DU Ad Platform_SDK for Android Access Guide

Version: DUAd_SDK_CW1.2.7.4

1. Introduction

This document describes how to integrate **DUAd Platform SDK** into Android apps.

DAP(short for DUAd platform) offers advertising services for helping apps to monetize. This version of SDK provides native ads, interstitial ads, offerwall and banner.

Prerequisites:

DU Ad Platform SDK currently supports Android 2.3 API level 9 (included) plus system versions.

2. Integration Workflow

This section describes the integration workflow of DU Ad Platform SDK.

1. Apply for App_ID and DAP Placement_ID.
2. Load SDK package; configure Androidmanifest.xml.
3. Initialize DU Ad Platform SDK.
4. Access Du ads.

3. Obtain Identity

This section describes the IDs needed during DU Ad Platform SDK integration: APP ID, DAP Placement ID.

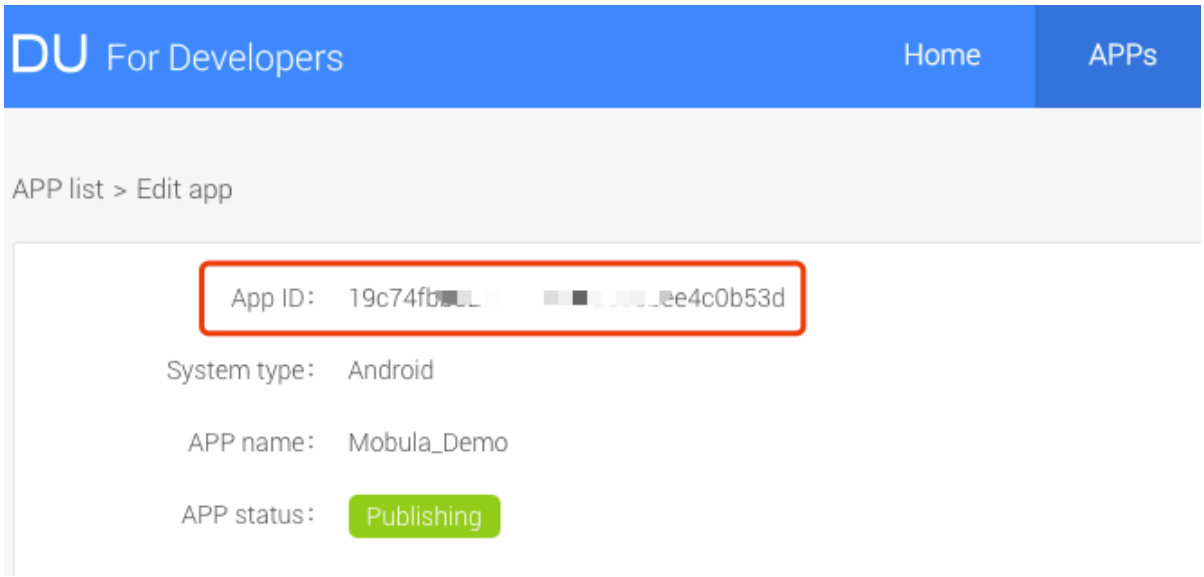
3.1 APP_ID

1. Definition

APP ID is a unique identifier of a developer's APP on **Du Ad Platform**. Each app has its own App ID.

2. Obtain method

   Visit our official website and register your app on **Du Ad Platform**, the APP ID will be generated automatically



3. Code

   `app_license`

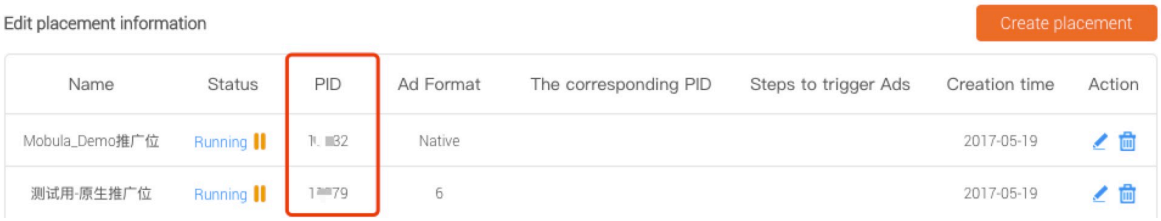
3.2 DAP Placement ID

1. Definition

   DAP Placement ID is a unique identifier of an ad slot on Du Ad platform. Developers can create multiple DAP Placement IDs for one app.

