广告SDK接入说明文档

1. SDK简介

1.1 广告类型

提供5种广告：横幅广告、视频广告、开屏广告、插屏广告、原生广告

注：具体广告效果参照广告sdk的Demo程序运行效果。

1.2 广告SDK版本升级

针对广告SDK版本升级，参照以下步骤：

1. 从官网下载最新版本(https://www.yousuode.cn/download/sdk) 解压获取aar包（在ngad-sdk-all-*.zip\04-依赖库\aar接入方式\ngad-sdk-release-*.aar 里），对比版本号，如果已经是最新版本则无需替换；如果不是最新版本则需要**替换更新aar包**。

2. Android Studio工程，比较简单。**删除本地工程的旧版aar包，替换最新版本**；

3. Eclipse 工程，**解压aar包分别拷贝到旧版内容的目录包括：assets、res、libs、libs/armeabi，覆盖替换相关的文件**

注意：旧版本sdk的多余资源文件需要删除。

1.3. ChangeLog

版本号	接入变更
2.21.2	1、合规升级，增加广告合规相关说明 2、优化广告模板，提升广告填充率 3、优化广告加载超时问题，提升用户体验
2.20.7	1、更新广告源
2.19.8	1、更新广告源

v2.17.2	1、更新广告源
v2.16.3	1、日志采集优化
v2.15.12	1、广告sdk隐私合规优化2、优化权限采集频率3、更新广告源
v2.14.0	1、广告sdk隐私合规优化2、优化权限采集频率3、更新广告源
v2.13.0	1、日志、网络优化2、更新检测工具
v2.12.6	1、修复开屏广告部分场景崩溃问题2、AndroidManifest.xml中移除不必要的声明
v2.12.0	1、更新广告源
v2.11.0	1、增加日志，提升问题排查能力
v2.10.0	1、支持横幅广告填充信息流广告
v2.9.0	1、新增广告源，提高广告填充率2、部分场景问题修复
v2.8.41	1、原生广告部分场景问题修复2、开屏广告部分场景问题修复
v2.8.30	1、加载广告流程优化2、增加日志，提升问题排查能力
v2.8.0	1、更新厂商sdk2、更新头条sdk版本为3.3.0.13、更新广点通sdk版本为4.290.11604、优化自有广告
v2.6.42	1、修复获取oaid在部分手机上会崩溃的问题
v2.6.41	1、修复自有广告在8.0以上手机会崩溃问题2、修复头条sdk激励视频填充率低的问题

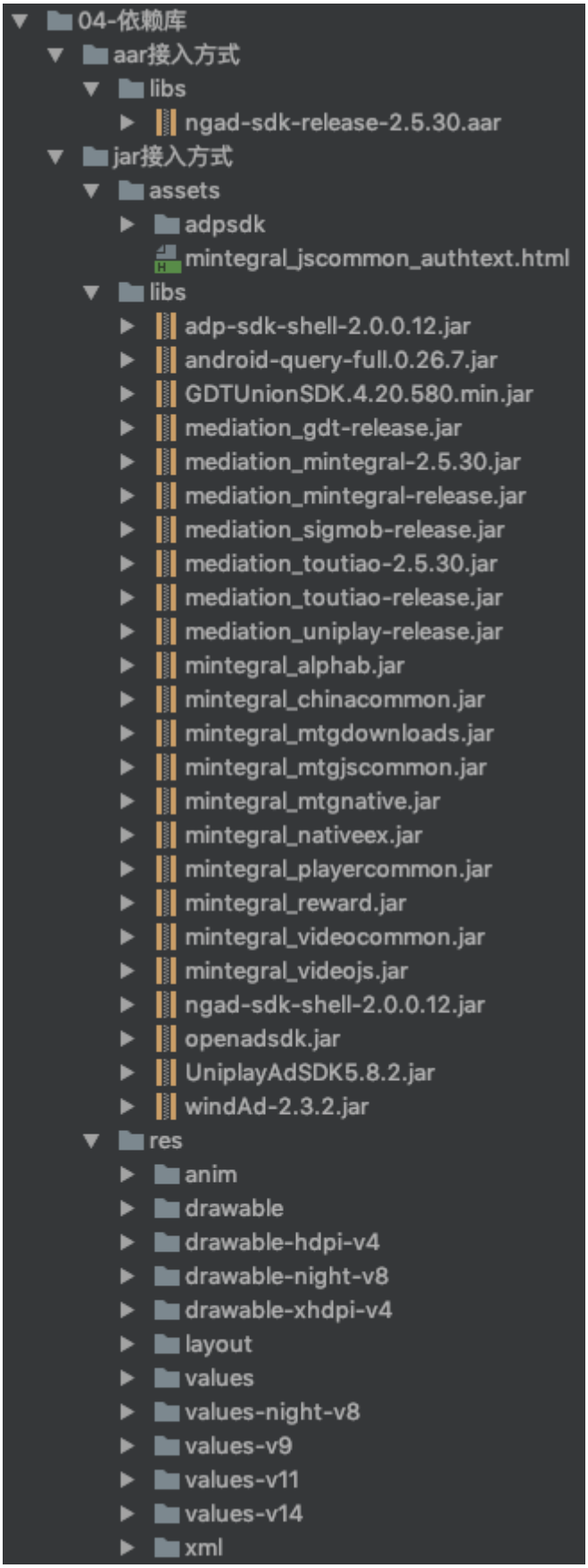
v2.6.40	1、新增魅族sdk_2.5.0.12、更新huawei_13.4.28.305,oppo_3.2.9
v2.6.30	1、支持广点通插屏、banner2.0接口2、修复了一些已知bug
v2.6.20	1、更新头条sdk_2.7.5.2, 广点通_4.150.1020, mintegral_10.2.12、广点通banner，插屏支持2.0接口
v2.5.70	1、更新vivo_3.2.0.1, sigmob_2.13.1，兼容android 10系统2、新增Xiaomi_2.4.3，Huawei渠道广告支持
v2.5.44	1、AndroidManifest.xml新增FileProvider组件声明变更2、更新UniplayAdSdk_6.0.2.jar，删除UniplayAdSdk_5.8.2.jar
v2.5.30	1、解决包冲突问题2、AndroidManifest.xml权限和组件声明变更3、去除依赖库gamesdk-framework-shellbase-7.1.5.61.aar(jar)、alisdk-utdid-20160516.jar、adp_sdk_shell_base-1.0.6.0.jar、assets/ucgamesdk/lib(如果集成游戏SDK，则不用去除)4、新增assets/adpsdk/lib、adp-sdk-shell-2.0.0.12.jar
v2.4.10	1、支持Android9运行，Android8适配2、AndroidManifest.xml权限和组件声明变更3、去除依赖库lmjoy_video-*.jar4、依赖依赖库support-v4要求升级到26+版本
v2.3.50	1、变更所有错误码
v2.2.90	1、添加检测工具（05-CheckTool）
v2.2.60	1、新增了原生广告接口

v2.2.50	1、AndroidManifest.xml的权限和组件声明2、Proguard文件3、libs目录引用的AAR或JAR文件版本号及数量4、res目录文件**5、新增targetVersion>=24时，FileProvider路径配置**
v2.2.30	1、AndroidManifest.xml的权限和组件声明2、Proguard文件3、libs目录引用的AAR或JAR文件版本号及数量4、res目录文件5、添加广告加载异常处理说明：**单次启动应用，如果连续5次加载广告失败，不再发起请求。**

2. SDK嵌入

2.1 步骤1：添加SDK到工程中

解压开ngad-sdk-all-*.zip文件，在04-依赖库目录下可以看到NGASDK提供的两种接入方式的依赖库。


04-依赖库
aar接入方式
libs
ngad-sdk-release-2.5.30.aar
jar接入方式
assets
adpsdk
mintegral_jscommon_authtext.html
libs
adp-sdk-shell-2.0.0.12.jar
android-query-full.0.26.7.jar
GDTUnionSDK.4.20.580.min.jar
mediation_gdt-release.jar
mediation_mintegral-2.5.30.jar
mediation_mintegral-release.jar
mediation_sigmob-release.jar
mediation_toutiao-2.5.30.jar
mediation_toutiao-release.jar
mediation_uniplay-release.jar
mintegral_alphab.jar
mintegral_chinacommon.jar
mintegral_mtgdownloads.jar
mintegral_mtgjscommon.jar
mintegral_mtgnative.jar
mintegral_nativeex.jar
mintegral_playercommon.jar
mintegral_reward.jar
mintegral_videocommon.jar
mintegral_videojs.jar
ngad-sdk-shell-2.0.0.12.jar
openadsdk.jar
UniplayAdSDK5.8.2.jar
windAd-2.3.2.jar
res
anim
drawable
drawable-hdpi-v4
drawable-night-v8
drawable-xhdpi-v4
layout
values
values-night-v8
values-v9
values-v11
values-v14
xml

【Android Studio工程引入】（推荐）

1. **把“aar接入方式”目录下的libs目录内容拷贝到游戏的Android项目工程对应的libs目录中**
2. **添加Gradle脚本**

```
//指定本地aar文件的存放目录，libs为build.gradle的相对地址
repositories {
    flatDir {
        dirs'libs'
    }
}

dependencies {
    //集成工程中libs目录下的jar文件（alisdk-utdid-20160516.jar）
    compile fileTree(include: ['*.jar'], dir: 'libs')
    compile 'com.android.support:support-v4:23.1.1'

    //集成工程中libs目录下的aar文件（ngad-sdk-release-*.aar）
    fileTree(dir: 'libs', include: ['*.aar']).each { file ->
        compile(name: file.name.lastIndexOf('.').with {
            it != -1 ? file.name[0..<it] : file.name
        }, ext: 'aar')
    }
}
```

3. **废弃资源删除**

- 检查libs目录或jni目录，删除旧版在libs目录中提供的 **libffmpeg.so**、 **libinitHelper.so**、**libng_util.so**、 **librotate.so**、 **libu3player.so**、 **libun7z.so**
- 去除lmjoy_video-*.jar、gamesdk-framework-shellbase-7.1.5.61.aar(jar)、alisdk-utdid-20160516.jar、adp_sdk_shell_base-1.0.6.0.jar、assets/ucgamesdk/lib(如果集成游戏SDK，则不用去除)**

