

# 金山云移动广告SDK Unity-Plugin-For-Android 快速接入文档 V4.0.3

## 更新日志

### 版本 4.0.3 [2018/3/9]

- 1.修改preloadAd接口，改名为loadAd接口
- 2.去除原有的hasLocalAd接口，后续统一使用hasAd接口

### 版本 4.0.2 [2018/3/5]

- 1.修复偶现的关闭广告按钮不出现问题
- 2.修复部分机型上，Home键退出后，返回APP时奖励视频播放退出问题。
- 3.修改FileProvider包名，避免集成其它SDK时发生冲突

### 版本 4.0.1 [2018/1/26]

- 1.新增hasLocalAd接口
- 2.新增沙盒环境

### 版本 4.0.0 [2017/12/15]

- 1.初版更新

## 目录

### 金山云移动广告SDK Unity-Plugin-For-Android快速接入文档 V4.0.3

#### 更新日志

#### 目录

- 1、文件清单
- 2、提供形式
- 3、Unity导出Eclipse工程相关说明
- 4、支持版本
- 5、环境说明
- 6、关于入口Activity的说明
- 7、SDK动态权限申请说明

- 8、快速集成
- 9、快速使用
  - 9.1、初始化及加载广告
  - 9.2、展示广告
- 10、高级用法
  - 10.1、SDK配置项
  - 10.2、广告事件监听
  - 10.3、广告资源加载事件监听

## 1、文件清单

SDK导入后，文件目录结构如下所示：

- Example目录，包含Demo相关的场景及脚本内容
- Plugins目录，包含AndroidManifest清单文件、sdk-aar库文件及support-v4包aar库文件
- Resources目录，主要是Demo相关图片资源
- Scripts目录，SDK脚本文件，其中KsyunAdSdk为对外提供的核心脚本类

客户可参考Example目录中，ExampleScript脚本调用KsyunAdSdk的方式，来进行SDK集成。

## 2、提供形式

SDK以unitypackage形式提供，内附简单的使用示例场景Example。

客户导入SDK Demo项目后，需将Unity编译环境切换至Android平台，然后导出对应Android Studio or Eclipse工程，编译之后运行。

## 3、Unity导出Eclipse工程相关说明

**注意，如果选择导出Eclipse工程形式，Unity默认会把所有的aar库，生成对应的Eclipse Library工程，客户需要导入并引用所有导出的Eclipse工程。此外，还需要额外做以下两件事：**

1、Unity导出eclipse工程，会把sdk-aar和support-v4导出成eclipse library，这两个库产生出来的jar包，都叫作class.jar，会产生冲突，需要手动改变其中一个的名称

2、因为在library工程中不能使用aidl文件，不能引用raw、assets下资源，需要手动拷贝步骤1中，导出所有library工程里的assets文件下所有内容，到主工程asstes目录下。

## 4、支持版本

**目前SDK只支持Unity5x以上版本，如果您使用的是Unity4x版本，建议直接使用Android SDK的jar包+asset资源文件形式集成。**

## 5、环境说明

- SDK分为沙盒环境(SANDOX\_ENV)及线上环境(RELEASE\_ENV)，默认会使用沙盒环境。
- 建议客户先使用沙盒环境进行开发联调，确认接口打通和数据无误后，再切换线上环境和对应线上AppId，进入生产环节。
- 沙盒环境的配置和线上环境基本保持一致，但考虑到测试方便性，广告请求方面会保证填充率，以便于测试阶段联调。
- SDK初始化前，可通过SDK配置项变更请求环境。
- 线上环境测试阶段，如果频繁遇到请求不到广告的错误码（2001），可能的原因有以下几种：
  - 1、当前广告请求的价格太低，导致竞价失败
  - 2、单设备请求超过限定频次
  - 3、线上广告没有余量问题

如需更详细的支持，请联系我方运营同学

## 6、关于入口Activity的说明

SDK默认会将入口Acitvity设置为KsyunAdSdkActivity。如果客户方有实现自己的入口Activity，那么请在AndroidManifest中注释掉与KsyunAdSdkActivity对应的...标签，并将以下代码添加至客户自己实现的Activity实现即可。

```
1.         protected void onPause() {
```

```

2.         KsyunAdSdk.getInstance().onPause(this);
3.         super.onPause();
4.     }
5.
6.     protected void onResume() {
7.         KsyunAdSdk.getInstance().onResume(this);
8.         super.onResume();
9.     }
10.
11.    protected void onDestroy() {
12.        KsyunAdSdk.getInstance().onDestroy(this);
13.        super.onDestroy();
14.    }

```

## 7、SDK动态权限申请说明

默认情况下，6.0以上系统，SDK内部会在初始化的时候，向APP申请以下动态权限。

如果APP方不希望SDK申请权限，后续说明中的SDK配置项中，有对应的开关可以关闭权限申请，

**对应的，APP应提供SDK运行所必须的动态权限，否则SDK将无法正常运行。**

- Manifest.permission.READ\_PHONE\_STATE（**必须**，用于生成唯一ID）
- Manifest.permission.ACCESS\_COARSE\_LOCATION,  
Manifest.permission.ACCESS\_FINE\_LOCATION(**非必须**，用于地理位置相关)

## 8、快速集成

1、双击sdk对应unitypackage文件，导入其中所有内容

2、打开Plugins/Android/AndroidManifest文件，修改其中标签中authorities属性，将com.xxx.xxx.xxx部分改为自己app的包名

```

1.    //此处添加provider，是为兼容7.0之后的应用下载完成后自动安装问题
2.    <provider
3.        android:name="com.ksc.ad.sdk.util.KsyunFileProvider"
4.        //注意下方的authorities中com.xxx部分取值，需要填写用户自己的包名
5.        android:authorities="com.xxx.xxx.xxx.fileprovider"
6.        android:exported="false"
7.        android:grantUriPermissions="true">
8.        <meta-data

