx(this))
55             .build();
56
57         NGAdBase adBaseLoader = NGAdManagerHolder.get().createAdNative(th
58 is);
59
60         initListeners();
61
62         // 4. 请求加载开屏广告
63         adBaseLoader.loadSplashAd(adPlacement, mNGSplashAdListener, 350
64 0);
65     }
66
67     private void initListeners() {
68         // 广告加载监听器
69         this.mNGSplashAdListener = new NGAdBase.SplashAdListener() {
70             @Override
71             public void onSplashRenderSuccess(NGSplashAd ad) {
72                 /* 5、渲染成功后，展示广告 */
73
74                 mSplashAd = ad;
75                 ad.setSplashAdListener(mNGSplashInteractionListener);
76                 View splashView = ad.getSplashView();
77                 UIUtil.removeFromParent(splashView);
78                 mSplashContainer.removeAllViews();
79                 mSplashContainer.addView(splashView);
80             }
81
82             @Override
83             public void onSplashLoadSuccess(NGSplashAd ad) {
84
85             }
86
87             @Override
88             public void onSplashLoadFail(NGAdError ngAdError) {
89                 // 广告加载失败，游戏执行兜底逻辑：比如初始化UCGameSDK
90             }
91
92             @Override

```

```

89         public void onSplashRenderFail(NGSplashAd ad, NGAdError ngAdE
90         rror) {
91             // 广告渲染失败，游戏执行兜底逻辑：比如初始化UCGameSDK
92         }
93     };
94
95     // 广告展示监听器
96     this.mNGSplashInteractionListener = new NGSplashAd.SplashAdListen
97     er() {
98         @Override
99         public void onSplashAdShow(NGSplashAd ad) {
100             // 广告成功展示（即开始展示）
101         }
102
103         @Override
104         public void onSplashAdClick(NGSplashAd ad) {
105             Log.d(Const.TAG, "splash click");
106             // 监听到点击事件，暂缓下一步逻辑，展示点击结果页面
107         }
108
109         @Override
110         public void onSplashAdClose(NGSplashAd ad, int closeType) {
111             // 最后一步：开屏广告展示成功结束
112             mSplashContainer.removeAllViews(); // 移除（如果开屏广告的加
113             载页面不是独立Activity，则需要移除防止遮罩游戏）
114
115             // 开始初始化UCGameSDK
116             // ucNetworkAndInitUCGameSDK();
117             // UCGameSdk.defaultSdk().registerSDKEventReceiver(receiv
118         er);
119     };
120 }

```

3.5.2 首次加载超时解决方案

针对应用首次安装后请求开屏广告超时的情况，提供了API给开发者以解决此问题，针对不同广告平台，开发者需要传递广告平台的各项参数给相关的API，SDK将直接发起请求（只会针对设置的那一家广告平台进行请求，后续将按照后台配置请求广告）

注意：

①确保传入MediationSplashRequestInfo中的是各个广告平台的参数，而非SDK的参数。

②兜底开屏广告支持：穿山甲（CSJ）、广点通（GDT）。

③兜底代码位和瀑布流无关，互不影响。

```
▼ Plain Text |
1  //第一步、创建开屏自定义兜底对象
2  MediationSplashRequestInfo ngSplashRequestInfo = new MediationSplashRequestInfo(
3      MediationConstant.ADN_GDT, // GDT
4      "代码位Id", // adn开屏广告代码位Id, 注意不是聚合广告位Id
5      "appId",    // adn应用id, 注意要跟初始化传入的保持一致
6      "appKey"    // adn没有appKey时, 传入空即可
7  ) {};
8
9  //第二步、创建AdPlacement
10 AdPlacement adPlacement = new AdPlacement.Builder()
11     .setCodeId(getResources().getString(R.string.splash_media_id))
12     .setImageAcceptedSize(widthPx, heightPx)
13     .setMediationAdSlot(
14         new MediationAdSlot.Builder()
15             //将自定义兜底对象设置给AdPlacement
16             .setMediationSplashRequestInfo(ngSplashRequestInfo)
17             .build())
18     .build();
19
20 // 第三步, 请求广告
21 NGAdBase adBaseLoader = NGAdapterManagerHolder.get().createAdNative(this);
22 adBaseLoader.loadSplashAd(adPlacement, mNGSplashAdListener, 3500);
```

示例代码

```
1  package cn.sirius.nga.activity;
2
3  import android.app.Activity;
4  import android.os.Bundle;
5  import android.util.Log;
6  import android.view.View;
7  import android.widget.FrameLayout;
8
9  import androidx.annotation.Nullable;
10
11  import cn.sirius.nga.R;
12  import cn.sirius.nga.ad.NGAdBase;
13  import cn.sirius.nga.ad.NGAdError;
14  import cn.sirius.nga.ad.NGSplashAd;
15  import cn.sirius.nga.ad.NGSplashCloseType;
16  import cn.sirius.nga.config.AdPlacement;
17  import cn.sirius.nga.config.NGAdManagerHolder;
18  import cn.sirius.nga.util.Const;
19  import cn.sirius.nga.util.UIUtil;
20
21  /**
22   * 融合demo，开屏广告使用示例。更多功能参考接入文档。
23   * <p>
24   * 注意：每次加载的广告，只能展示一次
25   * <p>
26   * 接入步骤：
27   * 1、创建AdPlacement对象
28   * 2、创建NGAdBase对象
29   * 3、创建加载、展示监听器
30   * 4、加载广告
31   * 5、加载并渲染成功后，展示广告
32   * 6、在onDestroy中销毁广告
33   */
34  public class MediationSplashActivity extends Activity {
35
36      private FrameLayout mSplashContainer;
37
38      private NGSplashAd mSplashAd;
39
40      private NGAdBase.SplashAdListener mNGSplashAdListener;
41
42      private NGSplashAd.SplashAdListener mNGSplashInteractionListener;
43
44      @Override
45      protected void onCreate(@Nullable Bundle savedInstanceState) {
```

```

46         super.onCreate(savedInstanceState);
47         setContentView(R.layout.mediation_activity_splash);
48         mSplashContainer = findViewById(R.id.fl_content);
49
50         // 加载并展示广告
51         loadAndShowSplashAd();
52     }
53
54     private void loadAndShowSplashAd() {
55         /** 1、创建AdPlacement对象 */
56         AdPlacement adPlacement = new AdPlacement.Builder()
57             .setCodeId(getResources().getString(R.string.splash_media
58 _id))
59             .setImageAcceptedSize(UIUtil.getScreenWidthInPx(this), UI
60 Util.getScreenHeightInPx(this))
61             .build();
62
63         /** 2、创建NGAdBase对象 */
64         NGAdBase adBaseLoader = NGAdapterManagerHolder.get().createAdNative(th
65 is);
66
67         /** 3、创建加载、展示监听器 */
68         initListeners();
69
70         /** 4、加载广告 */
71         adBaseLoader.loadSplashAd(adPlacement, mNGSplashAdListener, 350
72 0);
73     }
74
75     private void initListeners() {
76         // 广告加载监听器
77         this.mNGSplashAdListener = new NGAdBase.SplashAdListener() {
78             @Override
79             public void onSplashRenderSuccess(NGSplashAd ad) {
80                 /** 5、渲染成功后，展示广告 */
81                 Log.d(Const.TAG, "splash render success");
82                 mSplashAd = ad;
83                 ad.setSplashAdListener(mNGSplashInteractionListener);
84                 View splashView = ad.getSplashView();
85                 UIUtil.removeFromParent(splashView);
86                 mSplashContainer.removeAllViews();
87                 mSplashContainer.addView(splashView);
88             }
89
90             @Override
91             public void onSplashLoadSuccess(NGSplashAd ad) {
92                 Log.d(Const.TAG, "splash load success");
93             }
94         }
95     }

```

```

90
91         @Override
92         public void onSplashLoadFail(NGAdError ngAdError) {
93             Log.d(Const.TAG, "splash load fail, errCode: " + ngAdError
94 r.getCode() + ", errMsg: " + ngAdError.getMsg());
95         }
96
97         @Override
98         public void onSplashRenderFail(NGSplashAd ad, NGAdError ngAdE
99 rror) {
100             Log.d(Const.TAG, "splash render fail, errCode: " + ngAdEr
101 rror.getCode() + ", errMsg: " + ngAdError.getMsg());
102         }
103     };
104
105     // 广告展示监听器
106     this.mNGSplashInteractionListener = new NGSplashAd.SplashAdListen
107 er() {
108         @Override
109         public void onSplashAdShow(NGSplashAd ad) {
110             Log.d(Const.TAG, "splash show");
111         }
112
113         @Override
114         public void onSplashAdClick(NGSplashAd ad) {
115             Log.d(Const.TAG, "splash click");
116         }
117
118         @Override
119         public void onSplashAdClose(NGSplashAd ad, int closeType) {
120             if (closeType == NGSplashCloseType.CLICK_SKIP) {
121                 Log.d(Const.TAG, "开屏广告点击跳过");
122             } else if (closeType == NGSplashCloseType.COUNT_DOWN_OVE
123 R) {
124                 Log.d(Const.TAG, "开屏广告点击倒计时结束");
125             } else if (closeType == NGSplashCloseType.CLICK_JUMP) {
126                 Log.d(Const.TAG, "点击跳转");
127             }
128             finish();
129         }
130     };
131
132     @Override
133     protected void onDestroy() {

```