4. **解决冲突**

1. **如果已经集成九游的游戏SDK8.x版本，则广告SDK版本需升级至2.5.30以上版本**
2. **其他冲突解决方案就联系技术支撑人员**

【Eclipse工程引入】（不推荐）

1. **把“jar接入方式”目录下的libs、assets、res目录内容拷贝到游戏的Android项目工程中对应的libs、assets、res目录中**
2. **引入android-support-v4.jar （可以从网上获取，若已存在则忽略）**
3. **废弃资源删除**a) 检查libs目录或jni目录，删除旧版在libs目录中提供的 **libffmpeg.so、libinitHelper.so、 libng_util.so、 librotate.so、 libu3player.so、 libun7z.so、 lmjoy_video-*.jar*
4. **解决冲突**

1. **如果已经集成九游的游戏SDK8.x版本，则广告SDK版本需升级至2.5.30以上版本**
2. **通常通过jar包方式接入的厂商，经常会漏添加或更新res、assets目录下的资源、混淆规则，所以在尽可能保持**
3. **其他冲突解决方案就联系技术支撑人员**

2.2 步骤2：修改AndroidManifest.xml文件

添加权限声明：

```
  <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.REQUEST_INSTALL_PACKAGES"
/>    <!-- 必要权限，权限会用在下载类广告安装应用时使用 -->
    <!-- 更多的权限配置以输出的AndroidManifest.xml定义为准 -->
```

如果您打包App时的targetSdkVersion >= 23：请在先获取到SDK要求的所有权限，然后再调用SDK的广告接口。否则SDK将无法工作，我们建议您在App启动时就去获取SDK需要的权限，Demo工程中也提供了基本的权限处理示例代码供开发者参考。

如果您打包App时的targetSdkVersion >= 24：除了需要处理好权限申请以外，还需要处理好文件访问的兼容性。

添加SDK组件声明(更多组件配置以输出的AndroidManifest-must.xml定义为准)

```
        <!-- NGASDK START -->

        <!-- 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
${applicationId}来设置这个值 -->
        <provider
            android:name="android.support.v4.content.FileProvider"
            android:authorities="${applicationId}.fileprovider"
            android:exported="false"
            android:grantUriPermissions="true">
            <meta-data
                android:name="android.support.FILE_PROVIDER_PATHS"
                android:resource="@xml/adp_file_path" />
        </provider>
        <!-- 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
${applicationId}来设置这个值 -->
        <provider
            android:name="com.mintegral.msdk.MintegralFileProvider"
            android:authorities="${applicationId}.provider"
            android:exported="false"
            android:grantUriPermissions="true">
            <meta-data
                android:name="android.support.FILE_PROVIDER_PATHS"
                android:resource="@xml/adp_file_path"/>
        </provider>
        <provider
            android:name="com.uniplay.adsdk.UniPlayFileProvider"
            android:authorities="${applicationId}.uniplay.fileprovider"
            android:exported="false"
            android:grantUriPermissions="true">
            <meta-data
                android:name="android.support.FILE_PROVIDER_PATHS"
                android:resource="@xml/adp_file_path" />
        </provider>

        <activity
            android:name="cn.sirius.nga.activity.ProxyActivity"
            android:configChanges="keyboardHidden|orientation|screenSize"
            android:theme="@android:style/Theme.Translucent.NoTitleBar"
            android:windowSoftInputMode="adjustResize">
        </activity>
```

```
        <!-- 更多组件配置以输出的AndroidManifest-must.xml定义为准 -->

<!-- NGASDK END -->
```

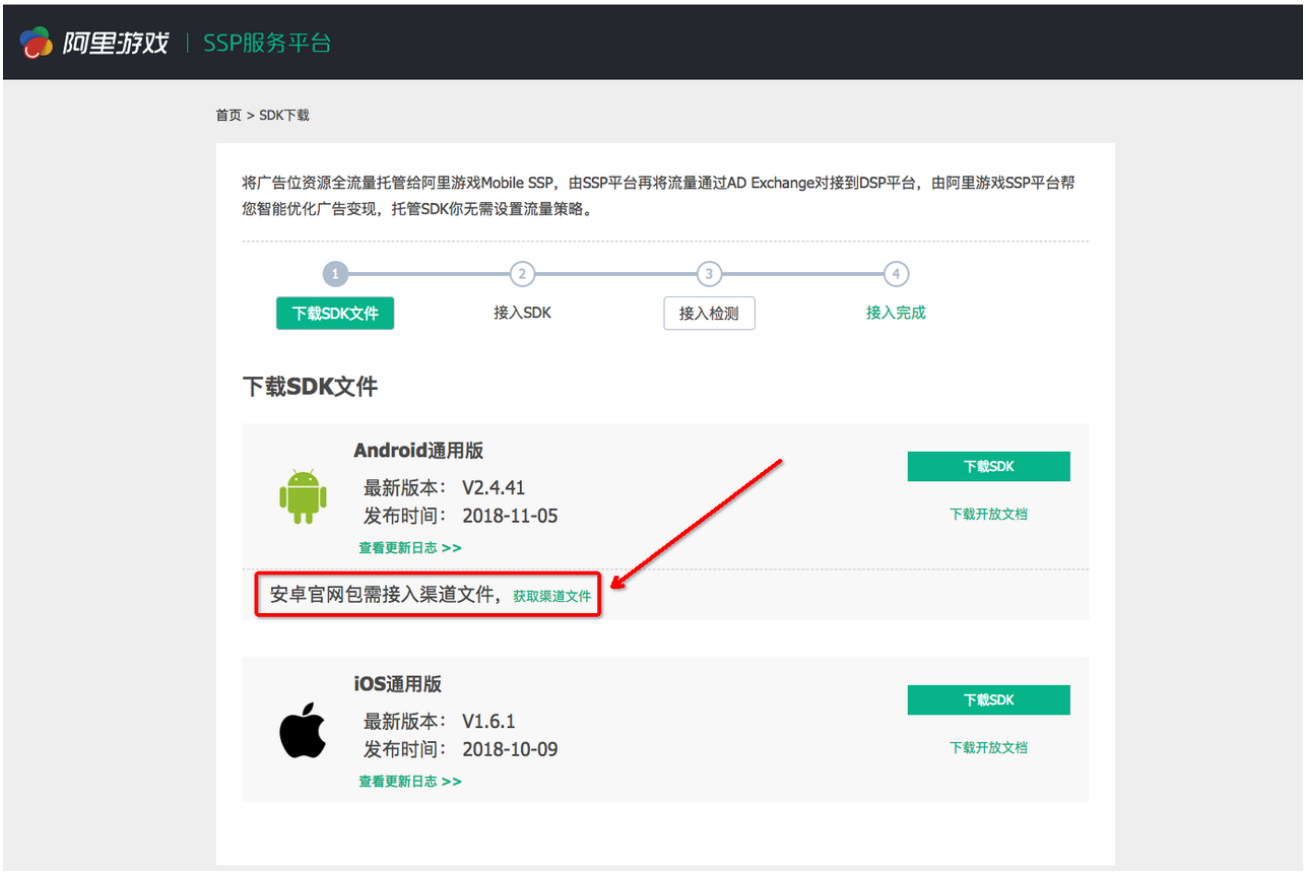
注意：这里的 ${applicationId} 不是广告的appId，而是Android工程（接入游戏包名）applicationId，比如 com.brianbaek.popstar

在项目结构下的 `res` 目录下添加一个 `xml` 文件夹，拷贝 `adp_file_path.xml` 文件到xml文件夹中，文件内容如下：

```
<paths xmlns:android="http://schemas.android.com/apk/res/android">
    <!-- 这个下载路径不可以修改，必须是GDTDOWNLOAD -->
    <external-path name="gdt_sdk_download_path" path="GDTDOWNLOAD" />
    <root-path name="download" path="" />
    <external-path name="external_files" path="."/>
    <external-files-path name="external_files_path" path="Download" />
</paths>
```

2.3 步骤3：非九游渠道加入渠道文件（九游渠道包可省略该步骤）

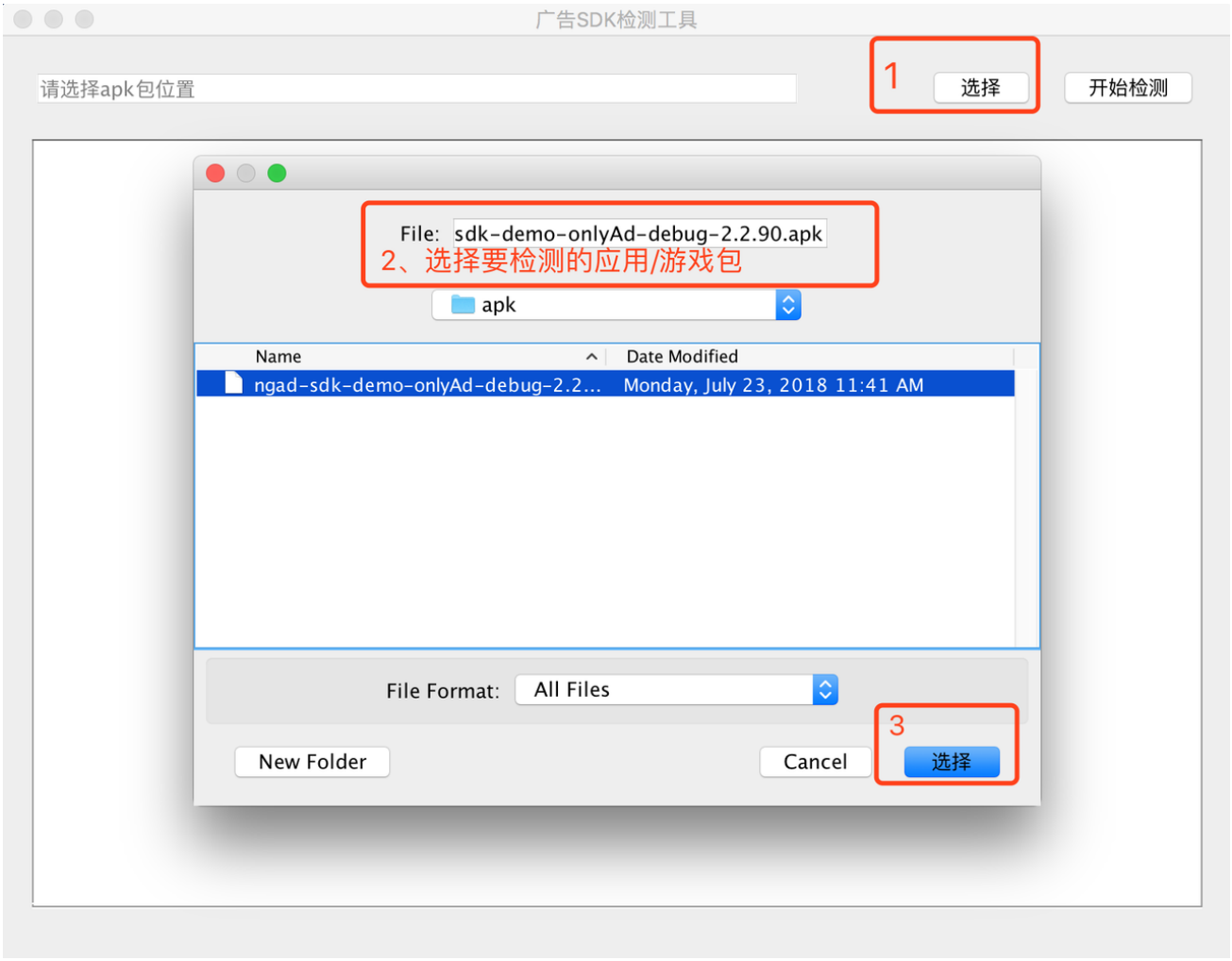
1、根据包的具体渠道下载相对应的渠道文件： http://www.yousuode.cn/download/sdk

