Avazu Native Ad SDK Setup for Android

Date	Version	Description	Developer
2015.07.17	v1.3.1	Add Function	曹诗聪
2015.08.27	v1.3.3	Add new parameter for C2S API	曹诗聪
2015.11.28	v2.0.0	Add App Market and News Feed	曹诗聪
2015.12.21	v2.0.2	Fix issue	曹诗聪
2015.2.2	v2.1.0	Add Facebook Ad in News and fix issue	曹诗聪

1.简介

1.1 SDK功能介绍

Avazu ADSDK 能够提供以下功能

- 展示单个广告banner（推荐尺寸： 320 * 100)
- 展示透明banner (推荐尺寸：400 * 110)
- 展示广告banner墙
- 展示单行矩形广告墙
- 展示多行矩形广告墙
- 通过C2S接口获取原始广告数据(包括Google Play广告， DDL广告以及全屏订阅类广告)
- 通过DirectToMarket接口, 随机提供一个Google Play地址进行跳转
- 通过showAppMarket接口，提供Appstore页面
- 通过showNewsFeed接口，提供带广告的新闻页面

以上这些广告展现方式，开发者可以根据SDK提供的方法实现广告外观和展示应用数量的定制

1.2 兼容性

支持Android 2.3及以上系统

2. 接入配置

2.1 Traffic Source ID

确保您已注册Avazu APX账号且已有可用的Traffic Source ID for Display

2.2 选择SDK文件，导入第三方广告SDK

文件清单：

- adsdk_2.0.2_with_fb.jar
- adsdk_2.0.2.jar

Facebook:

若您的项目尚未使用facebook广告库，请用adsdk_2.0.2_with_fb.jar

若你的项目已经使用facebook广告库，请使用adsdk_2.0.2.jar，并确保您使用的是最新的版本的AudienceNetwork.jar

Admob:

若您的项目尚未使用admob广告，请在项目中添加对google service的依赖，方法如下：

Gradle:

1. 确认您的Android SDK中， extra下的Google Repository是最新版本

2. 在项目的build.gradle文件中的dependencies添加：

compile ‘com.google.android.gms:play-services-ads:8.3.0’

注：若您已使用play-services，请不要重复添加

Eclipse:

1. 确认您的Android SDK中， extra下的Google Play Service是最新版本

2. 新建一个项目，从现有代码创建项目，路径为：

<android-sdk>/extras/google/google_play_services/libproject/google-play-services_lib/

3. 在您项目的属性 -> android菜单中，添加对刚刚新建项目的依赖

详细文档：

https://developers.google.com/mobile-ads-sdk/docs/admob/android/quick-start

2.3 导入SDK文件

- 将adsdk_2.1.0_with_fb.jar或adsdk_2.1.0.jar拷贝到您项目的libs文件夹中，并作为您项目的依赖库
- 将SDK所提供的assets文件夹中的所有文件拷贝到您项目的assets文件夹
- 添加对android-support-v4的依赖，若已经依赖该库请勿重复添加

2.4 代码混淆配置

在您的proguard文件中添加：

```
-keep class com.facebook.**{*;}
-keep class  com.google.ads.**{*;}
-keep class nativesdk.ad.adsdk.**{*;}
```

2.5 配置AndroidManifest.xml

为了保证SDK能够正确运行，您需要在AndroidManifest中添加如下权限：

```
<uses-permissionandroid:name="android.permission.INTERNET" />
<uses-permissionandroid:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permissionandroid:name="android.permission.READ_PHONE_STATE"/>
<uses-permissionandroid:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permissionandroid:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permissionandroid:name="android.permission.READ_EXTERNAL_STORAGE"/>
```