```

133         super.onDestroy();
134         /** 6、在onDestroy中销毁广告 */
135         if (mSplashAd != null && mSplashAd.getMediationManager() != null) {
136             mSplashAd.getMediationManager().destroy();
137         }
138     }
139 }

```

接入事例可参照demo内MediationSplashActivity类。

4. 激励视频广告

4.1 简介

本SDK为接入方提供激励视频广告，该广告的效果为观看完毕视频广告，发放奖励给用户。使用场景包括但不限于：

- ① 游戏等应用内观看视频广告获得游戏内金币等：用户必须观看完整视频才能获取奖励。
- ② 积分类应用接入；支持的广告尺寸：全屏横屏播放和竖屏。
- ③ 返回无网络相关的错误码，建议提示用户联网后再重试。

4.2 注意事项

- ① 聚合SDK是通过[广告位ID](#)发起广告请求的，切记不要使用混淆。
- ② 在广告接入前需要明确[各ADN对应聚合SDK广告样式](#)情况，以确保正确完成广告接入，避免由于广告类型不匹配导致接入报错等情况的发生。
- ③ 由于各广告平台对于包名校验规则不同，需确保在NGAd平台填写的包名符合各ADN平台规范，避免由于包名校验不匹配导致的无广告返回情况的产生。
- ④ 广告请求时机，**建议在收到SDK初始化成功回调后发起广告请求**，当SDK初始化回调一直失败时，建议首先明确应用ID及广告位ID是否赋值正确、是否有多余空格、是否是网络不稳定导致的超时等，当排查后无法定位问题时，建议通过抓包将config字段下的加密内容提供过来，我方协助定位问题。
- ⑤ 支持多进程，如需开启可在初始化时设置.supportMultiProcess(true)，默认false。

- ⑥ 为了保证播放流畅，建议在收到`onRewardVideoCached()`之后进行广告展示，广告展示前可通过`isReady`来判断当前广告是否可用。
- ⑦ 如若针对展示失败有重试机制，建议只重试一次即可，避免无限重试引发死循环场景。
- ⑧ 当展示失败/回调监听不执行时，建议检查是否有广告对象被覆盖或者被提前释放的场景导致，每次发次广告请求时，需要重新创建新的广告来调用广告请求方法。
- ⑨ `onVideoComplete()`视频播放完整回调和`onAdVideoBarClick()`点击回调不是所有ADN都会触发，如若在此回调内有相关逻辑处理或者打点统计需注意。

4.3 接入激励视频广告

一、创建NGAdBase对象

```
1  NGAdBase adBaseLoader = NGAdManagerHolder.get().createAdNative(this); //需要传activity，切记!!
```

二、创建广告请求AdPlacement

```
1  AdPlacement adPlacement = new AdPlacement.Builder()  
2      .setCodeId("聚合广告位ID")  
3      .setOrientation(NGAdConstant.VERTICAL)//横竖屏设置  
4      .setImageAcceptedSize(widthPx, heightPx)  
5      .build();
```

三、请求广告

```
1  adBaseLoader.loadRewardVideoAd(adPlacement, new NGAdBase.RewardVideoAdList
   ener() {
2      @Override
3      public void onError(int code, String message) {
4
5      }
6
7      @Override
8      public void onRewardVideoAdLoad(NGRewardVideoAd ad) {
9          mNGRewardVideoAd = ad;
10     }
11
12     @Override
13     public void onRewardVideoCached(NGRewardVideoAd ad) {
14         mNGRewardVideoAd = ad;
15     }
16 });
```

四、展示广告

为了确保播放流程建议在cached回调后调用广告展示并判断isReady状态

```
1  mNGRewardVideoAd.setRewardAdInteractionListener(new NGRewardVideoAd.Reward
   AdInteractionListener() {
2      @Override
3      public void onAdShow() {
4
5      }
6
7      @Override
8      public void onAdVideoBarClick() {
9
10     }
11
12     @Override
13     public void onAdClose() {
14
15     }
16
17     @Override
18     public void onVideoComplete() {
19
20     }
21
22     @Override
23     public void onVideoError() {
24
25     }
26
27     @Override
28     public void onRewardArrived(boolean isRewardValid, int rewardType, Bundle
   extraInfo) {
29         // 当用户的观看行为满足了奖励条件
30         RewardBundleModel rewardBundleModel = new RewardBundleModel(extraInfo);
31         Log.e(Const.TAG, "Callback --> rewardVideoAd has onRewardArrived " +
32             "\n奖励是否有效: " + isRewardValid +
33             "\n奖励类型: " + rewardType +
34             "\n奖励名称: " + rewardBundleModel.getRewardName() +
35             "\n奖励数量: " + rewardBundleModel.getRewardAmount() +
36             "\n建议奖励百分比: " + rewardBundleModel.getRewardPropose());
37
38         if (!isRewardValid) {
39             Log.d(Const.TAG, "发送奖励失败 code: " + rewardBundleModel.getServer
   ErrorCode() +
40                 "\n msg: " + rewardBundleModel.getServerErrorMsg());
41         }
```



```
42     }  
43  
44     @Override  
45     public void onSkippedVideo() {  
46  
47     }  
48 });  
49  
50 mNGRewardVideoAd.showRewardVideoAd(this);
```

五、获取展示后广告信息事例

```

1  MediationBaseManager mediationManager = mNGRewardVideoAd.getMediationManager();
2
3  printShowEcpmInfo(mediationManager)
4
5  /**
6   * 打印展示后Ecpm信息
7   */
8  public static void printShowEcpmInfo(MediationBaseManager mediationManager) {
9      if (mediationManager != null) {
10         MediationAdEcpmInfo showEcpm = mediationManager.getShowEcpm();
11         if (showEcpm != null) {
12             logEcpmInfo(showEcpm);
13         }
14     }
15 }
16
17 /**
18 * MediationAdEcpmInfo 字段参数如下:
19 */
20 public static void logEcpmInfo(MediationAdEcpmInfo item){
21     Log.d(Const.TAG, "EcpmInfo: \n" +
22         "adn名称 SdkName: " + item.getSdkName() + ",\n" +
23         "自定义adn名称 CustomSdkName: " + item.getCustomSdkName() + ",\n" +
24         "代码位Id PlacementID: " + item.getPlacementId() + ",\n" +
25         "广告价格 Ecpm: " + item.getEcpm() + ",\n" +
26         "广告竞价类型 ReqBiddingType: " + item.getReqBiddingType() + ",\n" +
27         "多阶底价标签 LevelTag: " + item.getLevelTag() + ",\n" +
28         "多阶底价标签解析失败原因 ErrorMsg: " + item.getErrorMsg() + ",\n" +
29         "adn请求Id RequestId: " + item.getRequestId() + ",\n" +
30         "广告类型 RitType: " + item.getRitType() + ",\n" +
31         "AB实验Id AbTestId: " + item.getAbTestId() + ",\n" +
32         "场景Id ScenarioId: " + item.getScenarioId() + ",\n" +
33         "流量分组Id SegmentId: " + item.getSegmentId() + ",\n" +
34         "流量分组渠道 Channel: " + item.getChannel() + ",\n" +
35         "流量分组子渠道 SubChannel: " + item.getSubChannel() + ",\n" +
36         "开发者传入的自定义数据 customData: " + item.getCustomData()
37     );
38 }

```

六、销毁广告