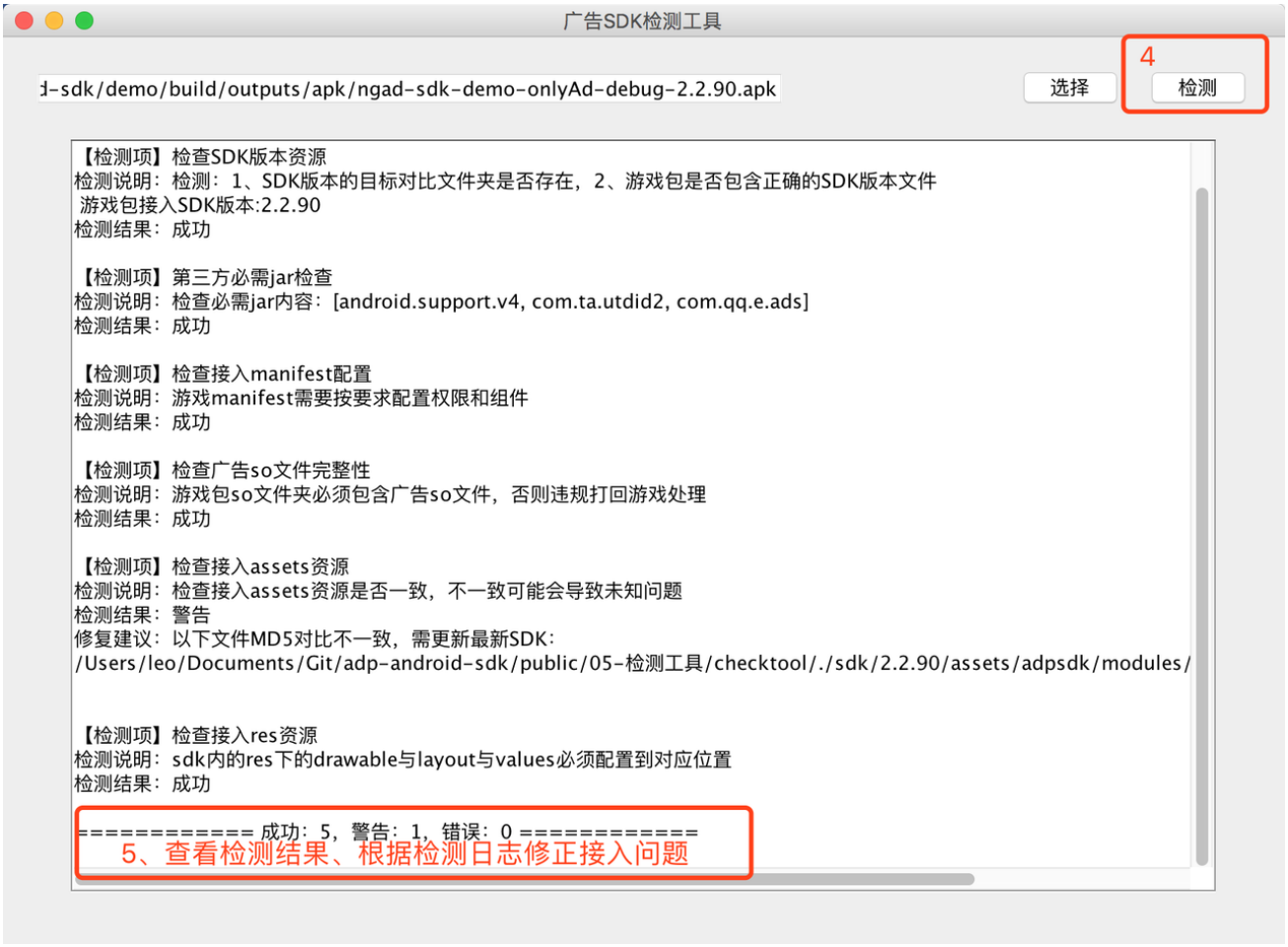


2、解压出文件 UCGameConfig.ini，放入游戏包的assets文件夹下

2.4 步骤4：使用检测工具验证是否存在接入问题

1. 解压checktool.zip到任意目录下
2. 在该目录下，mac和linux系统运行 start.sh，Windows运行 start.bat
3. 按步骤执行，根据检测结果修正接入问题


广告SDK检测工具
请选择apk包位置
1
选择
开始检测
File: sdk-demo-onlyAd-debug-2.2.90.apk
2、选择要检测的应用/游戏包
apk
Name
Date Modified
ngad-sdk-demo-onlyAd-debug-2.2...
Monday, July 23, 2018 11:41 AM
File Format:
All Files
3
New Folder
Cancel
选择



3. 接入代码

3.1 初始化SDK

在Application#onCreate中进行SDK的初始化。

NGASDK初始化是使用SDK所提供功能可以执行的前提，否则会导致广告加载失败。游戏在应用启动时Application的onCreate方法中插入广告SDK初始化逻辑。

```
@Override
public void onCreate() {
    super.onCreate();
    // 同意隐私政策后调用初始化
    initAdSdk();
}

protected void initAdSdk() {
```

```
        initSdk(this, new NGASDK.InitCallback() {
            @Override
            public void success() {
                //NGASDK init success, and can try to show splash ad.
                Log.d("NGAdSdk", "广告SDK初始化成功");
            }
            @Override
            public void fail(Throwable throwable) {
                Log.e("NGAdSdk", "广告SDK初始化异常: " +
Log.getStackTraceString(throwable));
            }
        });
    }

    protected void initSdk(Context context, final NGASDK.InitCallback
initCallback) {
        Log.d(TAG, MediaAdConfig.toStringFormat());
        NGASDK ngasdk = NGASDKFactory.getNGASDK();
        ngasdk.init(context, new AdConfig.Builder()
                .setAppId(MediaAdConfig.appId)
                .setDebug(true)
                .showNotification(true)
                .supportPersonalizedAd(true)
                .build(), initCallback);
    }
```

NGASDK主要API

public class NGASDKFactory

限定符和类型	方法和说明
static NGASDK	getNGASDK()

public interface NGASDK

限定符和类型	方法和说明

<T extends NGAProperties>	loadAd()加载广告请求注意：为了提高 率以及曝光量，建议重新加载广告时候 方法，重新请求新的广告内容。

3.2 横幅广告

广告接入代码示例

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_ad_control);
}

private NGABannerController mController;
private NGABannerProperties mProperties;
private ViewManager mWindowManager;
private RelativeLayout mBannerView;

//注意：请在Activity成员变量保存，使用匿名内部类可能导致回收
NGABannerListener mAdListener = new NGABannerListener() {
    @Override
    public void onRequestAd() {
        ToastUtil.show(TAG, "onRequestAd");
    }

    @Override
    public <T extends NGAdController> void onReadyAd(T controller) {
        mController = (NGABannerController) controller;
        ToastUtil.show(TAG, "onReadyAd");
    }

    @Override
    public void onShowAd() {
        ToastUtil.show(TAG, "onShowAd");
    }

    @Override
    public void onCloseAd() {
```

```
        //广告关闭之后mController置null，鼓励加载广告重新调用loadAd，提高广告填充率
        mController = null;
        ToastUtil.show(TAG, "onCloseAd");
        mBannerView.setVisibility(View.GONE);
    }

    @Override
    public void onErrorAd(int code, String message) {
        ToastUtil.show(TAG, "onErrorAd errorCode:" + code + ", message:" +
message);
    }

    @Override
    public void onClickAd() {
        ToastUtil.show(TAG, "onClickAd");
    }
};

//为了提高广告的填充率以及曝光量，建议重新加载广告时候重新调用此方法，重新请求新的广告
内容
private void loadAd(Activity activity) {
    if (mBannerView != null && mBannerView.getParent() != null) {
        mWindowManager.removeView(mBannerView);
    }
    mBannerView = new RelativeLayout(activity);
    WindowManager.LayoutParams params = new WindowManager.LayoutParams();
    params.width = ViewGroup.LayoutParams.WRAP_CONTENT;
    params.height = ViewGroup.LayoutParams.WRAP_CONTENT;
    params.gravity = Gravity.BOTTOM | Gravity.CENTER;
    params.flags = WindowManager.LayoutParams.FLAG_NOT_FOCUSABLE;
    mWindowManager = (WindowManager)
activity.getSystemService(Context.WINDOW_SERVICE);
    mWindowManager.addView(mBannerView, params);

    mProperties = new NGABannerProperties(activity, AdConfig.appId,
AdConfig.bannerPosId, mBannerView);
    mProperties.setListener(mAdListener);
    NGASDK ngasdk = NGASDKFactory.getNGASDK();
    ngasdk.loadAd(mProperties);