```

```
9.         android:name="android.support.FILE_PROVIDER_PATHS"
10.         android:resource="@xml/file_paths"/>
11.     </provider>
```

### 3、导出对应的Android工程并运行

根据导出Android工程类型不同，分为以下两种情况。

#### 3.1、导出Android Studio工程的形式

修改SDK根目录下的res/xml/file\_paths.xml文件，将其中external-path标签中的path属性值，改为Android/data/YOUR\_APP\_PACKAGE\_NAME/，并将其放置在导出的Android Studio工程目录下的res文件夹下，然后运行即可。

```
1.     //注意下方的path取值，需要填写用户自己的包名
2.     <external-path path="Android/data/com.xxx.xxx.xxx/"
3.         name="files_root" />
4.     <external-path path="cache/apk/." name="external_storage_root" />
```

#### 3.2、导出Eclipse工程的形式

1、Unity会将sdk-aar以及support-v4-aar库，导出成Eclipse Library Project,将主工程导出为Eclipse Android Project。

2、首先，你需要在Eclipse中，将这三个工程同时导入，并且添加主工程对其它两个工程的依赖。

3、然后将sdk-aar工程中的assets文件夹，拷贝到主工程的assets目录下。

4、最后修改主工程的res/xml/file\_paths.xml文件，将其中external-path标签中的path属性值，改为Android/data/YOUR\_APP\_PACKAGE\_NAME/，并将其放置在导出的Android Studio工程目录下的res文件夹下，然后运行即可。

```
1.     //注意下方的path取值，需要填写用户自己的包名
2.     <external-path path="Android/data/com.xxx.xxx.xxx/"
3.         name="files_root" />
4.     <external-path path="cache/apk/." name="external_storage_root" />
```

## 9、快速使用

### 9.1、初始化及加载广告

如果没有调用KsyunAdSdkConfig的setSdkEnvironment()方法，设置SDK请求环境，默认则

为测试环境。

初始化成功之后，请在合适的时机，提前调用loadAd()获取广告。

```
1.      //初始化成功回调
2.      KsyunAdSdk.initSdkSuccess = (string param) => {
3.          Log ("KsyunAdSdk initSdkSuccess");
4.      };
5.      //初始化失败回调
6.      KsyunAdSdk.initSdkFailure = (string msg) => {
7.          Log ("KsyunAdSdk initSdkFailure, msg = " + msg);
8.      };
9.      //初始化方法
10.     KsyunAdSdk.init ("YOUR_APP_ID");
11.
12.     //在合适的场景下，提前加载广告
13.     void onLevelStart() {
14.         KsyunAdSdk.loadAd();
15.     }
```

## 9.2、展示广告

在广告位入口展示前，先调用hasAd方法判断当前广告位有无广告，根据结果决定是否展示入口。

广告展现后，待用户点击时，再调用showAd方法展示广告。

```
1.     void onGameOver() {
2.         // 判断本地是否有广告
3.         if (KsyunAdSdk.hasAd (adSlotId)) {
4.             //广告存在，点击奖励视频入口后，调用showAd接口展示广告
5.             //KsyunAdSdk.showAd (adSlotId);
6.         } else {
7.             //广告不存在，需要调用loadAd单个广告位接口进行加载
8.             Log ("KsyunAdSdk onNoAd, adSlotId = " + adSlotId);
9.             KsyunAdSdk.loadAd(adSlotId);
10.        }
11.    }
```

## 10、高级用法

### 10.1、SDK配置项

在调用init初始化方法之前，可以通过设置SDK配置项，来进行环境及功能的可选配置。**具体支持的配置项定义及说明详情，请参见SDK接口文档附录表。**

```
1.      //构建SDK配置类
2.      KsyunAdSdkConfig config = new KsyunAdSdkConfig ();
3.      //是否允许广告展现中途显示关闭按钮
4.      config.setShowCloseBtnOfRewardVideo (false);
5.      //设置SDK请求环境为沙盒环境(默认)
6.      config.setSdkEnvironment (SDK_ENV_SANDBOX);
7.      KsyunAdSdk.init ("YOUR_APP_ID", "YOUR_CHANNEL_ID_IF_NEEDED", config);
```

## 10.2、广告事件监听

通过设置广告事件监听相关回调方法，监听广告播放过程中的关键事件

```
1.      //广告展示成功时回调
2.      public static Action<string> onShowSuccess;
3.      //广告展示失败时回调
4.      public static Action<string> onShowFailed;
5.      //广告内容播放完成，一般用于视频类广告
6.      public static Action<string> onADComplete;
7.      //广告被点击
8.      public static Action<string> onADClick;
9.      //广告被关闭
10.     public static Action<string> onADClose;
```

对于奖励视频类型的广告，通过设置以下奖励视频结果相关回调方法，可以监听奖励条件是否达成回调。

```
1.      //奖励视频条件已达成
2.      public static Action<string> onADAwardSuccess;
3.      //奖励视频条件已失败
4.      public static Action<string> onADAwardFailed;
```

## 10.3、广告资源加载事件监听

对于视频类型的广告，可以监听广告加载相关回调方法

```
1.      //广告配置获取成功，注意此回调仅仅是广告配置获取成功，不代表广告加载完成
2.      public static Action<string> loadAdInfoSuccess;
```

```
3. //广告配置获取失败
4. public static Action<string> loadAdInfoFailure;
5. //广告加载完成
6. public static Action<string> onAdLoaded;
```