2. Obtain method

   Visit our official website and after registered your app, you can create the placement for your app.



3. Code

   `pid`

4. Load SDK and Configuration

This section describes how to load the DU Ad Platform SDK into your android project, how to configure the *AndroidManifest.xml* file and how to obfuscate code against project needs.

4.1 Load DU Ad Platform SDK

1. Download the DU Ad Platform SDK package.
2. Unzip the package. Two folders are available in the subdirectory:
   - DUAd_SDK:

     This folder stores DU Ad Platform SDK aar: DuappsAd-CW-xxx-release.aar
   - DUAd_SDK_DEMO

     This folder stores a sample program, which integrates DU Ad Platform SDK. All interfaces in this document can be found in corresponding usage in this sample program.
3. Load DU Ad Platform SDK:
   - When using Android Studio:

     Copy the SDK aar to your Android Project, under the libs directory in root directory. Then configure build.gradle:

     ```
     repositories {
       flatDir {
         dirs 'libs'
       }
     }

     dependencies {
       compile fileTree(include: ['*.jar'], dir: 'libs')
       compile(name: 'DuappsAd-CW-xxx-release', ext: 'aar')
     }
     *Note: The assigned directory of flatDir is where the aar file is placed.
     ```

   - When using Eclipse:

     1. Create a new Eclipse Android library.
     2. Change the suffix of DuappsAd-CW-xxx-release.aar to zip and unzip it, then you will get a classes.jar, an AndroidManifest.xml and a res folder.
     3. Copy the classes.jar to the new created Android library, under the libs directory in root directory.
     4. Replace the AndroidManifest.xml in the new created Android library with the AndroidManifest.xml in unzipped DuappsAd-CW-xxx-release.aar
     5. Replace the res folder in the new created Android library with the res folder in unzipped DuappsAd-CW-xxx-release.aar.

4.2 Configure AndroidManifest.xml

Open the AndroidManifest.xml in your Android project and update it as below:

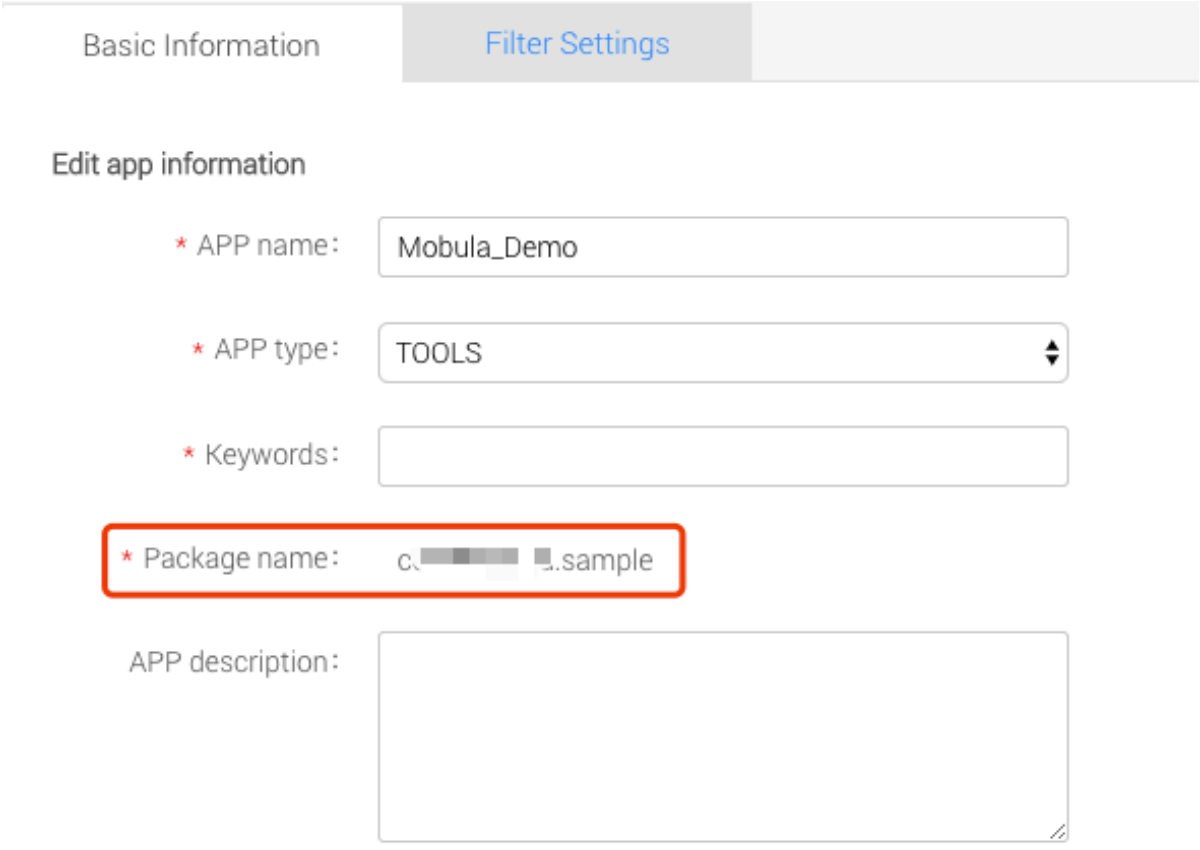
1. Add user-permission element. Least Privilege of DU Ad Platform SDK is shown below:

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

2. Add a meta-data element under the application element, and fill your DAP App ID as the value of "app_license". Declare the com.duapps.ad.stats.DuAdCacheProvider in the manifest. Replace the below packagename with your app's full package name.

```
<application
             android:name="com.mobula.sample.MobulaApplication"
             android:usesCleartextTraffic="true"  // Required for target
SDK 28
             android:icon="@drawable/ic_launcher"
             android:label="@string/app_name"
             android:theme="@style/mobulaTheme" >
  <meta-data
           android:name="app_license"
           android:value="YOUR_APP_LICENSE" />
</application>
```

Note: Please make sure the applicationId in build.gradle is exactly the same as the package name you filled on DAP when registering you app. Otherwise, it will fail to get ad from DAP.



3. For **Android 9.0 (Target SDK 28)**, need to add **android:usesCleartextTraffic="true"** under application tag to solve http URL.

4.3 Obfuscate Code

Please follow the below rules to obfuscate code. Otherwise, there might be exceptions at run time.

1. Add below classes to proguard configuration:

```
-keep class com.duapps.ad.**{*;}
-dontwarn com.duapps.ad.**

-keep public class * extends android.content.BroadcastReceiver
-keep public class * extends android.content.ContentProvider
-keepnames @com.google.android.gms.common.annotation.KeepName class *
-keepclassmembernames class * {
        @com.google.android.gms.common.annotation.KeepName *;}
-keep class com.google.android.gms.common.GooglePlayServicesUtil {
      public <methods>;}
-keep class com.google.android.gms.ads.identifier.AdvertisingIdClient {
      public <methods>;}
-keep class com.google.android.gms.ads.identifier.AdvertisingIdClient$Info
{
      public <methods>;}
```

   Note: For more about obfuscation methods, please refer to the official Android obfuscation document at: ${ android-sdk }/tools/proguard/

4.4 Kotlin configuration (Optional)

For Kotlin application integration, there are extra steps to do.

1. Add to build.gradle of the app:

```
apply plugin: 'kotlin-android'
apply plugin: 'kotlin-android-extensions'

..........

dependencies{
    ............
    implementation "org.jetbrains.kotlin:kotlin-stdlib-
jre7:$kotlin_version"
}
```

2. Add to build.gradle of the project:

```
buildscript {
  xt.kotlin_version = '1.2.51'
  ..........
  dependencies {
        classpath "org.jetbrains.kotlin:kotlin-gradle-
plugin:$kotlin_version"
        ............
  }
}
```

5. Initialization

This section describes how to initialize DAP SDK. You need to initialize DAP SDK before you can use it.

Placement id without initialization can not get ads.

1. Create a json file with mappings for the DAP Placement ID and other platform id.

```
{
  "native": [
    {
      "pid":"YOUR_DAP_PLACEMENT_ID"
    },
    {
      "pid":"YOUR_DAP_PLACEMENT_ID"
    }
  ],
  "list": [
    {
      "pid": "YOUR_DAP_PLACEMENT_ID"
    }
  ]
  "offerwall": [
     {
     "pid": "YOUR_DAP_PLACEMENT_ID"
     }
 ]
}
```

   Note: If you do not want to initialize by creating a json file statically, you can directly create a string that conforms to the json format and pass the value.