    // 若需要默认横幅广告不展示
    mBannerView.setVisibility(View.GONE);
}

private void destroyAd(Activity activity) {
```

```
    if (mWindowManager != null) {
        mWindowManager.removeView(mBannerView);
        mWindowManager = null;
    }
    if (mController != null) {
        mController.closeAd();
        mController = null;
    }
}

private void showAd(Activity activity) {
    if (mController != null) {
        mController.showAd();
        mBannerView.setVisibility(View.VISIBLE);
    }
}

private void closeAd(Activity activity) {
    if (mController != null) {
        mBannerView.setVisibility(View.GONE);
        mController.closeAd();
    }
}
```

注：更多详细代码示例，请参考Demo工程代码：**cn.sirius.adsdkdemo.BannerActivity**

注意：设置的广告回调对象应该是成员对象，不应该是new 的临时对象。因为广告sdk内部对回调对象做了一次软引用WeakReference包装，临时对象没有被其他逻辑引用可能会回收释放，最终会导致调用方收不到回调事件。

Banner广告主要API

public class NGABannerProperties extends NGAProperties<NGABannerController,NGABannerListener>

限定符和类型	方法和说明
NGABannerListener	getListener()获取广告的监听（回调）

void	setListener()设置广告的监听（回调）对 机是在发起广告请求之前。

public abstract class NGAProperties<Controller extends NGAdController,Listener extends NGAdListener>

限定符和类型	方法和说明
Activity	getActivity()返回广告显示的Activity对象
java.lang.String	getAppId()返回App ID
ViewGroup	getContainer()返回广告显示的容器
java.lang.Object	getExtraData()返回扩展属性（内部属性
abstract Listener	getListener()获取广告的监听（回调）对
java.lang.String	getPositionId()返回广告位ID
void	setExtraData(java.lang.Object data)设 （内部属性）
abstract void	setListener设置广告的监听（回调）对 机是在发起广告请求之前。

public interface NGABannerListener extends NGAdListener

public interface NGAdListener

限定符和类型	方法和说明
void	onClickAd()点击广告
void	onCloseAd()关闭广告

void	onErrorAd(int code, java.lang.String m 过程发生错误
<T extends NGAdController>	onReady(T controller)广告已准备好， 辑生成并获取得到控制对象
void	onRequestAd()通知广告请求
void	onShowAd()显示广告

public interface NGABannerController extends NGAdController

public interface NGAdController

限定符和类型	方法和说明
void	closeAd()关闭广告
void	showAd()显示广告

3.3 插屏广告

广告接入代码示例

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_ad_control);

}

private NGAInsertProperties mProperties;
private NGAInsertController mController;

//注意：请在Activity成员变量保存，使用匿名内部类可能导致回收
NGAInsertListener mAdListener = new NGAInsertListener() {
```

```
    @Override
    public void onShowAd() {
        ToastUtil.show(TAG, "onShowAd");
    }


    @Override
    public void onRequestAd() {
        ToastUtil.show(TAG, "onRequestAd");
    }

    @Override
    public <T extends NGAdController> void onReadyAd(T controller) {
        mController = (NGAInsertController) controller;
        ToastUtil.show(TAG, "onReadyAd");
    }

    @Override
    public void onCloseAd() {
        mController = null;
        ToastUtil.show(TAG, "onCloseAd");
    }

    @Override
    public void onClickAd() {
        ToastUtil.show(TAG, "onClickAd");
    }

    @Override
    public void onErrorAd(int code, String message) {
        ToastUtil.show(TAG, "onErrorAd errorCode:" + code + ", message:" +
message);
    }

};

//为了提高广告的填充率以及曝光量，建议重新加载广告时候重新调用此方法，重新请求新的广告
内容
private void loadAd(Activity activity) {
    mProperties = new NGAInsertProperties(activity, AdConfig.appId,
AdConfig.insertPosId, null);
    mProperties.setListener(mAdListener);
    NGASDK ngasdk = NGASDKFactory.getNGASDK();
    ngasdk.loadAd(mProperties);
}
```

```
public void destroyAd(Activity activity) {
    if (mController != null) {
        mController.cancelAd();
        mController.closeAd();
        mController = null;
    }
}

private void showAd(Activity activity) {
    if (mController != null) {
        mController.showAd();
    }
}

private void closeAd(Activity activity) {
    if (mController != null) {
        //mController.show(false);
        mController.cancelAd();
        mController.closeAd();
    }
}
```

注：更多详细代码示例，请参考Demo工程代码：**cn.sirius.adsdkdemo.InsertActivity**

插屏广告主要API

public class NGAInsertProperties extends NGAProperties>

限定符和类型	方法和说明
NGAInsertListener	getListener()获取广告的监听（回调）
void	setListener (NGAInsertListener listener)设置广告 调）对象，设置时机是在发起广告请求

public interface NGAInsertListener extends NGAdListener

public interface NGAInsertController extends NGAdController

限定符和类型	方法和说明
void	cancelAd()取消广告请求

3.4 视频广告

广告接入代码示例

```
public class VideoActivity extends BaseActivity {
   private static final String TAG = "VideoActivity";
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_video_ad_control);
   }
   private NGAVideoController mController;
   NGAVideoListener mAdListener = new NGAVideoListener() {

        @Override
        public void onShowAd() {
         ToastUtil.show(TAG, "onShowAd");
        }

        @Override
        public void onClickAd() {
         ToastUtil.show(TAG, "onClickAd");
        }

        @Override
      public void onCloseAd() {
         mController = null;
         ToastUtil.show(TAG, "onCloseAd");
      }

      @Override
      public void onErrorAd(final int code, final String message) {
         ToastUtil.show(TAG, "onErrorAd code=" + code);
      }

      @Override
```

```
        public void onRequestAd() {
            ToastUtil.show(TAG, "onRequestAd");
        }

        @Override
        public <T extends NGAdController> void onReadyAd(T controller) {
            mController = (NGAVideoController) controller;
            ToastUtil.show(TAG, "onReadyAd");
        }

        @Override
        public void onCompletedAd() {
            ToastUtil.show(TAG, "onCompletedAd");
        }
    };

    //为了提高广告的填充率以及曝光量，建议重新加载广告时候重新调用此方法，重新请求新的广告内容
    public void loadAd(Activity activity) {
        final NGAVideoProperties properties = new NGAVideoProperties(activity, AdConfig.appId, AdConfig.videoPosId);
        properties.setListener(mAdListener);
        NGASDK ngasdk = NGASDKFactory.getNGASDK();
        ngasdk.loadAd(properties);
    }

    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.btn_video_ad_init:
                loadAd(this);
                break;
            case R.id.btn_video_ad_uninit:
                if (mController != null) {
                    mController.destroyAd();
                }
                break;
            case R.id.btn_video_ad_has_cache:
                if (mController != null) {
                    boolean hasCacheAd = mController.hasCacheAd();
                    ToastUtil.show(TAG, "hasCacheAd=" + hasCacheAd);
                }
                break;
            case R.id.btn_video_ad_show:
                if (mController != null) {
                    mController.showAd();
```

```
            }
            break;
        }
    }
}
```

注：更多详细代码示例，请参考Demo工程代码：**cn.sirius.adsdkdemo.VideoActivity**

视频广告主要API

public class NGAVideoProperties extends NGAProperties>

限定符和类型	方法和说明
NGAVideoListener	getListener()获取广告的监听（回调）
void	setListener (NGAVideoListener listener)设置广告 调）对象，设置时机是在发起广告请求

public interface NGAVideoListener extends NGAdListener

限定符和类型	方法和说明
void	onCompletedAd()视频广告播放完成

public interface NGAVideoController extends NGAdController

限定符和类型	方法和说明
void	destroyAd()销毁广告
boolean	hasCacheAd()查询是否有缓存广告

注意： loadAd加载视频广告，如果有　onReady 回调，表明有新广告返回，调用showAd播放新的视频广告，而不是缓存的广告，如果没有 onReady 回调，接入逻辑使用旧的 mController 对象调用showAd 就会播放缓存的广告。

3.5 开屏广告

通常开屏广告会前嵌入在LauncherActivity / WelcomeActivity / SplashActivity里面，们在“NGASDK初始化”的success回调中插入请求加载开屏逻辑

广告接入代码示例

```
public class WelcomeActivity extends Activity {
   private static final String TAG = "WelcomeActivity";

   private ViewGroup container;
   // [非必须]根据场景需求决定是否需要自定义的跳过按钮
   //private TextView skipView;
   private ImageView splashHolder;
   private static final String SKIP_TEXT = "点击跳过 %d";

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_splash);
      container = (ViewGroup) this.findViewById(R.id.splash_container);
      //skipView = (TextView) findViewById(R.id.skip_view);
      //skipView.setVisibility(View.VISIBLE);

      splashHolder = (ImageView) findViewById(R.id.splash_holder);

      initSdk(this, new NGASDK.InitCallback() {
        @Override
        public void success() {
            //NGASDK初始化成功后，开始加载开屏广告
            showAd(WelcomeActivity.this);
        }

        @Override
        public void fail(Throwable throwable) {
            throwable.printStackTrace();
        }
```

```
    });

      splashHolder.postDelayed(new Runnable() {
         @Override
         public void run() {
            if (splashHolder.getVisibility() == View.VISIBLE) {
               // 超时3s开屏广告还没加载出来则关闭广告
               closeAd();
            }
         }
      }, 3000);
   }

   private NGAWelcomeProperties properties;
   private boolean canCloseAd = false;
   //注意：请在Activity成员变量保存，使用匿名内部类可能导致回收
   NGAWelcomeListener mAdListener = new NGAWelcomeListener() {

      @Override
      public void onClickAd() {
         ToastUtil.show(TAG, "onClickAd");
      }

      @Override
      public void onErrorAd(int code, String message) {
         ToastUtil.show(TAG, "onErrorAd errorCode:" + code + ", message:" +
message);
         closeAd();
      }

      @Override
      public void onShowAd() {
         splashHolder.setVisibility(View.INVISIBLE); // 广告展示后一定要把预设的
开屏图片隐藏起来
         ToastUtil.show(TAG, "onShowAd");
      }

      @Override
      public void onCloseAd() {
         //无论成功展示成功或失败都回调用该接口，所以开屏结束后的操作在该回调中实现
         ToastUtil.show(TAG, "onCloseAd");
         canCloseAd = true;
      }

      @Override
```

```
      public <T extends NGAdController> void onReadyAd(T controller) {
         // 开屏广告是闪屏过程自动显示不需要NGAdController对象，所以返回controller
为null;
         ToastUtil.show(TAG, "onReadyAd");
      }

      @Override
      public void onRequestAd() {
         ToastUtil.show(TAG, "onRequestAd");
      }


      /**
       * 倒计时回调，返回广告还将被展示的剩余时间。
       * 通过这个接口，开发者可以自行决定是否显示倒计时提示，或者还剩几秒的时候显示倒
计时
       *
       * @param millisUntilFinished 剩余毫秒数
       */
      @Override
      public void onTimeTickAd(long millisUntilFinished) {
         //skipView.setText(String.format(SKIP_TEXT,
Math.round(millisUntilFinished / 1000f)));
      }
   };

   /**
    * 开始广告，建议：闪屏Activity显示后延迟再加载广告
    * @param activity
     */
   public void showAd(Activity activity) {
      properties = new NGAWelcomeProperties(activity, AdConfig.appId,
AdConfig.welcomeId, container);
      // 支持开发者自定义的跳过按钮。SDK要求skipContainer一定在传入后要处于VISIBLE状
态，且其宽高都不得小于3x3dp。
      // 如果需要使用SDK默认的跳过按钮，可以选择上面两个构造方法。
      //properties.setSkipView(skipView);

      properties.setListener(mAdListener);
      NGASDK ngasdk = NGASDKFactory.getNGASDK();
      ngasdk.loadAd(properties);
   }

   /**
```

```
    * 关闭广告，当通知广告失败{@link NGAWelcomeListener#onErrorAd} 或关闭事件时候调用 {@link NGAWelcomeListener#onCloseAd}
    */
   private void closeAd() {
      // 如果是因为点击广告而关闭广告，则不能finish Activity
      if (canCloseAd) {
         this.startActivity(new Intent(this, MainActivity.class));
         this.finish();
      } else {
         canCloseAd = true;
      }
   }

   @Override
   protected void onPause() {
      super.onPause();
      canCloseAd = false;
   }

   @Override
   protected void onResume() {
      super.onResume();
      // 如果是因为点击广告而返回，则finish Activity
      if (canCloseAd) {
         this.startActivity(new Intent(this, MainActivity.class));
         this.finish();
      }
      canCloseAd = true;
   }
   /** 开屏页一定要禁止用户对返回按钮的控制，否则将可能导致用户手动退出了App而广告无法正常曝光和计费 */
   @Override
   public boolean onKeyDown(int keyCode, KeyEvent event) {
      if (keyCode == KeyEvent.KEYCODE_BACK || keyCode == KeyEvent.KEYCODE_HOME) {
         return true;
      }
      return super.onKeyDown(keyCode, event);
   }
}
```