在<application>中添加如下Activity

```
<activity
    android:name="natives dk.ad.adsdk.modules.activityad.MarketActivity"
    android:configChanges="orientation | keyboardHidden | screenSize"
    android:screenOrientation="portrait"/>

<activity
    android:name="natives dk.ad.adsdk.modules.activityad.AvLoadingActivity"
    android:configChanges="orientation | keyboardHidden | screenSize"
    android:screenOrientation="portrait"/>

<activity
    android:name="natives dk.ad.adsdk.modules.activityad.rss.NewsDetailActivity"
    android:configChanges="orientation | keyboardHidden | screenSize"
    android:screenOrientation="portrait"/>

<activity
    android:name="natives dk.ad.adsdk.modules.activityad.rss.NewsActivity"
    android:configChanges="orientation | keyboardHidden | screenSize"
    android:screenOrientation="portrait"/>

<activity
    android:name="com.facebook.ads.InterstitialAdActivity"
    android:configChanges="orientation | keyboardHidden | screenSize"
    android:exported="true"
    android:excludeFromRecents="true"
    android:noHistory="true"
    />
```

若选用创建Market桌面快捷方式接口，还需添加

```
<uses-permission
android:name="com.android.launcher.permission.INSTALL_SHORTCUT" />

<uses-permission
android:name="com.android.launcher.permission.READ_SETTINGS" />

<activity
    android:name="natives dk.ad.adsdk.modules.activityad.AvLoadingActivity"
    android:configChanges="orientation"
    android:screenOrientation="portrait">
    <intent-filter>
    <action android:name="natives dk.ad.adsdk.modules.activityad.
     AvLoadingActivity.SHORT_CUT" />
    <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

3. 展示广告

AvazuADSDK 提供了5种不同种类的Webview广告展现形式，包括单个广告banner，广告banner墙单行矩形广告墙， 多行矩形广告墙，透明banner。

您可以选择通过以下步骤展示广告：

3.1 初始化SDK

在应用启动的地方调用该初始化方法：

```
Adsdk.initialize(Context context);
```

3.2 创建AdView

创建Adview步骤如下：

1. 生成AvazuAdView

您可以通过以下方法生成AvazuAdView：

```
AvazuAdView adView =new AvazuAdView(Context context, String adType,
String sourceId, int width, int height)
```

参数说明：

- context: 上下文
- sourceId: 用来展示该广告的Traffic Source Id
- width:广告空间的宽度（单位为dip）
- height:广告空间的高度（单位为dip）
- show_type： 广告的展现形式，有以下五种：

广告形式	变量名
单个广告banner	AvazuAdView.SINGLE_BANNER
banner广告墙	AvazuAdView.MULTIPLE_LINE_BANNER
透明banner	AvazuAdView.SINGLE_TRANSPARENT_BANNER
单行矩形广告墙	AvazuAdView.SINGLE_LINE_RECTANGLE
多行矩形广告墙	AvazuAdView.MULTIPLE_LINE_RECTANGLE

2. 设置广告自定义参数

详情请参照3.3

3. 加载 AvazuAdView

在需要显示广告的时候用此方法加载广告:

```
adView.loadWebviewAd(Context context);
```

4.设置广告加载结果监听

调用方法:

```
adView.setAdViewStateListener(AdViewStateListener l);
```

回调方法说明:

```
public void onLoadAdStart(AdView view);
```

说明:在开始加载广告内容时会被回调

```
public void onLoadAdFinish(AdView view);
```

说明:在广告内容加载完成后会被回调

```
public void onLoadAdError(AdView view, String error);
```

说明:在加载广告内容时,出现错误时会被回调,error为错误信息

3.3 自定义AdView

AvazuADSDK支持丰富的定制化广告,您可以选择展示的广告的颜色搭配,展示内容以及展示应用的数量,只需要像如下的代码进行设置就可:

```
adView.setAdIconVisibility(bolean isVisible)
```

1.广告元素配置

方法名	变量类型	作用
setAdIconVisibility	boolean	设置广告中是否显示应用图标
setAdTitleVisibility	boolean	设置广告中是否显示应用名称
setAdCatagoryVisibility	boolean	设置广告中是否显示应用类别
setAdSizeVisibility	boolean	设置广告中是否显示应用尺寸
setAdRatingVisibility	boolean	设置广告中是否显示应用评分
setAdInstallButtonVisibility	boolean	设置广告中是否显示安装按钮
setAdReviewNumberVisibility	boolean	设置广告中是否显示应用查看次数
setAdInstallNumberVisibility	boolean	设置广告中是否显示应用安装次数
setAdLoadingIndicatorVisibility	boolean	设置广告是否显示加载页面

2. 广告颜色配置

方法名	类型	作用
setAdBlockBackgroundColor	String	设置广告块背景色
setAdTitleColor	String	设置应用名字体颜色
setAdInstallButtonBackgroundColor	String	设置下载按钮颜色
setAdInstallButtonTextColor	String	设置下载按钮上的字体颜色
setAdMainBackgroundColor	String	设置应用墙背景色

注意: 配置广告颜色时，所有颜色值的设置方式为 “#”加上6位十六进制数RGB格式的字符串，如“#FFFFFF”表示白色，“#000000”表示黑色

3. 展示应用数量配置

方法名	类型	作用
setAdNumber	int	设置广告中出现的应用数量

注意: 以下广告类型必须设置appCount：广告banner墙，单行矩形广告墙，多行矩形广告墙

4. 设置广告透明度

变量名	类型	作用
setTransparentBannerAlpha	int	设置透明广告的透明度

注意: 只有在透明banner的广告形式下才用设置transparentBannerAlpha, 合法取值范围0~100,0表示完全透明，100表示完全不透明

以下是一个定制banner的完整代码:

```
AvazuAdView adView = new AvazuAdView(this,
AvazuAdView.SINGLE_LINE_RECTANGLE, "15887", showWidthDip, showHeightDip);
adView.setAdCatagoryVisibility(false);
adView.setAdSizeVisibility(false);
adView.setAdNumer(6);
adView.loadWebviewAd(this);
```

4. C2S接口

您可以通过以下方式使用C2S接口:

```
AdSDK.getAdRawData(Context context, final String sourceId,
String excludePackages, int limitNumber, int creatives, String market,
FetchRawDataListener listener)
```

参数说明:

- Source ID: Traffic Source ID
- excludePackages: 传入不想显示的广告的 campaign_id, 用于翻页时不显示重复广告,若有多个以“,”分隔,若无则传入“”,例如 : “6184, 3241”
- limitNumber: 本次想要拉取广告的个数的上线
- creatives: 是否返回多尺寸图片素材,值为 1 表示返回 图片素材,默认为 0(不返回图片素材)。 可能返回的图片尺寸有:

尺寸	参数名
320x50	banner
320x480	phone_fullscreen
480x320	phone_fullscreen_landscape
1024x768	tablet_fullscreen
768x1024	tablet_fullscreen_landscape
300x250	medium
728x90	leaderboard
160x600	skyscraper
1200x627	content_stream_image

- market：选择获取Rawdata的广告类型，具体如下，若传入其他值默认传入googlebet
- listener：获取RawData的回调，返回RawData， 有以下可重写方法：

public void onLoadRawDataStart():
说明：开始加载RawData，在主线程中执行

public void onLoadRawDataSuccess(List<FetchAdResult.Ad> data):
说明：加载RawData结束，回传RawData，在主线程中执行

public void onLoadRawDataFail(Error mError):
说明：加载RawData失败，在主线程中执行

FetchAdResult.Ad为获取的广告数据，其数据结构如下：

字段	描述
campaignid	广告 ID
payout	一个转化的价格(CPI),单位为美元
pkgname	App 的包名(package name)
title	App 的标题
description	App 的描述(不超过 100 个英文字母)
icon	App 的 icon 地址(100x100 的 png 图片)
appcategory	App 分类

apprating	App 的评分(0-5 星)
appreviewnum	App 的评论数
appinstalls	App 的安装量
appsize	App 的安装文件大小
creatives	App 的多尺寸图片素材地址
clkurl	广告的点击 URL

注：在获取creatives中您想要的尺寸的url数组前，需进行判空处理，范例如下：

```
if (datadata.tablet_fullscreen != null) {
    L.d(datadata.tablet_fullscreen);
}
```

5. DirectToMarket

DirectToMarket接口能够实现挑选一个随机广告直接跳转Google Play页面，具有高度自定义性，请在您应用中自定义的广告跳转按钮按下时调用。

您可以通过以下方式使用 DirectToMarket 接口：