2. Add a call to `DuAdNetwork.init()` from `onCreate()` in your Application class.

   Interface Instruction:

   > public static void init(Context context, String pidsJson)

Parameters	Description
Context context	ACTIVITY CONTEXT
String pidsJson	The relationship between DAP Placement ID and other platform.

Java Code Sample:

```
public void onCreate() {
  super.onCreate();
  //Initialize the DAP SDK before executing any other operations
  DuAdNetwork.init(this, getConfigJSON(getApplicationContext()));

  //DuAdNetwork.setLaunchChannel("YOUR_APP_CHANNEL");
}

//Read the json.txt from assets
private String getConfigJSON(Context context) {
  BufferedInputStream bis = null;
  ByteArrayOutputStream bos = new ByteArrayOutputStream();
  try {
    bis = new BufferedInputStream(context.getAssets().open("json.txt"));
    byte[] buffer = new byte[4096];
    int readLen = -1;
    while ((readLen = bis.read(buffer)) > 0) {
      bos.write(buffer, 0, readLen);
    }
  } catch (IOException e) {
    Log.e("", "IOException :" + e.getMessage());
  } finally {
    closeQuietly(bis);
  }

  return bos.toString();
}
```

Kotlin Code Sample:

```
override fun onCreate() {
    super.onCreate()
    DuAdNetwork.setConsentStatus(this, true)
    DuAdNetwork.getConsentStatus(this)
    /**
    * the sdk initialization 初始化SDK
    */
    DuAdNetwork.init(this, getConfigJSON(getApplicationContext()))
}
```

```
fun getConfigJSON(context: Context): String {
    var bos = ByteArrayOutputStream()
    var bis = BufferedInputStream(context.assets.open("json.txt"))

    try {
        var readLen = -1
        while (bis.read().also { readLen = it } != -1) {
            bos.write(readLen)
        }

    } catch (e: Exception) {
        Log.e("", "IOException :" + e.message)
    } finally {
        closeQuietly(bis)
    }
    return bos.toString()
}
```

3. Add a `DuAdNetwork.setLaunchChannel()` from `onCreate()` in you Application class, to distinguish your data by your app distribution channel. It's optional.

   Interface Instruction:

   > public static void setLaunchChannel (String channelName)

Parameters	Description
String channelName	Your own distribution channel name to distinguish your data.

6.Matain the status of user's collecting consent

This is an optional configuration for GDPR compliance.

6.1 Set the status of user's consent

Please set the status during the initialization.

Interface Instrunction：

> public static void setConsentStatus(Context context, boolean consentStatus)

Parameters	Description
Context context	ACTIVITY CONTEXT
boolean consentStatus	User's consent status

6.2 Get the status of user's consent

Interface Instrunction：

```
public static boolean getConsentStatus(Context context)
```

Obtain the status of user's consent. Return True if user's consent is obtained, otherwise return False.

7. Request Single Native Ad

7.1 Construct Du Native Ad

Proceed as follows：

1. Create DuNative Ad Object.

   Must specify the corresponding pid into the object. You will get different ad data with different pid.

2. Set ad cache size

   Cache size could be set to 1-5. Recommend not to set cachesize. The default cachesize will be 1.

   Note:Cache size only takes effect when integrating other platform through the DU Ad Platform.

   Interface Instruction:

   ```
   public DuNativeAd (Context context, int pid)
   public DuNativeAd (Context context, int pid, int cacheSize)
   ```

Parameters	Description
Context context	ACTIVITY CONTEXT
int pid	DAP placement ID, this pid must declared on Json's **native** array
int cacheSize	Ad cache size.

7.2 Register the Callback Interface for Native Ad

Please register a callback interface for receiving the native ad data.

Interface Instruction:

```
public void setMobulaAdListener(DuAdListener adListener)
```

Parameters	Description
DuAdListener adListener	Callback function returns: ad error, ad data, and ad click event.

```
public interface DuAdListener  {
  public void onError(DuNativeAd ad, AdError error);
  public void onAdLoaded(DuNativeAd ad);
  public void onClick(DuNativeAd ad);
}
```

After called `load()`, three types of results could be returned:

- Retrieve ad successfully

  Modify the `onAdLoaded()` function above to retrieve the ad properties.