注：更多详细代码示例，请参考Demo工程代码： **cn.sirius.adsdkdemo.WelcomeActivity**

- **开屏广告主要API**public class NGAWelcomeProperties extends NGAProperties>

限定符和类型	方法和说明
NGAWelcomeListener	getListener()获取广告的监听（回调）
View	getSkipView()获取自定义的跳过按钮
void	setListener (NGAWelcomeListener listener)设置广（回调）对象，设置时机是在发起广告
void	setSkipView(View skipView)可以通过 skipContainer参数，支持开发者自定义 钮。

public interface NGAWelcomeListener extends NGAdListener

限定符和类型	方法和说明
void	onTimeTickAd(long millisUntilFinished 调，返回广告还将被展示的剩余时间。

public interface NGAWelcomeController extends NGAdController

3.6 模板插屏广告

模板插屏广告与插屏广告有展示方式类似，再普通插屏广告的基础上，提供了UI的模板，具有一定的自定义属性，使得模板插屏有更好的融入性，同时也能提升广告收益

- **广告接入代码示例：**

```
public class TemplateActivity extends BaseActivity {

    private static final String TAG = "TemplateActivity";
    Map<String, String> mShowParam = new HashMap<>();
    RadioGroup mTemplateSelector;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_template);
        mTemplateSelector = (RadioGroup)findViewById(R.id.template_choice);
        //UI参数准备，具体参数名称请查阅文档
        mShowParam = new HashMap<>();
        //bgRes：弹窗背景框图片资源（模板1、模板2支持）
        mShowParam.put("bgRes", "drawable/bg");
        //dialogTextColor：弹窗文案文字颜色（仅模板2支持）
        mShowParam.put("dialogTextColor", "#edeef2");
        //front：弹窗文字字体资源（仅模板2支持，仅支持assets资源）
        mShowParam.put("front", "assets/heiti.ttf");
        //btnText：弹窗操作按钮文案（仅模板2支持）
        mShowParam.put("btnText", "点击领取");
        //btnTextColor：弹窗操作按钮文案文字颜色（仅模板2支持）
        mShowParam.put("btnTextColor", "#36561f");
        //btnRes：弹窗操作按钮背景图片（仅模板2支持）
        mShowParam.put("btnRes", "drawable/button_bg");
    }

    private NGAGeneralProperties mProperties;
    private NGAGeneralController mController;

    //注意：请在Activity成员变量保存，使用匿名内部类可能导致回收
    NGAGeneralListener mAdListener = new NGAGeneralListener() {

        @Override
        public void onEvent(NGAdEvent event) {
            //具体eventId代表含义请查阅文档
            if (event.eventId == 1) {
                ToastUtil.show(TAG, "onGameBtnClick");
                //再次调用showAd(Map<String,String> param)，可以动态更换素材参数
                mShowParam.put("dialogText", "点击领取");
                mController.showAd(mShowParam);
            }
        }
```

```
        @Override
        public void onShowAd() {
            ToastUtil.show(TAG, "onShowAd");
        }


        @Override
        public void onRequestAd() {
            ToastUtil.show(TAG, "onRequestAd");
        }

        @Override
        public <T extends NGAdController> void onReadyAd(T controller) {
            mController = (NGAGeneralController) controller;
            ToastUtil.show(TAG, "onReadyAd");
        }

        @Override
        public void onCloseAd() {
            ToastUtil.show(TAG, "onCloseAd");
            mController = null;
        }

        @Override
        public void onClickAd() {
            ToastUtil.show(TAG, "onClickAd");
        }

        @Override
        public void onErrorAd(int code, String message) {
            ToastUtil.show(TAG, "onErrorAd errorCode:" + code + ", message:" +
message);
        }

    };

    //为了提高广告的填充率以及曝光量，建议重新加载广告时候重新调用此方法，重新请求新的
广告内容
    private void loadAd(Activity activity) {
        Map<String, Object> param = new HashMap<>();
        param.put(NGAGeneralProperties.APP_ID, AdConfig.appId);
        //不同广告位可以配置不同的模板效果，可以与我们联系进行选择
        param.put(NGAGeneralProperties.POSITION_ID, getSelectedPosId());
        param.put(NGAGeneralProperties.AD_TYPE, 20);
```

```
    mProperties = new NGAGeneralProperties(activity, null, param);
    mProperties.setListener(mAdListener);

    NGASDK ngasdk = NGASDKFactory.getNGASDK();
    ngasdk.loadAd(mProperties);
}

public void destroyAd(Activity activity) {
    if (mController != null) {
        mController.closeAd();
        mController = null;
    }
}

private void showAd(Activity activity) {
    if (mController != null) {
        mShowParam.put("dialogText", "小小礼物请领取");
        mController.showAd(mShowParam);
    }
}

private void closeAd(Activity activity) {
    if (mController != null) {
        //mController.show(false);
        mController.closeAd();
    }
}

public void onClick(View view) {
    if (view.getId() == R.id.btn_banner_create) {
        loadAd(this);
    } else if (view.getId() == R.id.btn_banner_destroy) {
        destroyAd(this);
    } else if (view.getId() == R.id.btn_banner_show) {
        showAd(this);
    } else if (view.getId() == R.id.btn_banner_close) {
        closeAd(this);
    }
}

private String getSelectedPosId() {
    switch (mTemplateSelector.getCheckedRadioButtonId()) {
        case R.id.rb_temp_1st:
            return AdConfig.templateId;
        case R.id.rb_temp_2nd:
```

```
                return AdConfig.templateId_2;
            default:
                return "";
        }
    }
}
```

注：更多详细代码示例，请参考Demo工程代码： **cn.sirius.adsdkdemo.TemplateActivity**

- **模板插屏广告主要API**

public class NGAGeneralProperties extends NGAProperties<NGAGeneralController, NGAGeneralListener>

限定符和类型	方法和说明
NGAGeneralListener	getListener()获取广告的监听（回调）
void	setListener(NGAGeneralListener lister 的监听（回调）对象，设置时机是在发 之前。

public interface NGAGeneralListener extends NGAdListener

public interface NGAWelcomeController extends NGAdController

模板传入参数详细

在请求到广告，获得controller后，可以调用showAd(Map<String,String> param)方法来传入自定义参数，来调整模板的样式，具体参数描述如下：

参数key	参数意义	参数格式	使用示例	备注	使
bgRes	弹窗背景图资源	String	mShowParam.put("bgRes", "drawable/bg");	获取drawable文件夹下的资源，支持drawable、mipmap	1号模
dialogText	弹窗游戏文案	String	mShowParam.put("dialogText", "点击领取");	直接传入文字内容传参	2号
dialogTextColor	弹窗游戏文案颜色	String	mShowParam.put("dialogTextColor", "#edeef2");	使用色码来表示颜色，需要“#”开头	2号
btnText	操作按钮文案	String	mShowParam.put("btnText", "点击领取");	直接传入文字内容传参	2号
btnTextColor	操作按钮文案颜色	String	mShowParam.put("btnTextColor", "#36561f");	使用色码来表示颜色，需要“#”开头	2号
btnRes	操作按钮图资源	String	mShowParam.put("btnRes", "drawable/button_bg");	获取drawable文件夹下的资源，支持drawable、mipmap，支持png、xml格式	2号

front	字体文件	String	mShowParam.put("btnRes", "assets/heiti.ttf");	获取assets文件夹下的资源，获得字体资源文件	2듁