```

1  protected void onDestroy() {
2      super.onDestroy();
3      if (mNGRewardVideoAd != null && mNGRewardVideoAd.getMediationManager() != null) {
4          mNGRewardVideoAd.getMediationManager().destroy();
5      }
6  }

```

4.4 接口说明

AdPlacement.Builder说明

方法名	说明
setCodeId(String codeId)	聚合广告位ID
setOrientation(@NGAdConstant.ORIENTATION_STATE int orientation)	设置横竖屏方向
setUserID(String userID)	设置userid
setMediationAdPlacement(MediationAdPlacement mediationAdPlacement)	设置聚合广告请求参数

MediationAdPlacement.Builder 说明

方法名	说明
setMuted(boolean var1)	设置静音
setVolume(float var1)	设置音量范围0~1；静音设置为0
setUseSurfaceView(boolean var1)	是否使用SurfaceView
setExtraObject(String var1, Object var2)	设置额外参数
setBidNotify(boolean var1)	bidding类型广告，竞价成功或者失败后是否通知对应的ADN
setScenarioId(String var1)	广告场景ID
setRewardName(String var1)	设置激励名称

setRewardAmount(int var1)	设置激励金额
---------------------------	--------

NGAdBase.RewardVideoAdListener 说明

广告加载回调

方法名	说明
void onRewardVideoAdLoad(NGRewardVideoAd ad)	加载完成的回调
void onRewardVideoCached(NGRewardVideoAd ad)	广告视频本地加载完成的回调
void onError(int code, String message)	加载失败回调

NGRewardVideoAd 说明

方法名	说明
void setRewardAdInteractionListener(RewardAdInteractionListener listener)	设置激励事件监听
void setRewardPlayAgainInteractionListener(RewardAdInteractionListener listener)	再看一个功能
void showRewardVideoAd(Activity activity)	展示激励广告
Map<String, Object> getMediaExtraInfo()	返回广告额外信息
void showRewardVideoAd(Activity activity, TTAdConstant.RitScenes ritScenes, String scenes)	展示广告，ritScenes, scenes参数不生效
MediationRewardManager getMediationManager()	获取聚合广告信息

NGAdConstant.RitScenes 说明

方法名	说明
CUSTOMIZE_SCENES("customize_scenes"),	常量定义，广告场景使用

<i>HOME_OPEN_BONUS</i> ("home_open_bonus"),
<i>HOME_SVIP_BONUS</i> ("home_svip_bonus"),
<i>HOME_GET_PROPS</i> ("home_get_props"),
<i>HOME_TRY_PROPS</i> ("home_try_props"),
<i>HOME_GET_BONUS</i> ("home_get_bonus"),
<i>HOME_GIFT_BONUS</i> ("home_gift_bonus"),
<i>GAME_START_BONUS</i> ("game_start_bonus"),
<i>GAME_REDUCE_WAITING</i> ("game_reduce_waiting"),
<i>GAME_MORE_KLLKRTUNITIES</i> (TTAdConstant.a("game_more_kllkrtunities")),
<i>GAME_FINISH_REWARDS</i> ("game_finish_rewards"),
<i>GAME_GIFT_BONUS</i> ("game_gift_bonus");

RewardAdPlayAgainController 说明

方法名	说明
String <i>KEY_PLAY_AGAIN_ALLOW</i> = "play_again_allow";	是否允许下次再看 <i>value</i> 为 <i>boolean</i> 类型
String <i>KEY_PLAY_AGAIN_REWARD_NAME</i> = "play_again_reward_name";	下次再看可得的奖励名称 <i>value</i> 为 <i>string</i> 类型
String <i>KEY_PLAY_AGAIN_REWARD_AMOUNT</i> = "play_again_reward_amount";	下次再看可得的奖励数量 <i>value</i> 为 <i>string</i> 类型

void getPlayAgainCondition(int nextPlayAgainCount, NGRewardVideoAd.RewardAdPlayAgainController.Callback callback)	<p>此方法在主线程回调，请注意另起子线程发起网络请求* 当广告可出下次再看时，回调给开发者，由开发者决定下次再看可否再出</p> <p>@param nextPlayAgainCount 下一次再看是第几次再看，值从1开始</p> <p>@param callback 开发者处理完返回给sdk的回调</p>
void onConditionReturn(Bundle playAgainBundle)	开发者处理完返回给sdk的回调

NGRewardVideoAd.RewardAdInteractionListener 说明

方法名	说明
void onAdShow()	广告展示
void onAdVideoBarClick()	广告的下载bar点击回调
void onAdClose()	广告关闭的回调
void onVideoComplete()	视频播放完毕的回调
void onVideoError()	视频播放失败的回调
void onRewardArrived(boolean isRewardValid, int rewardType, Bundle extraInfo)	<p>激励视频播放完毕，验证是否有效发放奖励的回调</p> <p>isRewardValid：是否发放奖励，true：发奖励；false：不发奖励</p> <p>rewardType：奖励类型，0:基础奖励 >0:进阶奖励</p> <p>extraInfo：奖励的额外参数</p>

MediationRewardManager 说明

方法名	说明
boolean isReady()	当前广告是否可以展示

MediationAdEcpmlInfo getShowEcpm()	为了确保数据的准确性，强烈建议在展示后获取展示广告的详细信息，包括广告位类型–getRitType、流量分组ID–getSegmentId、AB实验分组ID–getABTestId、渠道名称–getChannel、子渠道名称–getSubChannel、场景ID–getScenariold、价格–getEcpm、ADN平台–getSdkName
List<MediationAdLoadInfo> getAdLoadInfo()	load回调后获取广告加载信息
List<MediationAdEcpmlInfo> getMultiBiddingEcpm()	load回调之后获取获取多阶底价，clientBidding serverBidding的广告信息
MediationAdEcpmlInfo getBestEcpm()	load回调之后获取最优广告价格信息
List<MediationAdEcpmlInfo> getCacheList()	load回调之后，获取当前缓存池的全部信息
void destroy()	广告销毁

MediationAdEcpmlInfo说明

方法名	说明
Map<String, String> getCustomData()	自定义参数
String getSdkName()	获取SDK名称
String getCustomSdkName()	获取自定义ADN名称
String getSlotId()	获取代码位ID
String getLevelTag()	获取多阶底价标签
String getEcpm()	获取价格
int getReqBiddingType()	获取bidding类型
String getErrorMsg()	获取错误信息
String getRequestId()	获取请求唯一标识
String getRitType()	获取代码位类型
String getSegmentId()	获取流量分组ID

String getChannel()	获取渠道
String getSubChannel()	获取子渠道
String getAbTestId()	获取AB测试ID
String getScenarioId()	获取场景ID

4.5 示例代码


```
1 package cn.sirius.nga.activity;
2
3 import android.app.Activity;
4 import android.os.Bundle;
5 import android.util.Log;
6 import android.view.View;
7 import android.widget.TextView;
8
9 import androidx.annotation.Nullable;
10
11 import cn.sirius.nga.R;
12 import cn.sirius.nga.ad.NGAdBase;
13 import cn.sirius.nga.ad.NGRewardVideoAd;
14 import cn.sirius.nga.config.AdPlacement;
15 import cn.sirius.nga.config.NGAdConstant;
16 import cn.sirius.nga.config.NGAdManagerHolder;
17 import cn.sirius.nga.mediation.impl.gromore.MediationAdPlacement;
18 import cn.sirius.nga.util.Const;
19 import cn.sirius.nga.util.RewardBundleModel;
20 import cn.sirius.nga.util.ToastUtil;
21
22 /**
23  * 融合demo，激励视频广告使用示例。更多功能参考接入文档。
24  * <p>
25  * 注意：每次加载的广告，只能展示一次
26  * <p>
27  * 接入步骤：
28  * 1、创建AdPlacement对象
29  * 2、创建NGAdBase对象
30  * 3、创建加载、展示监听器
31  * 4、加载广告
32  * 5、加载成功后，展示广告
33  * 6、在onDestroy中销毁广告
34  */
35 public class MediationRewardActivity extends Activity {
36
37     public String mMediaId; // 融合广告位
38
39     private NGRewardVideoAd mNGRewardVideoAd; // 插全屏广告对象
40
41     private NGAdBase.RewardVideoAdListener mRewardVideoListener; // 广告加
    载监听器
42
43     private NGRewardVideoAd.RewardAdInteractionListener mRewardVideoAdInt
    eractionListener; // 广告展示监听器
```

```

44
45     @Override
46     protected void onCreate(@Nullable Bundle savedInstanceState) {
47         super.onCreate(savedInstanceState);
48         setContentView(R.layout.mediation_activity_reward);
49
50         // 聚合广告位 (在NG平台的广告位, 注意不是adn的代码位)
51         this.mMediaId = getResources().getString(R.string.reward_media_i
52 d);
53         TextView tvMediationId = findViewById(R.id.tv_media_id);
54         tvMediationId.setText(String.format(getResources().getString(R.st
55 ring.ad_mediation_id), mMediaId));
56
57         // 广告加载
58         findViewById(R.id.bt_load).setOnClickListener(new View.OnClickLis
59 tener() {
60             @Override
61             public void onClick(View v) {
62                 loadRewardVideoAd();
63             }
64         });
65
66         // 广告展示
67         findViewById(R.id.bt_show).setOnClickListener(new View.OnClickLis
68 tener() {
69             @Override
70             public void onClick(View v) {
71                 showRewardVideoAd();
72             }
73         });
74     }
75
76     private void loadRewardVideoAd() {
77         /** 1、创建AdPlacement对象 */
78         AdPlacement adPlacement = new AdPlacement.Builder()
79             .setCodeId(mMediaId)
80             .setMediationAdPlacement(new MediationAdPlacement.Builder
81 ()
82             .setMuted(true)//是否静音
83             .setVolume(0.0f)//设置音量
84             .setBidNotify(true)//竞价结果通知
85             .build())
86             .setOrientation(NGAdConstant.ORIENTATION_VERTICAL)
87             .build();
88
89         /** 2、创建NGAdBase对象 */
90         NGAdBase adBaseLoader = NGAdapterManagerHolder.get().createAdNative(th
91 is);

```