```
public static void directToMarket(Context context, String souceId,
DirectToMarketManager.DirectToMarketListener listener)
```

参数说明：

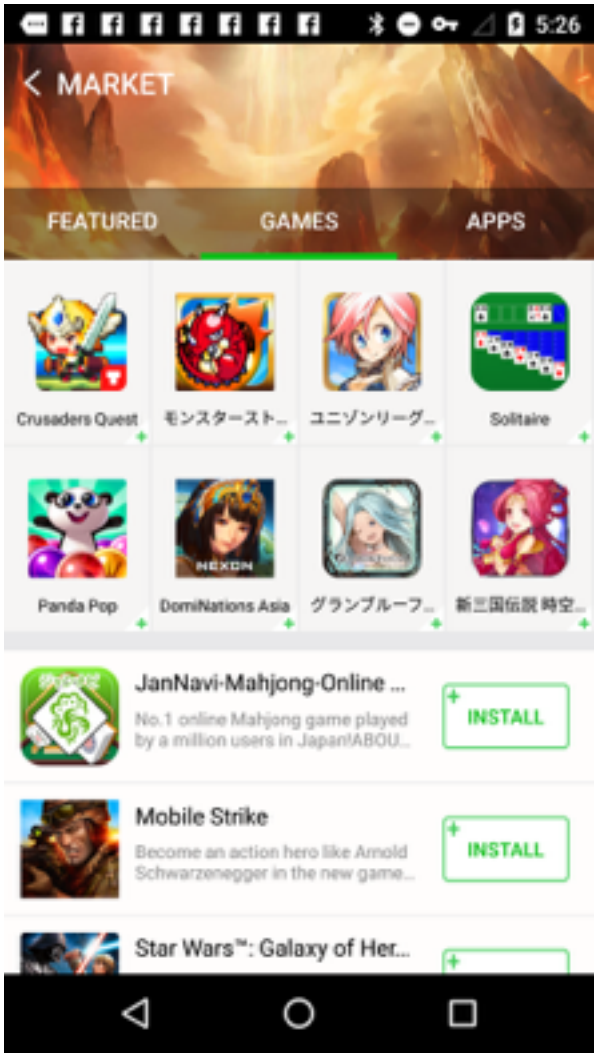
- Source ID: Traffic Source ID
- listener: 用来监听跳转Google Play的结果

public void DirectToMarketSuccess()：跳转成功

public void DirectMarketToFail()：跳转失败

6. App Market

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：



6.1 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. 激活market页面的facebook广告

```
public static void enableFacebookAdInMarket(Context context,
String placementId)
```

参数说明：

- context: 上下文
- placementId: 您申请的facebook广告placement Id

注：调用该接口请确保工程已使用最新的AudienceNetwork.jar，或使用adsdk_2.0_with_fb.jar，请参考本文2.2部分

3. 激活market页面的admob广告

```
public static void enableAdmobInMarket(Context context,
String unitId)
```

参数说明：

- context: 上下文
- unitId: 您申请的admob广告unitId

注：调用该接口请确保工程已添加对最新版Google Service的依赖，请参考本文2.2部分

4. 创建market桌面快捷方式

```
public static void createMarketShortcut
(Context context, Bitmap marketIcon, String marketName)
```

参数说明：

- context: 上下文
- marketName: Bitmap格式的桌面图标
- marketName: 桌面快捷方式的名字

自定义图片：

您可以从SDK文件里提供的九张图片中选择适合自己的页面头图，放入工程assets目录下即可

若您想自己设计图片并使用，您的图片的命名请与SDK文件提供的图片保持一致，包括图片的后缀格式，并把它们放入到工程assets目录下

6.2 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, String marketSourceId)
```

参数说明：

- context: 上下文
- marketSourceId: 市场页面的Traffic Source Id

注： 请确保您在应用中已经调用 Adsdk.initialize 接口

2. 用Fragment启动：

在打开market fragment前需要调用

```
public static void setMarketSourceId(Context context, String marketSourceId)
```

参数说明：

- context: 上下文
- marketSourceId: 市场页面的Traffic Source Id

```
public static void setMarketFragmentMode(Context context,
boolean isFragment)
```