- Get an error

  Get specific error information in `onError()` function above. Error code and description are shown as below:

Constants	Error Code	Description
NETWORK_ERROR_CODE	1000	Client network error
NO_FILL_ERROR_CODE	1001	No Ad data retrieved
LOAD_TOO_FREQUENTLY_ERROR_CODE	1002	Too many interface requests
IMPRESSION_LIMIT_ERROR_CODE	1003	Reach the daily impression limit
SERVER_ERROR_CODE	2000	Server error
INTERNAL_ERROR_CODE	2001	Network error
TIME_OUT_CODE	3000	Retrieve Ad data timed out
UNKNOW_ERROR_CODE	3001	Unknown error

- Retrieve a ad click event

  Get informed when an ad is clicked in `onClick()` function.

7.3 Retrieve Native Ad

The ad retrieving process is asynchronous, so it will not block developers' threads.

Interface Instruction:

> public void fill()

Use the `fill()` to pre-cache ad in advance for faster loading the ad when using `load()`. Recommend using the `fill()` at the page before the ad showing page.

Note: Ad data will be cached in client device's memory. Since SDK only caches the image's URL address not the image data, the cache size is small.

> public void load()

Acquire advertising object data **asynchronously**, making an ad request when there is no cache.

Recommend using `fill()` after `load()` to pre-cache again.

> public DuNativeAd getCacheAd()

Acquire advertising object data **synchronously**. It could be traversed until the number of cached ads goes to 0.

Please make sure the cache pool is not null before showing ad.

Recommend using `fill()` after `get()` to pre-cache again.

> public boolean isHasCached()

Check if there is cached ad. Return true for having cache.

Java Code Sample:

```
DuNativeAd nativeAd = new DuNativeAd(this, PID, CACHESZIE);

if (nativeAd != null) {
  nativeAd.setMobulaAdListener (mListener);
  nativeAd.load();
}

DuAdListener mListener = new DuAdListener () {
  @Override
  public void onError (DuNativeAd ad, AdError error) {
    Log.d(TAG, "onError : " + error.getErrorCode());
}

  @Override
  public void onClick (DuNativeAd ad) {
    Log.d(TAG, "onClick : click ad");
  }

  @Override
  public void onAdLoaded (final DuNativeAd ad) {
    Log.d(TAG, "onAdLoaded : " + ad.getTitle());
  }
};
```

Kotlin Code Sample:

```
lateinit var nativeAd: DuNativeAd
nativeAd = DuNativeAd(this, PID, DEFAULT_CACHE_SIZE);
nativeAd.setMobulaAdListener(mListener)
nativeAd.load()

var mListener = object : DuAdListener {
    override fun onClick(p0: DuNativeAd?) {
        Log.d(TAG,  "onClick")
    }

    override fun onError(p0: DuNativeAd?, p1: AdError?) {
        Log.d(TAG,  "onError")
    }

    override fun onAdLoaded(ad: DuNativeAd?) {
        Log.d(TAG,  "onAdLoaded")
    }
}
```

7.4 Destroy Ad Object

Recommend to destroy your ad object when exiting the native ad showing page.

Interface Instruction:

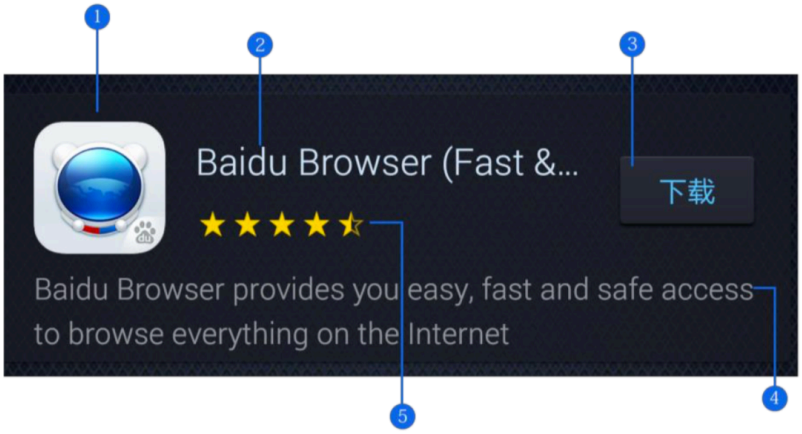
> public void destroy()

8. Native Ad Properties

When using the Native Ad, instead of receiving an ad ready to be displayed, you will receive a group of ad properties such as a title, an image, a call to action, and you will have to use them to construct a custom view where the ad is shown. This section describes the ad properties and how to get them.