```

86
87     /** 3、创建加载、展示监听器 */
88     initListeners();
89
90     /** 4、加载广告 */
91     adBaseLoader.loadRewardVideoAd(adPlacement, mRewardVideoListene
92 r);
93
94     ToastUtil.show(MediationRewardActivity.this, "广告加载中, 请稍后");
95 }
96
97 // 广告加载成功后, 开始展示广告
98 private void showRewardVideoAd() {
99     if (mNGRewardVideoAd == null) {
100         Log.i(Const.TAG, "请先加载广告或等待广告加载完毕后再调用show方法");
101         return;
102     }
103     /** 5、设置展示监听器, 展示广告 */
104     mNGRewardVideoAd.setRewardAdInteractionListener(mRewardVideoAdInt
105 eractionListener);
106
107     mNGRewardVideoAd.showRewardVideoAd(this);
108 }
109
110 private void initListeners() {
111     // 广告加载监听器
112     this.mRewardVideoListener = new NGAdBase.RewardVideoAdListener()
113 {
114     @Override
115     public void onError(int code, String message) {
116         Log.i(Const.TAG, "reward load fail: errCode: " + code +
117 ", errMsg: " + message);
118         ToastUtil.show(MediationRewardActivity.this, "广告加载失败
119 onError code = " + code + " msg = " + message);
120     }
121
122     @Override
123     public void onRewardVideoAdLoad(NGRewardVideoAd ad) {
124         Log.i(Const.TAG, "reward load success");
125         mNGRewardVideoAd = ad;
126         ToastUtil.show(MediationRewardActivity.this, "广告加载成功,
127 可展示");
128     }
129
130     @Override
131     public void onRewardVideoCached(NGRewardVideoAd ad) {
132         Log.i(Const.TAG, "reward cached success 2");
133         mNGRewardVideoAd = ad;

```

```

128         }
129
130     };
131
132     // 广告展示监听器
133     this.mRewardVideoAdInteractionListener = new NGRewardVideoAd.RewardAdInteractionListener() {
134         @Override
135         public void onAdShow() {
136             Log.i(Const.TAG, "reward show");
137         }
138
139         @Override
140         public void onAdVideoBarClick() {
141             Log.i(Const.TAG, "reward click");
142         }
143
144         @Override
145         public void onAdClose() {
146             Log.i(Const.TAG, "reward close");
147         }
148
149         @Override
150         public void onVideoComplete() {
151             Log.i(Const.TAG, "reward onVideoComplete");
152         }
153
154         @Override
155         public void onVideoError() {
156             Log.i(Const.TAG, "reward onVideoError");
157         }
158
159         @Override
160         public void onRewardArrived(boolean isRewardValid, int rewardType, Bundle extraInfo) {
161             Log.i(Const.TAG, "reward onRewardArrived");
162             // 当用户的观看行为满足了奖励条件
163             RewardBundleModel rewardBundleModel = new RewardBundleModel(extraInfo);
164             Log.e(Const.TAG, "Callback --> rewardVideoAd has onRewardArrived " +
165                 "\n奖励是否有效: " + isRewardValid +
166                 "\n奖励类型: " + rewardType +
167                 "\n奖励名称: " + rewardBundleModel.getRewardName() +
168                 "\n奖励数量: " + rewardBundleModel.getRewardAmount() +
169

```

```

170         "\n建议奖励百分比: " + rewardBundleModel.getRewardP
171         ropose());
172         if (!isRewardValid) {
173             Log.d(Const.TAG, "发送奖励失败 code: " + rewardBundleMo
del.getServerErrorCode() +
174                 "\n msg: " + rewardBundleModel.getServerError
Msg());
175         }
176     }
177
178     @Override
179     public void onSkippedVideo() {
180         Log.i(Const.TAG, "reward onSkippedVideo");
181     }
182 };
183 }
184
185 @Override
186 protected void onDestroy() {
187     super.onDestroy();
188     /** 6、在onDestroy中销毁广告 */
189     if (mNGRewardVideoAd != null && mNGRewardVideoAd.getMediationMana
ger() != null) {
190         mNGRewardVideoAd.getMediationManager().destroy();
191     }
192 }

```

接入事例可参照demo内MediationSplashActivity类。

5. 插全屏广告

5.1. 简介

该广告类型支持插屏广告、全屏广告混出。

5.2 注意事项

- ① 聚合SDK是通过[广告位ID](#)发起广告请求的，切记不要使用混淆。
- ② 在广告接入前需要明确[各ADN对应聚合SDK广告样式](#)情况，以确保正确完成广告接入，避免由于广告类型不匹配导致接入报错等情况的发生。

- ③ 由于各广告平台对于包名校验规则不同，需确保在NGAd平台填写的包名符合各ADN平台规范，避免由于包名校验不匹配导致的无广告返回情况的产生。
- ④ 广告请求时机，**建议在收到SDK初始化成功回调后发起广告请求**，当SDK初始化回调一直失败时，建议首先明确appid及广告位ID是否赋值正确、是否有多余空格、是否是网络不稳定导致的超时等，当排查后无法定位问题时，建议通过抓包将config字段下的加密内容提供过来，我方协助定位问题。
- ⑤ 为了保证播放流畅，建议在收到onFullScreenVideoCached()之后进行广告展示，广告展示前可通过isReady来判断当前广告是否可用。
- ⑥ 如若针对展示失败有重试机制，建议只重试一次即可，避免无限重试引发死循环场景。
- ⑦ 支持多进程，如需开启可在初始化时设置.supportMultiProcess(true)，默认false。

5.3 接入插全屏广告

一、创建NGAdBase对象

```
1  NGAdBase adBaseLoader = NGAdManagerHolder.get().createAdNative(this); //需要传activity, 切记!!
```

二、创建广告请求AdPlacement

```
1  AdPlacement adPlacement = new AdPlacement.Builder()  
2      .setCodeId("聚合广告位ID")  
3      .setOrientation(NGAdConstant.VERTICAL)//横竖屏设置  
4      .setMediationAdPlacement(new MediationAdPlacement.Builder()  
5          .setMuted(false)//是否静音  
6          .setVolume(0.6f)//设置音量  
7          .setBidNotify(true)//竞价结果通知  
8          .build())  
9      .build();
```

三、请求广告

```
1  adBaseLoader.loadFullScreenVideoAd(adPlacement, new NGAdBase.FullScreenVideoAdListener() {
2      @Override
3      public void onError(int code, String message) {
4
5      }
6
7      @Override
8      public void onFullScreenVideoAdLoad(NGFullScreenVideoAd ad) {
9          mNGFullScreenVideoAd = ad;
10     }
11
12     @Override
13     public void onFullScreenVideoCached(NGFullScreenVideoAd ad) {
14         mNGFullScreenVideoAd = ad;
15     }
16 });
```

四、展示广告

为了确保播放流程建议在cached回调后调用广告展示并判断isReady状态

```
1  mNGFullScreenVideoAd.setFullScreenVideoAdInteractionListener(new NGFullScr
   eenVideoAd.FullScreenVideoAdInteractionListener() {
2      @Override
3      public void onAdShow() {
4
5      }
6
7      @Override
8      public void onAdVideoBarClick() {
9
10     }
11
12     @Override
13     public void onAdClose() {
14
15     }
16
17     @Override
18     public void onVideoComplete() {
19
20     }
21
22     @Override
23     public void onSkippedVideo() {
24
25     }
26 });
27
28  this.mNGFullScreenVideoAd.showFullScreenVideoAd(this);
```

五、获取展示后广告信息事例


```

1  MediationBaseManager mediationManager = mNGFullScreenVideoAd.getMediationM
   anager();
2
3  printShowEcpmInfo(mediationManager)
4
5  /**
6   * 打印展示后Ecpm信息
7   */
8  public static void printShowEcpmInfo(MediationBaseManager mediationManage
   r) {
9      if (mediationManager != null) {
10         MediationAdEcpmInfo showEcpm = mediationManager.getShowEcpm();
11         if (showEcpm != null) {
12             logEcpmInfo(showEcpm);
13         }
14     }
15 }
16
17 /**
18 * MediationAdEcpmInfo 字段参数如下:
19 */
20 public static void logEcpmInfo(MediationAdEcpmInfo item){
21     Log.d(Const.TAG, "EcpmInfo: \n" +
22         "adn名称 SdkName: " + item.getSdkName() + ",\n" +
23         "自定义adn名称 CustomSdkName: " + item.getCustomSdkName() + ",\n" +
24         "代码位Id PlacementID: " + item.getPlacementId() + ",\n" +
25         "广告价格 Ecpm: " + item.getEcpm() + ",\n" +
26         "广告竞价类型 ReqBiddingType: " + item.getReqBiddingType() + ",\n" +
27         "多阶底价标签 LevelTag: " + item.getLevelTag() + ",\n" +
28         "多阶底价标签解析失败原因 ErrorMsg: " + item.getErrorMsg() + ",\n" +
29         "adn请求Id RequestId: " + item.getRequestId() + ",\n" +
30         "广告类型 RitType: " + item.getRitType() + ",\n" +
31         "AB实验Id AbTestId: " + item.getAbTestId() + ",\n" +
32         "场景Id ScenarioId: " + item.getScenarioId() + ",\n" +
33         "流量分组Id SegmentId: " + item.getSegmentId() + ",\n" +
34         "流量分组渠道 Channel: " + item.getChannel() + ",\n" +
35         "流量分组子渠道 SubChannel: " + item.getSubChannel() + ",\n" +
36         "开发者传入的自定义数据 customData: " + item.getCustomData()
37     );
38 }

```

六、销毁广告