showAd(Map<String,String> param)可以重复调用，动态改变弹窗样式，但曝光仅计算一次

如果上述参数不填写，则由sdk补充默认参数

模板事件回调id

当模板上触发特殊事件时，sdk会回调onEvent(NGAdEvent event)，接入方可根据event.eventId判断回调事件类型，具体参数如下：

eventId	参数意义
1	操作按钮被点击（首次）
2	操作按钮被点击（第二次或者以上）

3.7 原生广告

原生广告提供了高度自定义的广告类型，使得广告能与app高度融合，给开发者提供了高自由度的开发接口

广告接入代码示例：

```
public class NativeAdActivity extends BaseActivity {
    private static final String TAG = "NativeAdActivity";

    private NGANativeProperties mProperties;
    private NGANativeController mController;
```

```
    private NGANativeAdData mAdDataItem;

    private LinearLayout mMainContainer;
    private RelativeLayout mAdContainer;
    private ImageView mIvAppIcon;
    private TextView mTvAppTitle;
    private TextView mTvAppDesc;
    private TextView mTvAppScore;
    private ImageView mIvAppImg;
    private TextView mBtnClick;
    private ImageView mIvAdLogo;

    //注意：请在Activity成员变量保存，使用匿名内部类可能导致回收
    NGANativeListener mAdListener = new NGANativeListener() {

        @Override
        public void onAdStatusChanged(NGANativeAdData ngaNativeAd, int i,
Map<String, String> map) {
            Log.i(TAG, "onAdStatusChanged " + i);
        }

        @Override
        public void onShowAd() {
            ToastUtil.show(TAG, "onShowAd");
        }


        @Override
        public void onRequestAd() {
            ToastUtil.show(TAG, "onRequestAd");
        }

        @Override
        public <T extends NGAdController> void onReadyAd(T controller) {
            mController = (NGANativeController) controller;
            ToastUtil.show(TAG, "onReadyAd");
        }

        @Override
        public void onCloseAd() {
            ToastUtil.show(TAG, "onCloseAd");
            mController = null;
        }

        @Override
```

```
        public void onClickAd() {
            ToastUtil.show(TAG, "onClickAd");
        }

        @Override
        public void onErrorAd(int code, String message) {
            ToastUtil.show(TAG, "onErrorAd errorCode:" + code + ", message:" +
message);
        }

    };

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_ad_native);
        mMainContainer = (LinearLayout)
findViewById(R.id.rl_control_container);
    }

    private void refreshAdContainer() {
        if (mAdContainer != null) {
            mMainContainer.removeView(mAdContainer);
        }
        mAdContainer = (RelativeLayout)
LayoutInflater.from(this).inflate(R.layout.native_ad_contaner, null, false);
        mIvAppIcon = (ImageView) mAdContainer.findViewById(R.id.iv_app_icon);
        mTvAppTitle = (TextView) mAdContainer.findViewById(R.id.tv_app_title);
        mTvAppDesc = (TextView) mAdContainer.findViewById(R.id.tv_app_desc);
        mTvAppScore = (TextView) mAdContainer.findViewById(R.id.tv_app_score);
        mIvAppImg = (ImageView) mAdContainer.findViewById(R.id.iv_app_img);
        mBtnClick = (TextView) mAdContainer.findViewById(R.id.btn_app_click);
        mIvAdLogo = (ImageView) mAdContainer.findViewById(R.id.iv_ad_logo);
        mMainContainer.addView(mAdContainer);
    }

    private void loadAd(Activity activity) {
        //如果使用原有的容器再次请求广告，必须关闭原有的广告
        closeAd(activity);
        //每次加载广告必须使用新的广告容器view
        refreshAdContainer();
        Map<String, Object> param = new HashMap<>();
        //一次加载所提供的广告数量。不一定能够给到传入的数量，请以最终返回的广告数量为
准
        param.put(NGANativeProperties.KEY_AD_COUNT, 1);
```

```
    param.put(NGANativeProperties.APP_ID, AdConfig.appId);
    param.put(NGANativeProperties.POSITION_ID, AdConfig.nativeId);
    mProperties = new NGANativeProperties(activity, param);
    mProperties.setListener(mAdListener);
    NGASDK ngasdk = NGASDKFactory.getNGASDK();
    ngasdk.loadAd(mProperties);
}

public void destroyAd(Activity activity) {
    if (mController != null) {
        mController.closeAd();
        mController = null;
    }
}

private void showAd(Activity activity) {
    if (mController != null) {
        mAdDataItem = mController.getAdList().get(0);
        new LoadImageUtil(mAdDataItem.getIconUrl()){
            @Override
            public void onReceived(Drawable result) {
                mIvAppIcon.setImageDrawable(result);
                //广告曝光后，一定一定要调用该方法，否则无法计算曝光数量
                mAdDataItem.exposure(mAdContainer);
            }
        }.execute();
        new LoadImageUtil(mAdDataItem.getImgList().get(0)){
            @Override
            public void onReceived(Drawable result) {
                mIvAppImg.setImageDrawable(result);
            }
        }.execute();
        //根据相关规定，广告必须要有广告标识，请将该广告logo放置在广告右下角
        new LoadImageUtil(mAdDataItem.getAdLogo()) {
            @Override
            public void onReceived(Drawable result) {
                mIvAdLogo.setImageDrawable(result);
            }
        }.execute();
        mTvAppTitle.setText(mAdDataItem.getTitle());
        mTvAppDesc.setText(mAdDataItem.getDesc());
        if (mAdDataItem.getRating() > 0) {
            mTvAppScore.setText("评分：" + mAdDataItem.getRating());
        } else {
            mTvAppScore.setText("暂无评分");
```

```
            }
            mBtnClick.setText(mAdDataItem.getButtonText());
            mBtnClick.setBackgroundColor(Color.GRAY);
        }
    }

    private void closeAd(Activity activity) {
        if (mController != null) {
            mController.closeAd();
            mController = null;
        }
    }

    @Override
    protected void onDestroy() {
        //原生广告必须调用关闭，否则影响广告计费
        closeAd(this);
        super.onDestroy();
    }

    public void onClick(View view) {
        if (view.getId() == R.id.btn_native_create) {
            loadAd(this);
        } else if (view.getId() == R.id.btn_native_destroy) {
            destroyAd(this);
        } else if (view.getId() == R.id.btn_native_show) {
            showAd(this);
        } else if (view.getId() == R.id.btn_native_close) {
            closeAd(this);
        }
    }
}
```

注：更多详细代码示例，请参考Demo工程代码： **cn.sirius.adsdkdemo.NativeAdActivity**

- **原生广告主要API**

public class NGANativeProperties extends NGAProperties<NGANativeController,NGANativeListener>

限定符和类型	方法和说明
NGANativeListener	getListener()获取广告的监听（回调）
void	setListener(NGANativeListener listene 监听（回调）对象，设置时机是在发起 前。

public interface NGANativeListener extends NGAdListener

public interface NGANativeController extends NGAdController

限定符和类型	方法和说明
`List<NGANativeAdData>`	getAdList()获取广告对象列表
void	closeAd()标记广告关闭，务必调用，否

public interface NGANativeAdData

限定符和类型	方法和说明
String	getTitle();获得广告标题文字
String	getDesc();获得广告详细描述文字
String	getIconUrl();获得广告图标url
List	getImgList();获得广告图片url集合
double	getRating();获得广告评分

boolean	isApp();广告是否app类型
String	getButtonText();广告按钮文案
String	getAdLogo();获得广告logo图标url，根 定，请将该logo放在广告右下角
void	exposure(View container);标记广告曝 听点击功能，曝光后务必调用，否则影

4. Android 6.0以上版本添加运行时权限

4.1 接入说明

建议您在AndroidManifest.xml添加以下权限声明，若您的targetSDKVersion >= 23您还需要在运行时进行动态权限申请（可参考示例工程）

```
<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" />
<!-- 可选，如果需要精确定位的话请加上此权限 -->
```

注意：**SDK不强制校验上述权限（即:无上述权限sdk也可正常工作），但建议您申请上述权限。针对单媒体的用户，允许获取权限的，投放定向广告；不允许获取权限的用户，投放通投广告。媒体可以选择是否把上述权限提供给广告SDK，并承担相应广告填充和eCPM单价下降损失的结果。**

4.2 接入示例

如第三方接入已经有运行时权限检查的功能，请将1.2步骤AndroidManifest.xml中SDK运行需要的权限自行添加。

如第三方接入未添加运行时权限检查功能，根据实际业务情况，参考如下步骤添加，兼容Android 6.0以上的版本。

4.2.1 Acitivity中加入代码

```
    private PermissionHelper mPermissionHelper = new PermissionHelper();
    protected List<String> getOptionalPermissions() {
        List<String> optionalsPermissions = new ArrayList<>();
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M &&
Build.VERSION.SDK_INT <= Build.VERSION_CODES.P) {
            //只在Android6～9申请电话权限
            optionalsPermissions.add(Manifest.permission.READ_PHONE_STATE);
        }
        optionalsPermissions.add(Manifest.permission.ACCESS_COARSE_LOCATION);
        return optionalsPermissions;
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, String[]
permissions, int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions,
grantResults);
        mPermissionHelper.onRequestPermissionsResult(requestCode, permissions,
grantResults);
    }

    protected void loadAdWithPermission() {
        mPermissionHelper.requestPermission(this, getOptionalPermissions(), new
 PermissionHelper.RequestPermissionCallback() {
            @Override
            public void onPermissionGrantedResult(String[] grantedPermission,
boolean isAllGranted) {
                //TODO 加载广告
            }
        });
    }
```

4.2.2 在MainActivity（或应用的主Activity类型） #onCreate 函数增加权限判断，如下：

```
@Override
```

```
protected void onBeforeRequestPermission(Bundle savedInstanceState) {
    super.onBeforeRequestPermission(savedInstanceState);
    // ... 你的初始化代码 ...
}
```

5. 兼容性

- **请保持你接入的support-v4包保持与targetSdkVersion一致，v4建议26+版本**
- 广告SDK支持的安卓平台版本为14~24，也就是4.0以上到7.0的都支持，包括视频内核。
- **targetSDKVersion >= 24**时的文件访问兼容处理 如果您打包时的 `targetSDKVersion >= 24` ，为了让SDK能够正常下载、安装App类广告，必须按照下面的三个步骤做兼容性处理。注意：如果您的 `targetSDKVersion < 24` ，不需要做这个兼容处理。
  （1）在 `AndroidManifest.xml` 中的 `Application` 标签中添加 `provider` 标签，接入代码如下所示：

```
<application
    ... ...

    <!-- 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 -->
    <provider
        android:name="android.support.v4.content.FileProvider"
        android:authorities="${applicationId}.fileprovider"
        android:exported="false"
        android:grantUriPermissions="true">
        <meta-data
            android:name="android.support.FILE_PROVIDER_PATHS"
            android:resource="@xml/adp_file_path" />
    </provider>
    ... ...
</application>
```

需要注意的是provider的authorities值为${applicationId}在代码中和 `Context#getPackageName()` 值相等，是应用的唯一id。例如demo示例工程中的applicationId为"com.qq.e.union.demo"。

（2）在项目结构下的 `res` 目录下添加一个 `xml` 文件夹，再新建一个 `adp_file_path.xml` 的文件，文件内容如下：

```
<paths xmlns:android="http://schemas.android.com/apk/res/android">
    <external-path name="gdt_sdk_download_path" path="GDTDOWNLOAD" />
    <root-path name="download" path="" />
    <external-path name="external_files" path="."/>
</paths>
```

- **如果您打包 App 时的 targetSdkVersion >= 26：需要在 AndroidManifest.xml 增加权限声明 android.permission.REQUEST_INSTALL_PACKAGES**，详情见前面添加权限声明部分。

6. 合规指南

最新《合规指南》文档：
https://cdn.9game.cn/9game/app/pdf/%E5%B9%BF%E5%91%8ASDK%E5%90%88%E8%A7%84%E6%8C%87%E5%8D%97.pdf

最新《采集详情》文档：

https://cdn.9game.cn/9game/app/pdf/%E5%B9%BF%E5%91%8ASDK%E9%87%87%E9%9B%86%E8%AF%A6%E6%83%85.pdf

