Avazu Native Ad SDK Setup for Android

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Developer |
| 2017.04.26 | V2.2.5 | Improve Appwall | 卢红伍 |
| 2017.07.04 | V2.2.6 | Support native AD | 卢红伍 |
|  |  |  |  |

目录

[1. 简介 3](#_Toc487724020)

[1.1 SDK功能介绍 3](#_Toc487724021)

[1.2 兼容性 3](#_Toc487724022)

[2. 集成准备 3](#_Toc487724023)

[2.1 APP ID 3](#_Toc487724024)

[2.2 Unit ID 3](#_Toc487724025)

[3. 初始化SDK 3](#_Toc487724026)

[3.1代码混淆配置： 4](#_Toc487724027)

[3.2初始化 5](#_Toc487724028)

[3.3 AndroidMenifest.xml 配置 5](#_Toc487724029)

[3.4添加权限（可选） 5](#_Toc487724030)

[4. 广告墙接入 6](#_Toc487724031)

[4.1 配置 6](#_Toc487724032)

[4.2 预加载应用墙数据 9](#_Toc487724033)

[4.3 启动market页面 10](#_Toc487724034)

[5. 原生广告接入 10](#_Toc487724035)

[5.1接入流程 11](#_Toc487724036)



1. 简介

1.1 SDK功能介绍

Avazu ADSDK 能够提供以下功能：

* AppWall （广告墙）
* Native (原生广告)

1.2 兼容性

支持Android 2.3（API level 9）及以上系统

2. 集成准备

2.1 APP ID

确保您已注册Avazu APX账号且已有可用的APP ID 用来展示广告

2.2 Unit ID

在Avazu APX创建一个广告位后，系统会自动生成Unit id

1. 初始化SDK

Avazu SDK可以从 Avazu 运营获取，SDK包含多个aar包

**aar包名** **作用** **是否必须**

* avazu\_common.aar 公共基础包 是
* avazu\_appwall.aar appwall广告的功能包 否
* avazu\_native.aar native广告功能包 否

将aar文件保存到本地工程中:

* 将所需要的aar包拷贝到项目的libs文件夹中，并作为项目的依赖库
* 在项目的build.gradle文件中的dependencies添加：

compile 'com.google.code.gson:gson:2.8.0' // 必须

compile(name: avazu\_common, ext: 'aar') // 必须

compile(name: avazu\_appwall, ext: 'aar') // 可选，用于 广告墙

compile(name: avazu\_native, ext: 'aar') // 可选，用于原生广告

compile 'com.google.android.gms:play-services-ads:8.4.0' // Admob广告，可选，用于native广告

compile 'com.facebook.android:audience-network-sdk:4.23.0' // Fb 广告，

可选，用于native广告，

3.1代码混淆配置：

在您的proguard文件中根据所集成的包来添加：

# FB 若没加，可不添加

-keep class com.facebook.\*\* {\*;}

-dontwarn com.facebook.\*\*

# Admob 若没加，可不添加

-keep class com.google.android.gms.\*\* {\*;}

-dontwarn com.google.android.gms.\*\*

## 3.2初始化

在Application的onCreate里面调用该初始化方法:

|  |
| --- |
| Adsdk.initialize(Context context, String appId); |

参数说明：

* context: Application 上下文
* appId: 申请的应用ID

## 3.3 AndroidMenifest.xml 配置

若集成了paly-services-ads, 请在AndroidMenifest.xml 里面添加：

|  |
| --- |
| <activity  android:name="com.google.android.gms.ads.AdActivity"  android:configChanges="keyboard|keyboardHidden|orientation|screenLayout|uiMode|screenSize|smallestScreenSize"  android:theme="@android:style/Theme.Translucent" />  <meta-data  android:name="com.google.android.gms.version"  android:value="@integer/google\_play\_services\_version" /> |

## 3.4添加权限（可选）

部分广告主需要SDK回传device id，建议开发者添加 android.persmission.READ\_PHONE\_STATE 便于SDK获取设备id。

# 4. 广告墙接入

App Market接口能够通过简单的设定，获取一个完整的广告商店的Activity或者Fragment，其界面如下：



4.1 配置

您可以根据需求配置App Market

1. 设置商店名称：

|  |
| --- |
| public static void setAppMarketName(Context context, String name) |

参数说明：

context: 上下文

name: 您所希望的商店名，将显示在商店的左上角，若不调用该接口，默认商店名为：“Market”

2. 配置广告墙样式

|  |
| --- |
| public static void setMarketStyle  (Context context, HashMap<String, Integer> marketStyle) |