```

1  protected void onDestroy() {
2      super.onDestroy();
3      if (mNGFullScreenVideoAd != null && mNGFullScreenVideoAd.getMediationMa
nager() != null) {
4          mNGFullScreenVideoAd.getMediationManager().destroy();
5      }
6  }

```

5.4 接口说明

AdPlacement.Builder说明

方法名	说明
setCodeId(String codeId)	聚合广告位ID
setAdCount(int adCount)	加载广告数，最多不超过3个
setOrientation(@NGAdConstant.ORIENTATION_STATE int orientation)	设置横竖屏方向
setUserID(String userID)	设置userid
setMediationAdPlacement(MediationAdPlacement mediationAdPlacement)	设置聚合广告请求参数

MediationAdPlacement.Builder 说明

方法名	说明
setMuted(boolean var1)	设置静音
setVolume(float var1)	设置音量范围0~1；静音设置为0
setUseSurfaceView(boolean var1)	是否使用SurfaceView
setExtraObject(String var1, Object var2)	设置额外参数
setBidNotify(boolean var1)	bidding类型广告，竞价成功或者失败后是否通知对应的ADN

setScenarioId(String var1)	广告场景ID
----------------------------	--------

NGAdBase.FullScreenVideoAdListener 说明

广告加载回调

方法名	说明
void onFullScreenVideoAdLoad(NGFullScreenVideoAd ad)	广告加载成功
void onFullScreenVideoCached(NGFullScreenVideoAd ad)	广告视频本地加载完成的回调
void onError(int code, String message)	广告加载失败

NGFullScreenVideoAd 说明

方法名	说明
void setFullScreenVideoAdInteractionListener(FullScreenVideoAdInteractionListener listener)	注册插屏广告交互回调
void showFullScreenVideoAd(Activity activity)	展示广告
void showFullScreenVideoAd(Activity activity, TTAdConstant.RitScenes ritScenes, String scenes)	展示广告， ritScenes, scenes参数不生效
Map<String, Object> getMediaExtraInfo()	广告额外信息
MediationFullScreenManager getMediationManager()	获取聚合广告信息

MediationFullScreenManager 说明

方法名	说明
boolean isReady()	当前广告是否可以展示

MediationAdEcpmlInfo getShowEcpm()	为了确保数据的准确性，强烈建议在展示后获取展示广告的信息，包括广告位类型–getRitType、流量分组ID–getSegmentId、AB实验分组ID–getABTestId、渠道名称–getChannel、子渠道名称–getSubChannel、场景ID–getScenariold、价格–getEcpm、ADN平台–getSdkName
List<MediationAdLoadInfo> getAdLoadInfo()	load回调后获取广告加载信息
List<MediationAdEcpmlInfo> getMultiBiddingEcpm()	load回调之后获取获取多阶底价，clientBidding serverBidding的广告信息
MediationAdEcpmlInfo getBestEcpm()	load回调之后获取最优广告价格信息
List<MediationAdEcpmlInfo> getCacheList()	load回调之后，获取当前缓存池的全部信息
void destroy()	广告销毁

MediationAdEcpmlInfo说明

方法名	说明
Map<String, String> getCustomData()	自定义参数
String getSdkName()	获取SDK名称
String getCustomSdkName()	获取自定义ADN名称
String getSlotId()	获取代码位ID
String getLevelTag()	获取多阶底价标签
String getEcpm()	获取价格
int getReqBiddingType()	获取bidding类型
String getErrorMsg()	获取错误信息
String getRequestId()	获取请求唯一标识
String getRitType()	获取代码位类型
String getSegmentId()	获取流量分组ID

String getChannel()	获取渠道
String getSubChannel()	获取子渠道
String getAbTestId()	获取AB测试ID
String getScenarioId()	获取场景ID

5.6 示例代码

```
1  package cn.sirius.nga.activity;
2
3  import android.app.Activity;
4  import android.os.Bundle;
5  import android.util.Log;
6  import android.view.View;
7  import android.view.View.OnClickListener;
8  import android.widget.TextView;
9
10 import cn.sirius.nga.R;
11 import cn.sirius.nga.ad.NGAdBase;
12 import cn.sirius.nga.ad.NGFullScreenVideoAd;
13 import cn.sirius.nga.config.AdPlacement;
14 import cn.sirius.nga.config.NGAdConstant;
15 import cn.sirius.nga.config.NGAdManagerHolder;
16 import cn.sirius.nga.mediation.impl.gromore.MediationAdPlacement;
17 import cn.sirius.nga.util.Const;
18 import cn.sirius.nga.util.ToastUtil;
19
20 /**
21  * 融合demo，插全屏广告使用示例。更多功能参考接入文档。
22  * <p>
23  * 注意：每次加载的广告，只能展示一次
24  * <p>
25  * 接入步骤：
26  * 1、创建AdPlacement对象
27  * 2、创建NGAdBase对象
28  * 3、创建加载、展示监听器
29  * 4、加载广告
30  * 5、加载成功后，展示广告
31  * 6、在onDestroy中销毁广告
32  */
33 public final class MediationInterstitialFullActivity extends Activity {
34
35     public String mMediaId; // 融合广告位
36
37     private NGFullScreenVideoAd mNGFullScreenVideoAd; // 插全屏广告对象
38
39     private NGAdBase.FullScreenVideoAdListener mFullScreenVideoListener;
40     // 广告加载监听器
41
42     private NGFullScreenVideoAd.FullScreenVideoAdInteractionListener mFull
43     lScreenVideoAdInteractionListener; // 广告展示监听器
```

```

44     @Override
45     protected void onCreate(Bundle savedInstanceState) {
46         super.onCreate(savedInstanceState);
47         this setContentView(R.layout.mediation_activity_interstitial_full);
48     };
49
50     // 聚合广告位 (在NG平台的广告位, 注意不是adn的代码位)
51     this.mMediaId = getResources().getString(R.string.full_media_id);
52     TextView tvMediationId = this.findViewById(R.id.tv_media_id);
53     tvMediationId.setText(getString(R.string.ad_mediation_id, this.mMediaId));
54
55     // 加载广告
56     this.findViewById(R.id.bt_load).setOnClickListener(new OnClickListener() {
57         @Override
58         public void onClick(View it) {
59             loadInterstitialFullAd();
60         }
61     });
62
63     // 展示广告
64     this.findViewById(R.id.bt_show).setOnClickListener(new OnClickListener() {
65         @Override
66         public void onClick(View it) {
67             showInterstitialFullAd();
68         }
69     });
70
71     private void loadInterstitialFullAd() {
72         /** 1、创建AdPlacement对象 */
73         AdPlacement adPlacement = new AdPlacement.Builder()
74             .setCodeId(this.mMediaId)
75             .setMediationAdPlacement(new MediationAdPlacement.Builder()
76                 .setMuted(false)//是否静音
77                 .setVolume(0.6f)//设置音量
78                 .setBidNotify(true)//竞价结果通知
79                 .build())
80             .setOrientation(NGAdConstant.ORIENTATION_VERTICAL)
81             .build();
82
83         /** 2、创建NGAdBase对象 */
84         NGAdBase adBaseLoader = NGAdManagerHolder.get().createAdNative(this);
85     };

```

```

86         /** 3、创建加载、展示监听器 */
87         initListeners();
88
89         /** 4、加载广告 */
90         adBaseLoader.loadFullScreenVideoAd(adPlacement, this.mFullScreenV
91         ideoListener);
92
93         ToastUtil.show(MediationInterstitialFullActivity.this, "广告加载
94         中, 请稍后");
95     }
96
97     // 在加载成功后展示广告
98     private void showInterstitialFullAd() {
99         if (this.mNGFullScreenVideoAd == null) {
100             Log.d(Const.TAG, "请先加载广告或等待广告加载完毕后再调用show方法");
101             return;
102         }
103         /** 5、设置展示监听器, 展示广告 */
104         this.mNGFullScreenVideoAd.setFullScreenVideoAdInteractionListener
105         (this.mFullScreenVideoAdInteractionListener);
106
107         this.mNGFullScreenVideoAd.showFullScreenVideoAd(MediationIntersti
108         tialFullActivity.this);
109     }
110
111     private void initListeners() {
112         // 广告加载监听器
113         this.mFullScreenVideoListener = new NGAdBase.FullScreenVideoAdLis
114         tener() {
115             @Override
116             public void onError(int code, String message) {
117                 Log.d(Const.TAG, "InterstitialFull onError code = " + cod
118                 e + " msg = " + message);
119                 ToastUtil.show(MediationInterstitialFullActivity.this,
120                 "广告加载失败 onError code = " + code + " msg = " + message);
121             }
122
123             @Override
124             public void onFullScreenVideoAdLoad(NGFullScreenVideoAd ad) {
125                 Log.d(Const.TAG, "InterstitialFull onFullScreenVideoLoade
126                 d");
127                 mNGFullScreenVideoAd = ad;
128                 ToastUtil.show(MediationInterstitialFullActivity.this,
129                 "广告加载成功, 可展示");
130             }
131
132             @Override
133             public void onFullScreenVideoCached(NGFullScreenVideoAd ad) {

```