8.1 Introduction of Ad Properties

Ad properties include: Icon, title, Call to action (CTA) button, short description, rating, promotion image, etc.



1. Icon 2. Title 3. Call to action (CTA) button 4. Short description 5. Rating

8.2 Get the Ad Properties

The interfaces for retrieving the ad properties as shown below:

- Get Icon

  > public String getIconUrl()

  Return the URL address of icon.

- Get Title

  > public String getTitle()

  Return the title of ad.

  Please reserve at least 20 charactors' space to display the title. An ellipsis (...) can be used to indicate truncated text.

  Note: The ad title must be included in your native ad design.

- Get Call to Action (CTA) button

  > public String getCallToAction()

  Return the text of ad’s CTA button.

  Advertisers can specify the text of CTA button, e.g. Install Now. The max character length is 25. Please do not shorten or change the text. Note: The CTA button must be included in your native ad design.

- Get Short Description

  > public String getShortDesc()

  Return the short description of ad.

  Please reserve atleast 72 charactors' space to display the short description. If the space isnot big enough, it is recommended to use scrolling text effects, or do notdisplay the short description.

- Get Rating

  > public float getRatings()

  Return the ad's rating on Google Play.

- Get Promotion Image

  > public String getImageUrl()

  Return the URL address of ad'spromotion image.

  A promotion image can increase user's desire to click the ad. The image size is usually 796x416 pixels(1.91:1). You can zoom and cut part of the image, but do not distort or change it.

9. Register the Native Ad's View

The SDK will log the impression and handle the click automatically. Please note that you must register the ad's view with the DuNativeAd instance to enable that.

Interface Instruction:

> public void registerViewForInteraction(View view)
>
> public void registerViewForInteraction(View view, List<View> views)

Parameters	**Description**
View view	Clickable View in Adcontents
List <View> views	More detailed sub-View

Note: Don't recommend using this interface in multi-thread.

10. Request Native Ad List

Du Native Ad List is for showing multiple ads in one page atthe same time. (Please note that Du Native Ad List has relatively poormonetization efficiency compared with single Du Native Ad. Please use this accordingto your situation.)

The whole workflow of getting the Ad is done in AsyncTask.Please use this function in the main thread.

10.1 Construct Manager Class of Native Ad List

Interface Instruction:

> public DuNativeAdsManager(Context context, int pid, int cacheSize)

Parameters	Description
Context context	ACTIVITY CONTEXT
int pid	DAP placement ID, this pid must declared on Json's **list** array
int cacheSize	Ad cache size.

10.2 Construct Class of Sub-Native Ad

> public NativeAd()

10.3 Register the Listener for Manager Class

Interface Instruction:

> public void setListener(AdListArrivalListener adListener)

Parameters	Description
AdListArrivalListener adListener	Listener for NativeAd list. Callback function returns: ad error and ad data.

```
public interface AdListArrivalListener  {
  public void onError(AdError error);
  public void onAdLoaded(List<NativeAd> mNativeAd);
}
```

After called `load()`, two types of results could be returned:

- Retrieve ad successfully

Modify the `onAdLoaded()` function above to retrieve the ad properties.

- Get an error

  Get specific error information in `onError()` function above.

10.4 Register a listener for each single ad in ad List

Interface Instruction:

> public void setMobulaAdListener(DuAdDataCallBack mCallBack)

Parameters	Description
DuAdDataCallBack mCallBack	Callback function returns: click event. There is no `onAdLoaded()` and `onAdError()` callback for singe ad in ad list.

```
public interface DuAdDataCallBack  {
  public void onError(AdError error);
  public void onAdLoaded(NativeAd mNativeAd);
  public void onAdClick();
}
```

- Retrieve a ad click event

  Get informed when an ad is clicked in `onClick()` function.