参数说明：

- context: 上下文
- isFragmentMode: 设为true则为fragment接入模式

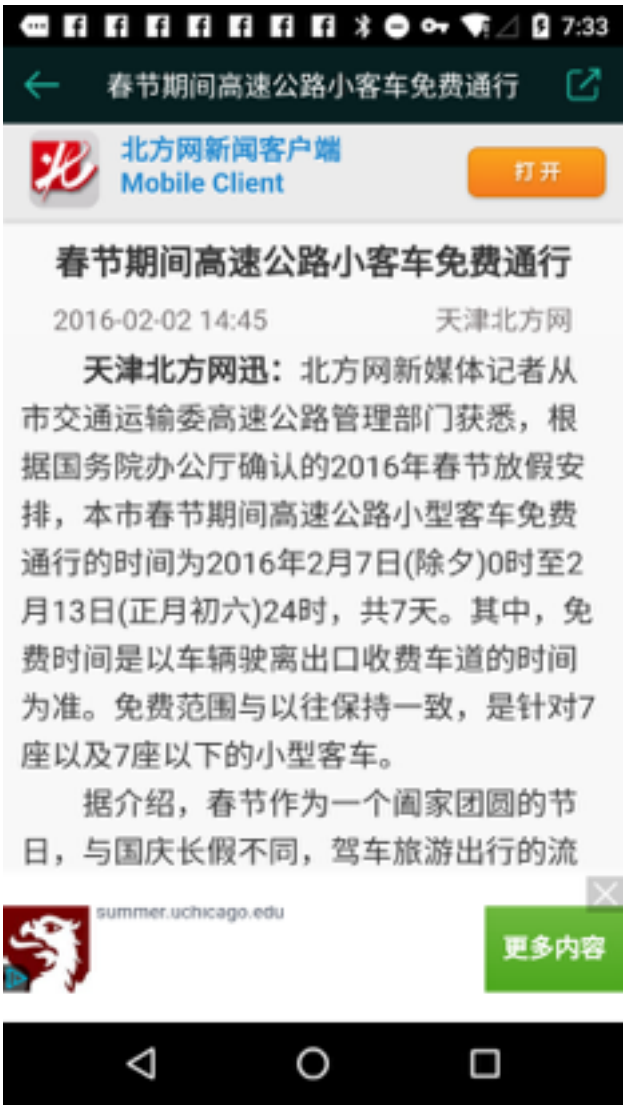
在需要获取Fragment对象的地方用如下方法获取

```
Fragment fr = new FeatureFragment();
```

7. News Feed

News Feed接口能够通过简单的设定，获取一个完整的带广告的新闻页Activity或者Fragment，其界面如下：

新闻列表页



新闻详情页

7.1 配置

1. 激活新闻页面的Facebook Native广告

```
public static void enableFacebookNativeAdInNewsFeed(Context context,
String placementId)
```

参数说明:

- context: 上下文
- placementId: 您申请的facebook广告placement Id

2. 激活新闻页面的facebook banner广告

```
public static void enableFacebookBannerInNewsFeed(Context context,
String placementId)
```

参数说明:

- context: 上下文
- placementId: 您申请的facebook广告placement Id

3. 激活新闻页面的facebook 全屏广告

```
public static void enableFacebookInterstitialInNewsFeed(Context context,
String placementId)
```

参数说明:

- context: 上下文
- placementId: 您申请的facebook广告placement Id

注：调用以上接口请确保工程已使用最新的AudienceNetwork.jar，或使用adsdk_2.0_with_fb.jar，请参考本文2.1部分

4. 激活新闻页面的APX Native广告

```
public static void enableApxNativeAdInNewsFeed(Context context,
String newsSourceId)
```

参数说明:

- context: 上下文
- newsSourceId: 您申请的APX广告的source id

注：调用以上接口请确保工程已使用最新的AudienceNetwork.jar，或使用adsdk_2.0_with_fb.jar，请参考本文2.1部分

7.2 启动News Feed页面

有两种方式启动News Feed页面

1.用Activity启动

```
public static void showNewsFeed(Context context,  boolean newTask)
```

参数说明:

- context: 上下文
- newTask: 是否从Activity外部启动News Fragment, 若是则设为true

注: 请确保您在应用中已经调用 Adsdk.initialize 接口

2.用Fragment启动

在打开News fragment前需要调用

```
public static void setNewsFeedFragmentMode(Context context,
boolean fragmentMode)
```

参数说明:

- context: 上下文
- fragmentMode: 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 = new NewsTabFragment()
```