```

125         Log.d(Const.TAG, "InterstitialFull onFullScreenVideoCache
126         d");
127         mNGFullScreenVideoAd = ad;
128     }
129 };
130
131 // 广告展示监听器
132 this.mFullScreenVideoAdInteractionListener = new NGFullScreenVideo
133 oAd.FullScreenVideoAdInteractionListener() {
134
135     @Override
136     public void onAdShow() {
137         Log.d(Const.TAG, "InterstitialFull onAdShow");
138     }
139
140     @Override
141     public void onAdVideoBarClick() {
142         Log.d(Const.TAG, "InterstitialFull onAdVideoBarClick");
143     }
144
145     @Override
146     public void onAdClose() {
147         Log.d(Const.TAG, "InterstitialFull onAdClose");
148     }
149
150     @Override
151     public void onVideoComplete() {
152         Log.d(Const.TAG, "InterstitialFull onVideoComplete");
153     }
154
155     @Override
156     public void onSkippedVideo() {
157         Log.d(Const.TAG, "InterstitialFull onSkippedVideo");
158     }
159 };
160
161 @Override
162 protected void onDestroy() {
163     super.onDestroy();
164     /** 6、在onDestroy中销毁广告 */
165     if (mNGFullScreenVideoAd != null && mNGFullScreenVideoAd.getMedia
166 tionManager() != null) {
167         mNGFullScreenVideoAd.getMediationManager().destroy();
168     }
169 }

```

接入事例可参照demo内MediationSplashActivity类。

6. Banner广告

6.1 简介

Banner广告为一张图片的View，此View的宽高默认为match_parent，用户可以自行定义宽高或通过设置父布局的宽高的方式来限定Banner广告的大小（注：Banner广告的宽高请尽量与AdPlacement中设置的宽高比例接近，以避免过多非等比例的拉伸）。

Banner 的承载容器，需要与广告View尺寸的宽高比例接近，不要固定写死，不要有遮挡物，否则影响广告结算。

6.2 注意事项

- ① 在广告接入前需要明确[各ADN对应广告样式](#)情况，以确保正确完成广告接入，避免由于广告类型不匹配导致接入报错等情况的发生。
- ② 由于各广告平台对于包名校验规则不同，需确保在NGAd平台填写的包名符合各adn平台规范，避免由于包名校验不匹配导致的无广告返回情况的发生。
- ③ 广告请求时机，**建议在收到SDK初始化成功回调后发起广告请求**，当SDK初始化回调一直失败时，建议首先明确应用ID及广告位ID是否赋值正确、是否有多余空格、是否是网络不稳定导致的超时等，当排查后无法定位问题时，建议通过抓包将config字段下的加密内容提供过来，我方协助定位问题。
- ④ .onRenderSuccess方法内返回的view为null需要通过getAdView获取view。**需注意在展示广告时再通过getAdView获取view否则可能会导致素材浪费场景的发生。**

6.3 接入Banner广告

一、创建NGAdBase对象

```
1  NGAdBase adBaseLoader = NGAdManagerHolder.get().createAdNative(this);
```

二、创建广告请求AdPlacement

```
1  AdPlacement adPlacement = new AdPlacement.Builder()  
2      .setCodeId("聚合广告位ID")  
3      .setImageAcceptedSize(widhtPx, heightPx) // 单位px  
4      .build();
```

三、请求广告

```
1  adBaseLoader.loadBannerExpressAd(adPlacement, new NGAdBase.ExpressAdListen  
   er() {  
2      @Override  
3      public void onError(int i, String s) {  
4  
5      }  
6  
7      @Override  
8      public void onExpressAdLoad(List<NGExpressAd> list) {  
9          if (list != null && list.size() > 0) {  
10             Log.d(TAG, "banner load success");  
11             mBannerAd = list.get(0);  
12         } else {  
13             Log.d(TAG, "banner load success, but list is null");  
14         }  
15     }  
16 });
```

四、展示广告

```
1 private void showBannerAd() {
2     if (mBannerAd != null) {
3         mBannerAd.setExpressInteractionListener(new NGExpressAd.ExpressAdI
4             nteractionListener() {
5             @Override
6             public void onAdClicked(View view, int i) {
7                 Log.d(Const.TAG, "banner clicked");
8             }
9
10            @Override
11            public void onAdShow(View view, int i) {
12                Log.d(Const.TAG, "banner showed");
13            }
14        });
15        mBannerAd.setDislikeCallback(this, new NGAdDislike.DislikeInteract
16            ionCallback() {
17            @Override
18            public void onShow() { }
19
20            @Override
21            public void onSelected(int i, String s, boolean b) {
22                Log.d(Const.TAG, "banner closed");
23            }
24
25            @Override
26            public void onCancel() { }
27        });
28        View bannerView = mBannerAd.getExpressAdView();
29        if (bannerView != null && mBannerContainer != null) {
30            // 获取广告View的宽高，计算宽高比例，承载容器按宽高比例缩放设置。
31            // 注意：banner 的承载容器，需要与广告View尺寸的宽高比例接近，不要固定写
32            // 死，不要有遮挡物，否则影响广告结算。
33            // 建议使用 wrap_content（部分广告服务器返回的尺寸无法提前知道）
34            mBannerContainer.removeAllViews();
35            mBannerContainer.addView(bannerView);
36        } else {
37            Log.d(Const.TAG, "请先加载广告或等待广告加载完毕后再展示广告");
38        }
39    }
}
```

五、获取展示后广告信息事例

```

1  MediationBaseManager mediationManager = mBannerAd.getMediationManager();
2
3  printShowEcpmInfo(mediationManager)
4
5  /**
6   * 打印展示后Ecpm信息
7   */
8  public static void printShowEcpmInfo(MediationBaseManager mediationManager) {
9      if (mediationManager != null) {
10         MediationAdEcpmInfo showEcpm = mediationManager.getShowEcpm();
11         if (showEcpm != null) {
12             logEcpmInfo(showEcpm);
13         }
14     }
15 }
16
17 /**
18  * MediationAdEcpmInfo 字段参数如下:
19  */
20 public static void logEcpmInfo(MediationAdEcpmInfo item){
21     Log.d(Const.TAG, "EcpmInfo: \n" +
22         "adn名称 SdkName: " + item.getSdkName() + ",\n" +
23         "自定义adn名称 CustomSdkName: " + item.getCustomSdkName() + ",\n" +
24         "代码位Id PlacementID: " + item.getPlacementId() + ",\n" +
25         "广告价格 Ecpm: " + item.getEcpm() + ",\n" +
26         "广告竞价类型 ReqBiddingType: " + item.getReqBiddingType() + ",\n" +
27         "多阶底价标签 LevelTag: " + item.getLevelTag() + ",\n" +
28         "多阶底价标签解析失败原因 ErrorMsg: " + item.getErrorMsg() + ",\n" +
29         "adn请求Id RequestId: " + item.getRequestId() + ",\n" +
30         "广告类型 RitType: " + item.getRitType() + ",\n" +
31         "AB实验Id AbTestId: " + item.getAbTestId() + ",\n" +
32         "场景Id ScenarioId: " + item.getScenarioId() + ",\n" +
33         "流量分组Id SegmentId: " + item.getSegmentId() + ",\n" +
34         "流量分组渠道 Channel: " + item.getChannel() + ",\n" +
35         "流量分组子渠道 SubChannel: " + item.getSubChannel() + ",\n" +
36         "开发者传入的自定义数据 customData: " + item.getCustomData()
37     );
38 }

```

六、销毁广告

```

1  protected void onDestroy() {
2      super.onDestroy();
3      if (mBannerAd != null) {
4          mBannerAd.destroy();
5      }
6  }

```

6.4 接口说明

AdPlacement.Builder说明

方法名	说明
setCodeId(String codeId)	聚合广告位ID
setImageAcceptedSize(int width, int height)	设置宽高，单位px
setUserID(String userID)	设置userid
setMediationAdPlacement(MediationAdPlacement mediationAdPlacement)	设置聚合广告请求参数

MediationAdPlacement.Builder 说明

方法名	说明
setMuted(boolean var1)	设置静音
setVolume(float var1)	设置音量范围0~1；静音设置为0
setUseSurfaceView(boolean var1)	是否使用SurfaceView
setExtraObject(String var1, Object var2)	设置额外参数
setBidNotify(boolean var1)	bidding类型广告，竞价成功或者失败后是否通知对应的ADN
setScenarioId(String var1)	广告场景ID
boolean isAllowShowCloseBtn()	是否显示关闭按钮

NGExpressAd 说明

方法名	说明
View getViewExpressAdView()	得到原生模板广告view
void setExpressInteractionListener(ExpressAdInteractionListener expressAdInteractionListener)	广告交互回调
setExpressInteractionListener(AdInteractionListener adInteractionListener)	广告交互回调
void destroy()	销毁
void setDislikeCallback(Activity activity, TTAdDislike.DislikeInteractionCallback dislikeInteractionCallback)	设置dislike回调
Map<String, Object> getMediaExtraInfo()	广告额外信息
MediationNativeManager getMediationManager()	获取聚合广告信息

ExpressAdInteractionListener 说明

方法名	说明
void onAdClicked(View view, int type)	广告的点击回调, view、type无效
void onAdShow(View view, int type)	广告的展示回调, view、type无效

NGAdDislike.DislikeInteractionCallback 说明

void onShow()	dislike show
void onSelected(int position, String value, boolean enforce)	选择
void onCancel()	取消

MediationNativeManager 说明

方法名	说明
boolean isReady()	当前广告是否可以展示

MediationAdEcpmlInfo getShowEcpm()	为了确保数据的准确性，强烈建议在展示后获取展示广告の詳細信息，包括广告位类型–getRitType、流量分组ID–getSegmentId、AB实验分组ID–getABTestId、渠道名称–getChannel、子渠道名称–getSubChannel、场景ID–getScenariold、价格–、ADN平台–
List getAdLoadInfo()	load回调后获取广告加载信息
List getMultiBiddingEcpm()	load回调之后获取获取多阶底价，clientBidding serverBidding的广告信息
MediationAdEcpmlInfo getBestEcpm()	load回调之后获取最优广告价格信息
List getCacheList()	load回调之后，获取当前缓存池的全部信息

MediationAdEcpmlInfo说明

方法名	说明
Map<String, String> getCustomData()	自定义参数
String getSdkName()	获取SDK名称
String getCustomSdkName()	获取自定义ADN名称
String getSlotId()	获取代码位ID
String getLevelTag()	获取多阶底价标签
String getEcpm()	获取价格
int getReqBiddingType()	获取bidding类型
String getErrorMsg()	获取错误信息
String getRequestId()	获取请求唯一标识
String getRitType()	获取代码位类型
String getSegmentId()	获取流量分组ID
String getChannel()	获取渠道
String getSubChannel()	获取子渠道
String getAbTestId()	获取AB测试ID

String getScenarioId()	获取场景ID
------------------------	--------

6.5 示例代码