根据《个人信息保护法》、《数据安全法》、《网络安全法》等法律法规和监管部门规章要求，App开发运营者（以下简称为“开发者”）在提供网络产品服务时应尊重和保护最终用户的个人信息，不得违法违规收集使用个人信息，保证和承诺就个人信息处理行为获得最终用户的授权同意，遵循最小必要原则，且应当采取有效的技术措施和组织措施确保个人信息安全。为帮助开发者在使用广告SDK的过程中更好地落实用户个人信息保护相关要求，避免出现侵害最终用户个人信息权益的情形，特制定本合规使用说明，供开发者在接入使用广告SDK服务时参照自查和合理配置，不断提升个人信息保护水平。

APP合规仍为监管执法和各大应用市场的重要关注点。为避免App被下架，请务必做好两件事：

1、首先将SDK升级至满足监管要求的最新版本；

2、按下文合规解法进行配置。

6.1、升级到满足监管新规的最新版本

针对广告SDK版本升级，参照以下步骤：

1. 从官网下载最新版本(https://www.yousuode.cn/download/sdk) 解压获取aar包（在ngad-sdk-all-*.zip\04-依赖库\aar接入方式\ngad-sdk-release-*.aar 里），对比版本号，如果已经是最新版本2.21.2+则无需替换；如果不是最新版本则需要**替换更新aar包**。

6.2、SDK隐私政策披露要求与示例

接入说明： 开发者在App集成广告SDK后，广告SDK的正常运行会收集必要的最终用户信息用于展示内容及向最终用户推荐可能感兴趣的内容。 请开发者根据集成广告SDK的实际情况，在您App的隐私政策中，对广告SDK名称、公司名称、处理个人信息种类及目的、采集方式、隐私政策链接等内容进行披露。建议：确认您所接入的广告SDK版本和功能模块；根据上述版本和模块，从隐私政策中确定与广告SDK交互的数据内容；在您App的隐私政策中，以文字或列表的方式向公众披露广告SDK的相关信息。

披露示例（仅供参考，请以实际合作情况为准）：

SDK名称：九游广告SDK

运营方：广州爱九游信息技术有限公司

使用目的：为开发者提供帮助调整广告变现策略及数据分析的服务

使用场景：在开发者进行广告投放，需要进行广告变现策略调整及数据分析时进行使用

收集方式：SDK自行采集

收集个人信息类型：设备品牌、型号、操作系统及api版本信息、系统时区、系统语言、屏幕密度、屏幕分辨率、CPU信息、设备标识符（如AndroidID、IMEI、MAC地址、OAID、IMSI，具体字段因软硬件版本不同而存在差异）、网络状态（WiFi状态）、应用的名称、应用的包名、应用的版本号、应用安装列表、位置信息

隐私权政策链接：https://open.9game.cn/sdkprivacy

SDK名称：优量汇

使用目的：向用户展示广告，及广告监测归因、反作弊

运营方：深圳市腾讯计算机系统有限公司

收集个人信息类型：移动设备国家代码MCC+移动设备网络代码MNC，国际移动设备识别码（IMEI），运行中进程、packagename，位置信息(GPS信息)、精确地理位置信息、大致地理位置信息，基站经纬度，设备信息（设备型号、操作系统版本、唯一设备标识符、电池、信号强度等），用户私有目录文件，android_id，位置经纬度，外部文件存储目录信息、运行中应用列表、文件目录、OAID、软件安装列表

隐私权政策链接：https://www.tencent.com/zh-cn/privacy-policy.html

SDK名称：穿山甲

使用目的：向用户展示广告，及广告监测归因、反作弊

运营方：北京巨量引擎网络技术有限公司

收集个人信息类型：国际移动设备识别码（IMEI），存储文件、目录、空间状态，设备信息（例如硬件型号、操作系统版本号、国际移动设备身份识别码（IMEI）、网络设备硬件地址（MAC）），用户私有目录文件，设备序列码，BSSID，SSID，移动设备国家代码，MCC+移动设备网络代码MNC，运行中应用程序信息，运行中进程、packagename，IMSI，位置信息(GPS信息)、精确地理位置信息、大致地理位置信息，android_id，WiFi 列表，网络设备硬件MAC地址，外部文件存储目录信息，位置经纬度、系统SHELL命令、运行中应用列表、文件目录、OAID、软件安装列表

隐私权政策链接：https://www.csjplatform.com/privacy

6.3、SDK初始化及业务功能调用时机

请务必在用户同意您App中的隐私政策后，再进行广告SDK的初始化。用户同意隐私政策之前，避免动态申请涉及用户个人信息的敏感设备权限；用户同意隐私政策前，您应避免私自采集和上报个人信息。当您的App未向用户提供服务时，例如App在后台运行时，请勿请求广告SDK的相关服务。具体的初始化时机参考示例。

本方案支持九游广告SDK2.21.2及更高版本，强烈建议开发者将九游广告SDK升级到2.21.2版本。

- **初始化步骤：** 请确保在App安装后首次冷启动时按照如下方式进行初始化。

【1】在Applicaiton.onCreate函数中调用预初始化函数UMConfigure.preInit()，预初始化函数不会采集设备信息，也不会向后台上报数据。

```
public class XXXApplication extends Application {
    protected static final String TAG = "DemoApp";
    @Override
    public void onCreate() {
        super.onCreate();
        // 同意隐私政策后调用初始化
        initAdSdk();
    }

    protected void initAdSdk() {
        initSdk(this, new NGASDK.InitCallback() {
            @Override
            public void success() {
                //NGASDK init success, and can try to show splash ad.
                Log.d(TAG, "广告SDK初始化成功");
            }

            @Override
            public void fail(Throwable throwable) {
                Log.e(TAG, "广告SDK初始化异常：" + Log.getStackTraceString(throwable));
```

```
        }
    });
}

protected void initSdk(Context context, final NGASDK.InitCallback initCallback) {
    Log.d(TAG, MediaAdConfig.toStringFormat());
    NGASDK ngasdk = NGASDKFactory.getNGASDK();
    ngasdk.init(context, new AdConfig.Builder()
            .setAppId(MediaAdConfig.appId)
            .setDebug(true)
            .showNotification(true)
            .supportPersonalizedAd(true)
            .build(), initCallback);
}
}
```

【2】确保App首次冷启动时，在用户阅读您的《隐私政策》并取得用户授权之后，才调用正式初始化函数NGASDKFactory.getNGASDK().init()初始化广告SDK，反之，如果用户不同意《隐私政策》授权，则不能调用NGASDKFactory.getNGASDK().init()初始化函数。

【3】一旦App获取到《隐私政策》的用户授权，后续的App冷启动，开发者应该保证在Applicaiton.onCreate函数中调用预始化函数NGASDKFactory.getNGASDK().init()。

6.4、个性化开关配置开关配置说明

个性化推荐广告设置

为落实个人信息保护相关的规定，SDK为开发者提供退出个性化广告能力的接口，开发者可以调用接口，向用户提供退出个性化广告的能力。退出后，看到的广告数量不变，相关度会降低。开发者需遵守相关法律法规的要求，在开发者应用内为用户提供退出个性化广告的功能。

API说明

开发者在初始化广告SDK成功后，可以开关进行设置：

```
cn.sirius.nga.shell.GlobalSetting.setPersonalizedState(int state)
```

参数	描述	值
state	是否屏蔽个性化广告	当用户选择拒绝个性化广告时上报传入 GlobalSetting.STATE_PERSONALIZED_UNSUPPORT 当用户同意时上报传入 GlobalSetting.STATE_PERSONALIZED_SUPPORT

6.5、SDK可选个人信息配置说明

相关信息的不收集会对其对应的功能造成影响，请开发者结合业务实际需要进行合理配置。

API说明（NGACustomController）

接口	描述	备注
allowUsePhoneState	设备标识符（如：MAC地址、OAID、IMSI、IMEI，具体字段因软硬件版本不同而存在差异）	是否允许SDK获取设备ID，用于广告投放、反作弊追踪
getOaid	设备标识符-OAID	Android10+采用oaid追踪用户的广告行为； 请传入原始oaid，无需使用md5加密，共享信息给SDK
getImei	设备标识符-IMEI	Android10-采用imei追踪用户的广告行为； 请传入原始的imei信息，sdk使用您传入的原始imei信息
getMacAddress	设备标识符-MAC地址信息	共享mac地址给SDK

allowUseAndroidId	设备标识符-AndroidID	AndroidID，辅助用户行为分析
allowUseWifiState	网络状态（WiFi状态）	共享WiFi状态给SDK
allowReadInstalledPackages	应用安装列表	是否允许读取应用安装列表，用于判断用户已安装应用，优化广告投放、下载、安装逻辑
allowReadLocation	位置信息	共享位置信息给SDK
allowUseStoragePermission	存储空间权限	是否允许使用存储空间权限

接入示例

在SDK初始化阶段（即在NGASDKFactory.getNGASDK().init()处）传入

```
/**
 * 用户信息采集配置
 */
 NGACustomController ngaCustomController = new NGACustomController() {
        @Override
        public boolean allowReadLocation() {
            return super.allowReadLocation();
        }

        @Override
        public Location getLocation() {
            return super.getLocation();
        }
```

```
@Override
public boolean allowUsePhoneState() {
    return super.allowUsePhoneState();
}

@Override
public String getImei() {
    return super.getImei();
}

@Override
public String getOaid() {
    return super.getOaid();
}

@Override
public boolean allowUseAndroidId() {
    return super.allowUseAndroidId();
}

@Override
public String getMacAddress() {
    return super.getMacAddress();
}

@Override
public boolean allowUseWifiState() {
    return super.allowUseWifiState();
```

```
        }

        @Override
        public boolean allowUseStoragePermission() {
            return super.allowUseStoragePermission();
        }

        @Override
        public boolean allowReadInstalledPackages() {
            return super.allowReadInstalledPackages();
        }
    };

NGASDK ngasdk = NGASDKFactory.getNGASDK();
ngasdk.init(activity, new AdConfig.Builder()
        .setAppId(MediaAdConfig.appId)
        .setGameId("")
        //打Release包的时候，需要把DebugMode设置为false
        .setDebug(true)
        .showNotification(true)
        .supportPersonalizedAd(true)
        //可选个人信息配置
        .setCustomController(ngaCustomController)
        .setFetchConfig(new AdConfig.FetchConfigBuilder()
                .setSplashLimitTime(2500)
                .setRewardVideoLimitTime(15000)
                .setLimitTime(10000)
                .build())
```

.build(), initCallback);

6.6、SDK申请系统权限的说明

要求内容：《SDK合规使用说明》应详细说明SDK所需的系统权限与各业务功能间的关系，并说明权限申请时机。

接入说明：对于广告SDK可选申请的系统权限，您可以参考相关如下表格的内容，详细了解相关权限与各业务功能的关系及其申请时机，因相关权限的不申请将会对其对应的功能造成影响，您可以结合业务实际需要进行合理配置。