Java Code Sample:

```
private NativeAd mNativeAD;
private LinkedList<NativeAd> lists = new LinkedList<NativeAd>();
private DuNativeAdsManager adsManager = = new
DuNativeAdsManager(getApplicationContext(), PID, CACHESZIE);

@Override
protected void onResume() {
  super.onResume();
  if (adsManager != null) {
    adsManager.setListener(listener);
    adsManager.load();

    mNativeAD = lists.get(mPositon);
    mNativeAD.setMobulaAdListener(callback);
    mNativeAD.registerViewForInteraction(btnView);
  }
}
```

```
AdListArrivalListener listener = new AdListArrivalListener() {
  NativeAd nativeAD;
  //Return ad list
  @Override
  public void onAdLoaded(List arg0) {
    for (int i = 0; i < arg0.size(); i++) {
      //Get single ad object
      nativeAD = (NativeAd) arg0.get(i);
      if (!(nativeAD.equals(null))) {
        lists.add(nativeAD);
      }
    }
  }

  //Get the error
  @Override
  public void onAdError(AdError arg0) {
    Log.d(TAG, "onError : " + arg0.getErrorCode());
  }
};

DuAdDataCallBack callback = new DuAdDataCallBack() {
  @Override
  public void onAdLoaded(NativeAd data) {
  }

  @Override
  public void onAdError(AdError error) {
  }

  @Override
  public void onAdClick() {
    Log.d(TAG, "onClick : click list ad");
  }
};
```

Kotlin Code Sample:

```
lateinit var mDuNativeAdsManager: DuNativeAdsManager
val mAdList = arrayListOf<NativeAd>()
lateinit var mNativeAD: NativeAd

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    mDuNativeAdsManager = DuNativeAdsManager(this, PID, CACHESIZE).apply {
        setListener((object : AdListArrivalListener {
      override fun onAdLoaded(p0: MutableList<NativeAd>?) {
                Log.d(TAG, "onAdLoaded")
                mAdList?.clear()
```

```
                p0?.forEach {
                    if (it != null) mAdList.add(it)
                }
                if (mAdList.size == CACHESIZE) {
                    mHandler.apply {
                        removeCallbacksAndMessages(null)
                        post(mRunnable)
                    }
                }
            }
      override fun onAdError(p0: AdError?) {
                Log.d(TAG, "onError : " + p0?.getErrorCode());
            }

        }))
    }

    Log.d(TAG, "load list ad....")
    mDuNativeAdsManager.load()
}

private val mRunnable = object : Runnable {
    override fun run() {
        if (mPositon < mAdList.size) {
            mNativeAD = mAdList.get(mPositon);
            mNativeAD.setMobulaAdListener(callback);
            val url = mNativeAD.adCoverImageUrl
            if (url.isEmpty()) showSmallAdView(mNativeAD) else
showBigAdView(mNativeAD)

            mPositon++

        } else {
            mPositon = 0
        }
        mHandler.postDelayed(this, 8000)
    }
}

var callback = (object : DuAdDataCallBack {
    override fun onAdClick() {
        Log.d(TAG, "onClick : click list ad");
    }

    override fun onAdLoaded(p0: NativeAd?) {
        Log.d(TAG, "onAdLoaded: adloead list ad")
    }

    override fun onAdError(p0: AdError?) {
```

```
        Log.d(TAG, "onAdError: onAdError list ad:" + p0.toString())
    }
})
```

10.5 Retrieve Native Ad

Interface Instruction:

> public void fill()

Use the `fill()` to pre-cache ad in advance for faster loading the ad when using `load()`. Recommend using the `fill()` at the page before the ad showing page.

Note: Ad data will be cached in client device's memory. Since SDK only caches the image's URL address not the image data, the cache size is small.

> public void load()

Acquire advertising object data **asynchronously**, making an ad request when there is no cache.

Recommend using `fill()` after `load()` to pre-cache again.

10.6 Get the Ad Properties

- Get Icon

  > public String getAdIconUrl()

  Return the URL address of icon.

- Get Title

  > public String getAdTitle()

  Return the title of ad.

  Please reserve at least 20 charactors' space to display the title. An ellipsis (...) can be used to indicate truncated text. Please note the ad title must be included in your native ad design.

- Get Call to Action (CTA) button

  > public String getAdCallToAction()

  Return the text of ad's CTA button.

  Advertisers can specify the text of CTA button, e.g. Install Now. The max character length is 25. Please do not shorten or change the text. Please note the CTA button must be included in your native ad design.

- Get Short description

  > public String getAdShortDesc()

  Return the short description of ad.

  Please reserve atleast 72 charactors' space to display the short description. If the space isnot big enough, it is recommended to use scrolling text effects, or do notdisplay the short description.