```
1  package cn.sirius.nga.activity;
2
3  import android.app.Activity;
4  import android.os.Bundle;
5  import android.util.Log;
6  import android.view.View;
7  import android.widget.FrameLayout;
8  import android.widget.TextView;
9
10 import androidx.annotation.Nullable;
11
12 import java.util.List;
13
14 import cn.sirius.nga.R;
15 import cn.sirius.nga.ad.NGAdBase;
16 import cn.sirius.nga.ad.NGAdBase.ExpressAdListener;
17 import cn.sirius.nga.ad.NGExpressAd;
18 import cn.sirius.nga.config.AdPlacement;
19 import cn.sirius.nga.config.NGAdManagerHolder;
20 import cn.sirius.nga.dislike.NGAdDislike;
21 import cn.sirius.nga.util.ToastUtil;
22 import cn.sirius.nga.util.UIUtil;
23
24 /**
25  * 融合demo, banner广告使用示例。更多功能参考接入文档。
26  * <p>
27  * 注意：每次加载的广告，只能展示一次
28  * <p>
29  * 接入步骤：
30  * 1、创建AdPlacement对象
31  * 2、创建NGAdBase对象
32  * 3、创建加载、展示监听器
33  * 4、加载广告
34  * 5、加载成功后，展示广告
35  * 6、在onDestroy中销毁广告
36  */
37 public class MediationBannerActivity extends Activity {
38
39     private static final String TAG = "MediationBannerActivity";
40
41     public String mMediaId; // 融合广告位
42
43     private NGExpressAd mBannerAd; // Banner广告对象
44
45     private ExpressAdListener mBannerListener; // 广告加载监听器
```

```

46
47     private NGExpressAd.ExpressAdInteractionListener mBannerInteractionLi
48 stener; // 广告展示监听器
49
50     private NGAdDislike.DislikeInteractionCallback mDislikeCallback; //
51 接受广告关闭回调
52
53     private FrameLayout mBannerContainer; // banner广告容器view
54
55     @Override
56     protected void onCreate(@Nullable Bundle savedInstanceState) {
57         super.onCreate(savedInstanceState);
58         setContentView(R.layout.mediation_activity_banner);
59
60         // 聚合广告位（在NG平台的广告位，注意不是adn的代码位）
61         this.mMediaId = getResources().getString(R.string.banner_media_i
62 d);
63         TextView tvMediationId = this.findViewById(R.id.tv_media_id);
64         tvMediationId.setText(getString(R.string.ad_mediation_id, this.mM
65 ediaId));
66
67         // banner广告容器
68         mBannerContainer = findViewById(R.id.banner_container);
69
70         // 广告加载
71         findViewById(R.id.bt_load).setOnClickListener(new View.OnClickListener() {
72             @Override
73             public void onClick(View v) {
74                 loadBannerAd();
75             }
76         });
77
78         // 广告展示
79         findViewById(R.id.bt_show).setOnClickListener(new View.OnClickListener() {
80             @Override
81             public void onClick(View v) {
82                 showBannerAd();
83             }
84         });
85
86     }
87
88     private void loadBannerAd() {
89         /** 1、创建AdPlacement对象 */
90         AdPlacement adPlacement = new AdPlacement.Builder()
91             .setCodeId(this.mMediaId)

```

```

88         .setImageAcceptedSize(UIUtil.dp2px(this, 320f), UIUtil.dp
89         2px(this, 150f)) // 单位px
90         .build();
91
92         /** 2、创建NGAdBase对象 */
93         NGAdBase adBaseLoader = NGAdManagerHolder.get().createAdNative(th
94         is);
95
96         /** 3、创建加载、展示监听器 */
97         initListeners();
98
99         /** 4、加载广告 */
100        adBaseLoader.loadBannerExpressAd(adPlacement, mBannerListener);
101
102        ToastUtil.show(MediationBannerActivity.this, "广告加载中, 请稍后");
103    }
104
105    /**
106     * 5、广告加载成功后, 设置监听器, 展示广告
107     */
108    private void showBannerAd() {
109        if (mBannerAd != null) {
110            mBannerAd.setExpressInteractionListener(mBannerInteractionLis
111            tener);
112            mBannerAd.setDislikeCallback(this, mDislikeCallback);
113            mBannerAd.uploadDislikeEvent("mediation_dislike_event");
114            /** 注意: 使用融合功能时, load成功后可直接调用getExpressAdView获取广
115            告view展示, 而无需调用render等onRenderSuccess后再 */
116
117            View bannerView = mBannerAd.getExpressAdView();
118            if (bannerView != null && mBannerContainer != null) {
119                mBannerContainer.removeAllViews();
120                mBannerContainer.addView(bannerView);
121            }
122        } else {
123            Log.d(TAG, "请先加载广告或等待广告加载完毕后再展示广告");
124        }
125    }
126
127    private void initListeners() {
128        // 广告加载监听器
129        this.mBannerListener = new ExpressAdListener() {
130            @Override
131            public void onError(int i, String s) {
132                Log.d(TAG, "banner load fail: errCode: " + i + ", errMs
133                g: " + s);
134                ToastUtil.show(MediationBannerActivity.this, "广告加载失败
135                onError code = " + i + " msg = " + s);

```

```

130         }
131
132         @Override
133         public void onExpressAdLoad(List<NGExpressAd> list) {
134             if (list != null && list.size() > 0) {
135                 Log.d(TAG, "banner load success");
136                 mBannerAd = list.get(0);
137                 ToastUtil.show(MediationBannerActivity.this, "广告加载
成功,可展示");
138             } else {
139                 Log.d(TAG, "banner load success, but list is null");
140                 ToastUtil.show(MediationBannerActivity.this, "广告加载
成功,但list为空");
141             }
142         }
143     };
144     // 广告展示监听器
145     this.mBannerInteractionListener = new NGExpressAd.ExpressAdIntera
ctionListener() {
146         @Override
147         public void onAdClicked(View view, int i) {
148             Log.d(TAG, "banner clicked");
149         }
150
151         @Override
152         public void onAdShow(View view, int i) {
153             Log.d(TAG, "banner showed");
154         }
155     };
156
157     // dislike监听器, 广告关闭时会回调onSelected
158     this.mDislikeCallback = new NGAdDislike.DislikeInteractionCallbac
k() {
159         @Override
160         public void onShow() {
161         }
162
163         @Override
164         public void onSelected(int i, String s, boolean b) {
165             if (mBannerContainer != null) {
166                 mBannerContainer.removeAllViews();
167             }
168             Log.d(TAG, "banner closed");
169         }
170
171         @Override
172         public void onCancel() {
173         }

```