参数说明：

context: 上下文

marketStyle : 广告墙样式配置，可以配置项如下（若某一项不设置或设置为0，表示该项用默认配置）：

TITLE\_BACKGROUND\_COLOR 标题背景颜色

TITLE\_TEXT\_COLOR 标题字体颜色

TITLE\_BAR\_HEIGHT 标题栏高度

TABLE\_BACKGROUND\_COLOR TAB标签背景颜色

TABLE\_TEXT\_COLOR TAB标签字体颜色

TABLE\_BAR\_HEIGHT TAB栏 高度

TABLE\_INDICATOR\_COLOR TAB指示器颜色

DK\_BUTTON\_BACKGROUND\_COLOR 大卡按钮背景颜色

DK\_BUTTON\_TEXT\_COLOR 大卡按钮字体颜色

INSTALL\_TEXT\_BACKGROUND\_DRAWABLE 安装按钮自定义图片

INSTALL\_TEXT\_COLOR 安装按钮字体颜色

WALL\_STATUS\_COLOR 状态栏颜色

WALL\_NAVIGATION\_COLOR 导航栏颜色

BACK\_BUTTON\_DRAWABLE 返回按钮自定义图片

APPWALL\_BACKGROUND\_COLOR 应用墙内容背景颜色

SUBTITLE\_TEXT\_COLOR 广告分类标题字体颜色

AD\_TITLE\_TEXT\_COLOR 广告标题字体颜色

AD\_DESCRIPTION\_TEXT\_COLOR 广告描述字体颜色

AD\_CLICK\_COVER\_LAYER\_TRANS\_BACKGROUND 广告点击时蒙层的颜色

使用示例：

|  |
| --- |
| Protected void onCreate(Bundle savedInstanceState) {  setMarketStyle();  }  private void setMarketStyle() {  marketStyle.put(Constants.MarketStyle.TITLE\_BACKGROUND\_COLOR, R.color.white);  marketStyle.put(Constants.MarketStyle.TITLE\_TEXT\_COLOR, R.color.gray);  marketStyle.put(Constants.MarketStyle.TABLE\_BACKGROUND\_COLOR, R.color.white);  marketStyle.put(Constants.MarketStyle.TABLE\_TEXT\_COLOR, R.color.gray);  marketStyle.put(Constants.MarketStyle.DK\_BUTTON\_BACKGROUND\_COLOR, R.color.black);  marketStyle.put(Constants.MarketStyle.DK\_BUTTON\_TEXT\_COLOR, R.color.white);  marketStyle.put(Constants.MarketStyle.INSTALL\_TEXT\_BACKGROUND\_DRAWABLE,  R.drawable.apx\_appwall\_adress\_button\_type1);  marketStyle.put(Constants.MarketStyle.INSTALL\_TEXT\_COLOR, R.color.gray);  marketStyle.put(Constants.MarketStyle.BACK\_BUTTON\_DRAWABLE, R.drawable.apx\_appwall\_adrss\_ic\_back);  marketStyle.put(Constants.MarketStyle.WALL\_STATUS\_COLOR, 0);  marketStyle.put(Constants.MarketStyle.WALL\_NAVIGATION\_COLOR, 0);  marketStyle.put(Constants.MarketStyle.APPWALL\_BACKGROUND\_COLOR, R.color.white);  marketStyle.put(Constants.MarketStyle.TITLE\_BAR\_HEIGHT, 0);  marketStyle.put(Constants.MarketStyle.TABLE\_BAR\_HEIGHT, 0);  marketStyle.put(Constants.MarketStyle.TABLE\_INDICATOR\_COLOR, R.color.light\_gray);  marketStyle.put(Constants.MarketStyle.SUBTITLE\_TEXT\_COLOR, R.color.gray);  marketStyle.put(Constants.MarketStyle.AD\_TITLE\_TEXT\_COLOR, R.color.gray);  marketStyle.put(Constants.MarketStyle.AD\_DESCRIPTION\_TEXT\_COLOR, R.color.light\_gray);  marketStyle.put(Constants.MarketStyle. AD\_CLICK\_COVER\_LAYER\_TRANS\_BACKGROUND, R.color.cover\_layer\_background);  AdSdk.setMarketStyle(this, marketStyle);  } |

4.2 预加载应用墙数据

为提高market页面的加载速度，以及优化广告的填充和转化，强烈建议开发者预加载应用墙的数据：

|  |
| --- |
| public static void preloadMarketData (Context context) |

参数说明：

* context: 上下文，建议使用ApplicationContext

使用示例：

|  |
| --- |
| @Override  protected void onResume() {  super.onResume();  preloadMarketWall();  }  private void preloadMarketWall() {  AdSdk.preloadMarketData(this.getApplicationContext());  } |

4.3 启动market页面

有两种方式启动market页面：

1. 用Activity启动：

在需要打开App market处调用该接口

|  |
| --- |
| public static void showAppMarket(Context context) |

2. 用Fragment启动：

在打开market fragment前需要调用

|  |
| --- |
| public static void setMarketFragmentMode(Context context, boolean isFragmentMode) |

参数说明：

context: 上下文

isFragmentMode: 设为true则为fragment接入模式

在需要获取Fragment对象的地方用如下方法获取

|  |
| --- |
| Fragment fr = AdSdk.getFeatureFragment(Context context); |

5. 原生广告接入

集成后的可能效果



## 5.1接入流程

请将avazu\_common.aar, avazu\_native.aar复制到相应位置

1. 确保SDK已经在Application中初始化  
2. 若集成第三方广告源， 请在build.gradle文件中增加相应依赖：

compile 'com.google.android.gms:play-services-ads:8.4.0' // AdMob

compile 'com.facebook.android:audience-network-sdk:4.23.0' // Fb

3. 初始化和预加载

构建一个NativeAd 对象，并传入该广告位unit ID，设置广告加载监听；展示广告之前，提前将广告拉到本地，提升用户体验，提高广告收入

|  |
| --- |
| INativeAd nativeAd = new NativeAd(Context activityContext, String unitId);  nativeAd.setNativeAdListener(new NativeAdListener() {  @Override  public void onAdLoaded() { // 广告加载成功回调  }  @Override  public void onError(String error) {  }  })  mNativeAd.load(); // 加载广告 |

4. 显示广告

当广告预加载成功后，调用接口show展示广告；adContainer 是用来展示原生广告的容器。

|  |
| --- |
| if (nativeAd.isLoaded()) {  nativeAd.show(ViewGroup adContainer); // 显示广告  } |

广告容器配置参考：

|  |
| --- |
| <FrameLayout  android:id="@+id/native\_container"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"/> |