安卓操作系统应用权限列表：

权限	功能	用途和目的	申请时机
ACCESS_FINE_LOCATION	【可选】访问精准定位	根据位置情况实现流量分组功能	开发者在调用需要该权限的SDK功能时进行调用。例如当开发者需要根据精确位置情况使用调整广告变现策略及数据分析服务时进行时申请。
ACCESS_COARSE_LOCATION	【可选】访问粗略位置	根据位置情况实现流量分组功能	开发者在调用需要该权限的SDK功能时进行调用。例如当开发者需要根据粗略位置情况使用调整广告变现策略及数据分析服务时进行时申请。

READ_PHONE_STATE	【可选】读取手机设备标识(设备 IMSI/IMEI 号）等信息	根据设备信息情况实现流量分组功能	开发者在调用需要该权限的SDK功能时进行调用。例如当开发者需要根据设备信息情况使用调整广告变现策略及数据分析服务时进行时申请。
WRITE_EXTERNAL_STORAGE	【可选】写入外置存储器	写入配置、文件等数据	开发者在调用需要该权限的SDK功能时进行调用。例如当开发者投放其他广告平台的广告时，其他广告平台进行广告投放时进行申请。
READ_EXTERNAL_STORAGE	【可选】读取外置存储器	读取配置、文件等数据	开发者在调用需要该权限的SDK功能时进行调用。例如当开发者投放其他广告平台的广告时，其他广告平台进行广告投放时进行申请。

QUERY_ALL_PACKAGES 应用软件列表	【可选】获取应用软件列表	由其他广告平台调取，用于广告投放	开发者在调用需要该权限的SDK功能时进行调用。例如当开发者投放其他广告平台的广告，其他广告平台需要依据应用软件列表实现广告投放时进行申请。

6.7、最终用户行使权利的配置说明

接入说明：开发者在其App中集成广告SDK后，广告SDK的正常运行会收集必要的最终用户信息用于实现业务功能。开发者应根据相关法律法规为最终用户提供行使个人信息主体权利的路径或功能， 需要广告SDK配合的，请与广告SDK及时进行联系，我们将与开发者协同妥善解决最终用户的诉求。

7. 注意事项

开发套件

确保所使用的 android-support-v4.jar 包中的 android.support.v4.app.NotificationCompat.Builder 类包含 setProgress 方法，如果不包含此方法请升级 android 开发套件

代码混淆

如果您的发布包（release包）需要使用proguard混淆代码，需确保不要混淆SDK的代码。请在proguard.cfg或proguard-rules.pro文件(或其他混淆文件)尾部添加如下配置：

```
-keepattributes SourceFile,LineNumberTable
-keepattributes Signature
-keepattributes *Annotation*
```

```
## common
-keep public class * extends android.app.Activity
-keep public class * extends android.app.Application
-keep public class * extends android.app.Service

-keep class android.app.**{*;}
-dontwarn  android.app.**

-keep class android.support.v7.media.*{public *;}
-keep class android.support.v4.** { *; }
-dontwarn android.support.**

## network libs
-keep class android.net.http.** { *; }
-dontwarn android.net.**
-dontnote android.net.http.*

-keep class org.apache.http.** { *; }
-dontwarn org.apache.**
-dontnote org.apache.commons.codec.**
-dontnote org.apache.http.**

# Keep native methods
-keepclasseswithmembers class * {
    native <methods>;
}

### utdid
-keep class com.ta.utdid2.**{*;}
-keep class com.ut.device.**{*;}
-dontwarn com.ta.utdid2.**
-dontwarn com.ut.device.**

# Keep ngad-sdk classes
-keep class cn.sirius.nga.** {*; }
-dontwarn cn.sirius.nga.**

-keep class cn.ninegame.library.** {*; }
-dontwarn cn.ninegame.library.**

-keep class com.qq.e.** {*; }
-dontwarn com.qq.e.**

-keep class com.taobao.** {*; }
-dontwarn com.taobao.**
```

```
-keep class android.taobao.** {*; }
-dontwarn android.taobao.**

-keep class com.UCMobile.Apollo.**{*;}


-dontwarn com.mobvista.**
-keep class com.mobvista.** {*; }
-keep interface com.mobvista.** {*; }
-keep class **.R$* { public static final int mobvista*; }
-keep class com.alphab.** {*; }
-keep interface com.alphab.** {*; }

-dontwarn com.lm.**
-keep class com.lm.** { *; }

-dontwarn com.uniplay.**
-keep class com.uniplay.** { *; }

-keep class com.bytedance.sdk.openadsdk.** { *; }
-keep class com.androidquery.callback.** {*;}
-keep public interface com.bytedance.sdk.openadsdk.downloadnew.** {*;}

-keepattributes Signature
-keepattributes *Annotation*
-keep class com.mintegral.** {*; }
-keep interface com.mintegral.** {*; }
-keep class android.support.v4.** { *; }
-dontwarn com.mintegral.**
-keep class **.R$* { public static final int mintegral*; }
-keep class com.alphab.** {*; }
-keep class com.ss.android.socialbase.**{*;}
-keep class com.androidquery.**{*;}

-keep class sun.misc.Unsafe { *; }
-dontwarn com.sigmob.**
-keep class com.sigmob.**.**{*;}
```

错误码

错误码	错误描述	处理方案

8101	初始化sdk失败，传入参数错误，请检查appId、posId和activity	
8102	该广告位无法请求广告，请检查包名是否正确	
8103	短时间内请求广告过于频繁，请1分钟后重试或使用其他广告位	
8104	网络响应错误，数据解析异常，请稍候重试	
8105	网络请求错误，请检查网络状态	
8106	网络响应错误，广告模板缺失	
8107	网络请求错误，配置平台ID缺失	
8108	网络请求错误，曝光ID缺失	
8109	网络请求错误，业务ID（bid）缺失	
8110	网络请求错误，配置媒体ID缺失	
8111	网络请求错误，广告数据缺失	
8201	广告加载错误，暂时未有合适的广告	
8202	广告加载错误，加载广告失败，请重试	
8203	广告加载错误，不支持的广告平台	
8204	广告加载错误，加载平台广告异常	

8205	广告加载错误，平台媒体ID缺失	
8206	广告加载错误，加载广告模板异常	
8207	广告加载错误，加载广告模板素材异常	
8208	广告加载错误，传入UI容器为空，请检查container参数	
8209	广告加载错误，当前设备的网络类型不符合加载广告的条件，请尝试WIFI环境	
8210	广告加载错误，广告视频预加载异常	
8211	广告加载错误，广告插件启动失败	
8212	广告加载错误，加载广告异常	
8213	广告加载错误，加载广告超时，请检查网络情况或稍候重试	
8301	广告曝光错误，曝光失败	
8302	广告曝光错误，广告页面处于不可见状态，请检查您的代码逻辑，保证容器可见	
8303	广告曝光错误，开屏广告的高度不得小于400dp	
8304	广告曝光错误，视频广告数据异常	

8305	广告曝光错误，广告视频播放错误	
8306	广告曝光错误，传入的activity已经销毁，广告无法展示	
8307	广告曝光错误，广告已经过期，请重新加载	
8308	广告曝光错误，广告暂时未准备好，请重新加载	
8xxx	内部错误，请与客服联系并描述错误回调值	
50010001	appId_posId_invalid	请检测传入广告位ID
50010006	adx_request_error	服务端异常，建议重
50010007	ad_policy_empty	检查是否有配置流量同学
50010008	ad_type_incorrect	请检查广告类型 找运
50010009	adx_no_ad_return	没有广告返回，建议
50010010	ad_policy_platform_id_incorrect	请检查平台id
50010011	media_game_id_could_not_be_empty	游戏ID 不能为空 找运
50010012	game_id_rel_media_no_ad_position	游戏id对应的媒体没位，找运营同学
50010013	ad_disabled_on_distribution	当前媒体没有开启这告。返回错误，不加
50010014	adposition_biz_status_abnormal	广告位状态异常，找

50010015	invalid_media_uuid	媒体不存在，找运营
50010016	invalid_adposition_uuid	指定广告位不存在，
50010017	invalid_media_pkg_name	媒体和应用包名配置 营同学
50010019	media_status_abnormal	媒体状态不对，找运
50010020	media_status_not_allow_cancel	媒体状态不对，找运
50010021	traffic_control_frequency_limit	请求过于频繁，建议 请求
50010022	ad_disabled_on_ch	渠道不准确 ，找运营
50010026	adposition_frozen	广告位暂停，找运营
50010027	user_imp_over_limit	用户曝光次数超出限 一天后重试
50010028	traffic_control_upper_limit	用户请求次数超出限 一天后重试
50010029	invalid_imei	imei不合法，请自检
50010030	invalid_idfa_idfv	idfa和idfv不合法，请

常见问题FAQ

问题：广告无法加载出来的场景问题

原因分析，可能有以下情况导致：

1. 没有调用sdk初始化：init ， 建议在Application OnCreate做广告sdk初始化；
2. Eclipse工程引入sdk情况下，缺失jar、so文件；
3. Eclipse工程引入sdk情况下，没有引入support包；

4. 网络不通（有WiFi但无法联网），导致广告请求连接失败；或者：
    - App安装后首次启动初始化逻辑比较多网络请求比较多，导致广告请求连接超时失败；
    - 这种情况下，建议延迟广告请求（如延迟2s后），这样不影响你们其他正常业务的网络请求，同时也保证尽量让每次广告请求都成功。
5. release包混淆配置没有添加广告sdk过滤配置，参考接入说明文档。

问题：设置了广告回调 （NGAdListener子类对象），但是没有收到回调通知。

原因分析：设置的广告回调对象是new 的临时对象，不是成员对象，没有被其他逻辑引用可能会回收释放，因为广告sdk内部对回调对象做了一次软引用WeakReference包装。

问题：获取到广告控制器（NGAdController对象）**之后，能不能重复调用展示/关闭广告操作（showAd/closeAd）？**

答：横幅、普通插屏、开屏、视频广告不可以， 调用closeAd关闭广告之后，再调用showAd，很可能无法正常显示广告。只有模板广告允许多次调用，以供微调自定义界面。重新加载广告需要重新调用NGASDK#loadAd方法，等待onReadyAd事件回调获取广告控制器重新加载新的广告（showAd），此操作同时提高广告的填充率以及曝光量。

问题：如何处理广告加载失败情况?

答：根据具体错误码处理。特别地，单次启动应用，如果加载广告失败，切莫即刻再发起请求（可做定时重试），除非是用户行为。