```

174         };
175     }
176
177     @Override
178     protected void onDestroy() {
179         super.onDestroy();
180         /** 6、在onDestroy中销毁广告 */
181         if (mBannerAd != null) {
182             mBannerAd.destroy();
183         }
184     }
}

```

接入事例可参照demo内MediationSplashActivity类。

7. 融合SDK版本接入注意事项

7.1 常见问题

Q：接入融合SDK优点是什么？

A：可以缩短开发者接入成本，仅引入一个融合SDK即可完成相关接入。

Q：融合SDK初始化是否支持多次初始化场景及SDK初始化注意事项是什么？

A：默认仅支持初始化SDK一次，多次初始化以第一次初始化为准。

Q：无网络如何处理

A：部分ADN平台要求实时联网，在无网络时，使用缓存广告可能影响广告效果的统计与结算。

因此建议CP在发起请求广告时，先检查是否有网络或引导用户联网。

Q：一个完整的广告链路是如何的？

A：参数-> 请求load->展示show->关闭close；show** 前一定要 load**，这个链路与是否有缓存无关。

Q：为什么不建议游戏自建缓存？

A：

1. 游戏不需要在 **Listener 里 show 或 close 通过 NGAdLoadType.PRELOAD 再 load 下一次的广告。
2. 融合SDK会根据时机判断进行缓存，游戏只要每次按【单个广告链路】调用就行。
3. **核心原因：**展示前广告时，还存在竞价逻辑，只要网络畅通，竞价逻辑生效，广告服务器可能返回最新的高价广告，而不使用缓存广告。
4. 因为实时竞价逻辑的存在，缓存广告不一定会被展示，但是多次缓存，会在竞价中浪费掉，导致展示率过低，进而影响到 eCPM。
- 5.

Q：广告的请求时长是多少？

A：实时请求一次广告，正常联网下，大概1.5s；存在有效缓存时大概是60%的时间。除了异常网络，实际从请求到可展示广告的耗时是很短，包括竞价的时间。

8. SDK错误码问题定位总结思路及错误码解析

在接入过程中或者线上用户最常反馈的就是广告无填充问题，当遇到这类问题时，建议：

- ① 及时查看相关错误码埋点情况或错误日志。
- ② 当通过①无法定位问题时，可通过抓包或日志检索关键字TMe或mediation明确ADN具体错误码。
- ③ 支持通过NGxxxAd.getAdLoadInfo(), 获取加载失败的ADN错误信息，根据LoadInfoList详情发现并定位问题。

8.1 广告SDK高频错误码解析

错误码	说明	排查建议
900400	网络不可用，无法连接到服务器	提示用户联网，特别是在激励广告场景。
3003	网络不可用，无法连接到服务器	提示用户联网，特别是在激励广告场景。
20005	所有层数都没有广告返回	<ol style="list-style-type: none"> 1. 建议增加兜底代码位； 2. 如果已配置兜底代码位，建议查看兜底代码位ID的诊断分析模块进行定位 3. 通过抓包来明确广告无填充原因；

10010	所有代码位还未请求完成就触发了总超时时间	建议延长瀑布流总超时时长或者缩短层超时并添加兜底代码位
41005	代表了当前的context传入为空	一般Mintegral会触发此场景，Mintegral要求必须传入的是activity
40040	暂无配置信息	当发生此场景时，建议优先检查应用ID是否赋值正确、是否有多余空格等、当无法定位到相关问题时，建议通过抓包查看config文件夹下的请求信息反馈给对应的技术支持同学
40047	同一个广告对象多次加载广告导致的	需注意每次加载广告时需重新创建新的广告对象，不能使用同一个广告对象多次请求
40048	代表了同一个广告对象多次展示导致的	需注意已经展示过的广告对象不能重复使用，需要使用新的广告对象
40051	ADN版本错误引起的加载失败	请参照对接文档SDK接入模块，对照接入的ADN版本
40052	无可用广告引发的展示失败场景	开发者可在展示失败回调确认该错误码情况，如果是穿山甲广告可优先检查是否是调整了系统时间导致的
40200	请求方法不匹配引发的无广告填充	建议通过各ADN对应聚合广告样式明确接入的广告类型情况，参照对接文档各广告类型模块说明明确对应广告请求方法，以确保正确完成广告接入，避免由于广告类型不匹配导致的无广告填充
41044	已调用过destroy方法	请勿重复调用destroy方法
44404	网络环境异常，当前无网络	建议检查网络环境是否正常，重新发起请求或者自行判断网络状态决定是否发起广告请求
44405	网络超时	建议检查网络环境是否正常，更换网络重新发起请求
44406	广告位ID不合法	当发生此场景时建议优先检查广告位ID的状态是否正常

当开启融合SDK功能时，如果发现以下错误码可按照处理建议进行排查

错误码	说明	排查方向
840027	开屏广告自定义兜底参数不正确	建议检查设置的自定义兜底参数是否准确
840028	开屏广告开发者自定义兜底中CSJ应用ID与SDK初始化的应用ID不一致	<ul style="list-style-type: none">如若设置开屏广告的自定义兜底代码位是CSJ，请确保设置的应用ID与SDK初始化保持一致由于自定义兜底代码位建议创建非聚合属性代码位
840031	聚合代码位对应的广告类型和当前广告类型不一致	建议在ADN后台明确创建的广告类型，需确保在穿山甲媒体平台匹配设置
840040	暂无配置信息	<p>当发生此场景时，建议优先检查应用ID是否赋值正确，是否有多余空格等，当无法定位到相关问题时，建议通过</p> <p>抓包查看config文件夹下的请求信息反馈给对应的技术支持同学</p>

8.2 ADN广告常见错误码

8.2.1 GDT

常见错误码	说明	处理建议
4001	初始化错误, 包括广告位为空、App ID为空、Context/Activity为空	在广告位ID和应用ID无误的前提下，请检查展示广告的Context/Activity是否为空
4007	当前设备或版本不支持	请参照本文档中相应广告类型对设备或版本的限制
5002	视频素材下载错误	建议稍后重试，如果重试仍然有错误，请反馈给优量汇运营
5004	未匹配到合适的广告	此情况下禁止多次重试请求广告，否则可能影响系统对您流量的评价从而影响变现效果

5006	包名校验错误，当前 App 的包名和优量汇官网注册媒体时填写的包名不一致，因此无广告返回	请检查接入优量汇 SDK 的 App 包名是否和注册时填写的一致，否则将影响您的收益
5010	广告样式校验失败，请检查广告位与接口使用是否一致	目前后台开放权限的是平台模板广告，请根据平台模板广告的接入文档进行接入，如果是按照自渲染广告接入文档进行接入的话，也会报错107034
2001	初始化错误	开发者自查参数问题
2003	SDK未初始化	根据 Logcat 中的错误信息提示修改嵌入代码
3003	网络不可用	网络不可用，无法连接到服务器
5013	广告请求过于频繁	请求过于频繁，服务器繁忙时会返回该错误码，请检测与控制请求频率
4011	开屏广告拉取超时，请自查开屏广告的拉取超时时间设置是否过短	根据 Logcat 中的错误信息提示修改嵌入代码
5010	广告样式校验失败，请检查广告位与接口使用是否一致	目前后台开放权限的是平台模板广告，请根据平台模板广告的接入文档进行接入，如果是按照自渲染广告接入文档进行接入的话，也会报错107034
2001	初始化错误	开发者自查参数问题
5022	模板激励视频渲染失败	建议稍后重试，如果重试仍然有错误，请反馈给优量汇运营
4002	请检查 Manifest 文件中的 Activity/Service/Permission 的声明是否正确以及声明的权限是否都已授予	请检查 Manifest 文件中的 Activity/Service/Permission 的声明是否正确以及声明的权限是否都已授予
5012	广告数据过期	部分广告（如激励视频）可以预拉取，拉取广告后广告数据会有存在一个过期时间，当开发者调用展示广告的接口但此时当前时间已经超过过期时间时会返回此错误码

102006	没有匹配到合适的广告。	禁止重试，否则可能触发系统策略导致流量收益下降。
100133	广告位填写错误，或广告位状态处于关闭状态；如是新建广告位并立刻请求了广告，则会由于系统索引不到而导致请求失败	请求中包含无效的广告位，请确认对应的广告位ID的状态；新建广告位，请在新建30分钟后请求广告
100135	广告位状态冻结	请检查广告位状态。
109502	请求过于频繁或返回未曝光，且产生的收入低，触发平台出于成本考虑的填充限制	无法识别当前的网络环境，视频广告需要在WIFI和4G网络环境下请求，其他环境会返回这个报错，建议切换当前网络到WIFI或者4G再次尝试一下 无合适广告资源返回，请控制广告请求频次

更多相关错误信息请查看：[GDT SDK错误码](#)

8.2.2 KS

常见错误	说明
40001	没有网络
40002	数据解析失败
40003	广告数据为空
100001	参数有误
310001	appld未注册
310002	appld无效
310003	appld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	appld对应账号无效
320003	appld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

更多的KS错误信息请查看：KS错误码

8.3 问题反馈模版

为了确保定位问题更高效，较少双方沟通成本，建议反馈问题时按照如下模版提供对应信息

- ① SDK版本（必要）
- ② ADN版本（必要）
- ③ 手机机型及系统版本（必要）
- ④ 详细的问题描述（必要）
- ⑤ 广告类型（模版/自渲染）（必要）
- ⑥ 应用ID/广告位ID/代码位ID（必要）
- ⑦ 是否必现（可选）
- ⑧ 操作录屏或截图（最好是录屏）（可选，展示异常时必要）
- ⑨ 素材抓包（展示异常场景，可选）
- ⑩ 堆栈信息（最好提供文本信息；截图不建议，崩溃场景必要）
- ⑪ debug包（可选，方便问题排查及场景复现）

8.4 抓取排查日志

接入过程中遇到问题需要技术协助排查，可以在要求协助排查时同时给予抓取的本地日志，便于快速排查解决

打开日志方式一

初始化时 `cn.sirius.nga.config.NGAdConfig.Builder#setDebug` 设置为true，打开日志

打开日志方式二

如果是release包，需要动态打开，`${applicationId}` 替换成当前包名

```
1 adb shell content insert --uri content://${applicationId}.NGAdContentProvider --bind openLog:b:true
```

抓取日志

过滤NGAdSdk，抓取所有日志

PS：为保证日志全，需要从冷起开始抓取