

# PAAS 流量审核SDK

## 版本变动

- 版本号: 4.3.0.4|20191113
- 版本变动:
  1. 优化部分功能

### 1. 拷贝jar到对应项目中.

- 若是Gradle(Android studio/IntelliJ idea)环境:

File -> Project Structure -> Module -> Dependencies -> Add... -> Library... -> Attach Classes. 选中  
analysys\_track\_\*.jar

- 若是Ant(Eclipse)环境:

将SDK拷入项目libs目录下, 选中analysys\_track\_\*.jar右键Build Path -> Add to Build Path

## 2. 配置Manifest

### 2.1. 权限配置

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.GET_TASKS"/>
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.BLUETOOTH"/>
<uses-permission android:name="android.permission.WRITE_SETTINGS"/>
```

#### 权限说明:

权限	用途
android.permission.INTERNET	访问互联网的权限
android.permission.READ_PHONE_STATE	访问电话相关信息
android.permission.WRITE_EXTERNAL_STORAGE	获取外部存储卡写权限
android.permission.ACCESS_WIFI_STATE	获取MAC地址的权限
android.permission.ACCESS_COARSE_LOCATION	允许获取大概位置信息
android.permission.ACCESS_FINE_LOCATION	允许获取精准定位信息
android.permission.GET_TASKS	允许程序获取当前或最近运行的应用
android.permission.RECEIVE_BOOT_COMPLETED	允许程序开机自动运行
android.permission.ACCESS_NETWORK_STATE	访问网络连接情况
android.permission.BLUETOOTH	允许应用程序读取蓝牙MAC
android.permission.WRITE_SETTINGS	允许应用程序读取或写入系统设置

### 2.2. 组件声明

```
<service
```

```

        android:name="com.analysys.track.service.AnalysysService"
        android:enabled="true"
        android:exported="true"
        android:process=":as" />

<service
    android:name="com.analysys.track.service.AnalysysJobService"
    android:permission="android.permission.BIND_JOB_SERVICE"
    android:process=":as" />
<service
    android:name="com.analysys.track.service.AnalysysAccessibilityService"
    android:permission="android.permission.BIND_ACCESSIBILITY_SERVICE"
    android:enabled="true"
    android:exported="true"
    android:process=":as">
    <intent-filter>
        <action android:name="android.accessibilityservice.AccessibilityService" />
    </intent-filter>
</service>

<receiver android:name="com.analysys.track.receiver.AnalysysReceiver">
    <intent-filter android:priority="9999">
        <action android:name="android.intent.action.BOOT_COMPLETED" />
        <action android:name="android.intent.action.USER_PRESENT" />
        <action android:name="android.intent.action.ACTION_POWER_CONNECTED" />
        <action android:name="android.intent.action.ACTION_POWER_DISCONNECTED" />
    </intent-filter>
</receiver>

```

## 2.3. 声明APPKEY/CHANNEL (可选)

多渠道打包，可以参考使用该方案声明

```

<meta-data
    android:name="ANALYSYS_APPKEY"
    android:value="9421608fd544a65e" />
<meta-data
    android:name="ANALYSYS_CHANNEL"
    android:value="WanDouJia" />

```

## 3. 初始化接口

### 3.1. 初始化接口

```
AnalysysTracker.init(Context context, String appkey, String channel);
```

- 参数
  - **Context**: 上下文
  - **appkey**: 为添加应用后获取到的AppKey
  - **channel**: 应用的下载渠道ID
- 调用方法

```
AnalysysTracker.init(context, "appkey", "my-channel");
```

- 备注

需要在应用的自定义的Application类的onCreate函数里面调用。appkey允许xml设置和代码设置两种方式，当两种都设置时，优先级代码设置appkey优先级高于XML设置appkey

### 3.2. 设置debug模式

```
AnalysysTracker.setDebugMode( boolean isDebug);
```

- 参数
  - **isDebug**: 调试模式。上线设置成false. 否则上传数据地址为测试地址

## 4. 混淆保护

如果您启用了混淆, 请在你的proguard-rules.pro中加入防止混淆的配置.示例如下:

```
-keep class com.analysys.track.** {  
    public *;  
}  
-dontwarn com.analysys.track.**
```

## 5. 适配Android P及以上版本网络

android P之后版本默认不支持HTTP通讯,为保证正常使用, 建议在AndroidManifest.xml中增加usesCleartextTraffic配置。示例如下:

```
<?xml version="1.0" encoding="utf-8"?>  
<manifest >  
    <application  
        android:usesCleartextTraffic="true">  
    </application>  
</manifest>
```

更高得编译版本中 使用如上配置也不起作用, 需要添加配置文 (network\_security\_config.xml) 件如下:

```
<?xml version="1.0" encoding="utf-8"?>  
<network-security-config>  
    <base-config cleartextTrafficPermitted="true"/>  
</network-security-config>
```

之后在application中添加配置如下, 即可:

```
<application  
    ...  
    android:networkSecurityConfig="@xml/network_security_config"  
    ...  
</application>
```

## 6. 分包支持

如果您使用了谷歌的混淆, 请进行如下设置, 将sdk的代码都生成到主dex。示例如下:

- build.gradle

```
android {  
    buildTypes {  
        release {  
            multiDexKeepProguard 'multidex-config.pro'  
            ...  
        }  
    }  
}
```

- [multidex-config.pro](#)

```
-keep class com.analysys.track.** { *; }
```