- Get Rating

  > public float getAdRatings()

  Return the ad's rating on Google Play.

- Get Promotion Image

  > public String getAdImageUrl()

  Return the URL address of ad'spromotion image.

  A promotion image can increase user's desire to click the ad. The image size is usually 796x416 pixels(1.91:1). You can zoom and cut part of the image, but do not distort or change it.

10.7 Destroy the Object and Listener Interface

When exiting the native ad list showing page, the object(DuNativeAdsManager) and listener(AdListArrivalListener) must be destroyed.

Interface Instruction:

> public void destroy()

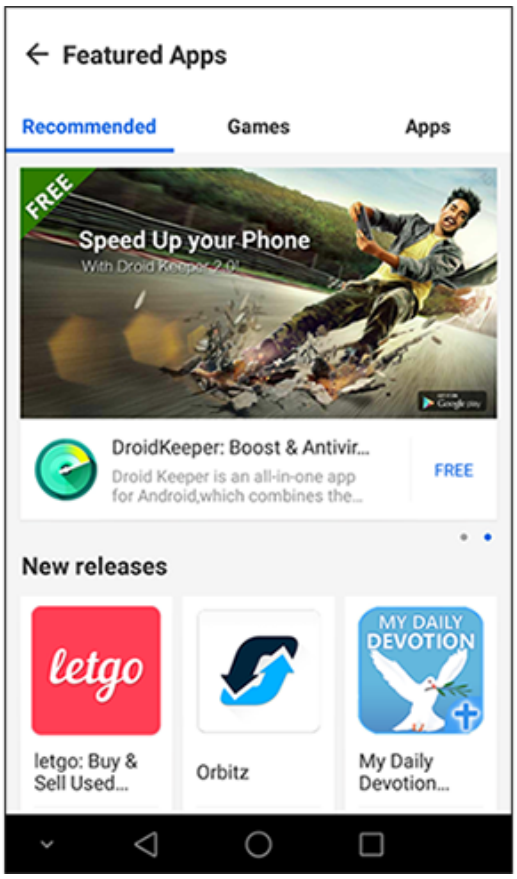
Java Code Sample:

```
@Override
protected void onDestroy() {
  super.onDestroy();
  adsManager.setListener(null);
  adsManager.destroy();
}
```

Kotlin Code Sample:

```
override fun onDestroy() {
  super.onDestroy()
  mDuNativeAdsManager.apply {
        setListener(null)
        destroy()
    }
}
```

11. Request Offerwall



11.1 Configure AndroidManifest

Open AndroidManifest.xml and add the below code.

```
<activity
    android:name="com.duapps.ad.offerwall.ui.OfferWallAct"/>
```

11.2 Mandatory Parameter

> OfferWallAct.KEY_PID

OfferWallAct.class is the encapsulated activity class of offer wall. Please fill in your DAP placement ID before using it. this pid must declared on Json's **offerwall** array.

Please make sure the Ad Format of this DAP placement ID is selected as『Offerwall』during creation on du ad platform. Otherwise, it will fail to get ad.

> OfferWallAct.KEY_TITLE_ID

Customize your offerwall title text, The id of string defined in xml file. The default value is "Featured Apps" or the corresponding language text.

> OfferWallAct.KEY_TAB_BACKGROUND_COLOR

Customize the background color of title and tab. e.g #FFFFFF. The default value is #EDEDED.

> OfferWallAct. KEY_TAB_INDICATOR_COLOR

Customize the color of selected tab text. e.g #FFFFFF. The default value is #1C86EE.

> OfferWallAct.KEY_TAB_TEXT_COLOR

Customize the color of title text and the color of unselected tab text. e.g #FFFFFF. The default value is #000000.

Java Code Sample:

```
Intent intent = new Intent(MainActivity.this, OfferWallAct.class);

Bundle b = new Bundle();
b.putInt("pid", YOUR_PID);
b.putInt(OfferWallAct.KEY_TITLE_ID, R.string.app_name); //Optional
b.putString(OfferWallAct.KEY_TAB_BACKGROUND_COLOR, "#EDEDED"); //Optional
b.putString(OfferWallAct.KEY_TAB_INDICATOR_COLOR, "#1C86EE"); //Optional
b.putString(OfferWallAct.KEY_TAB_TEXT_COLOR, "#000000"); //Optional

intent.putExtras(b);
startActivity(intent);
```

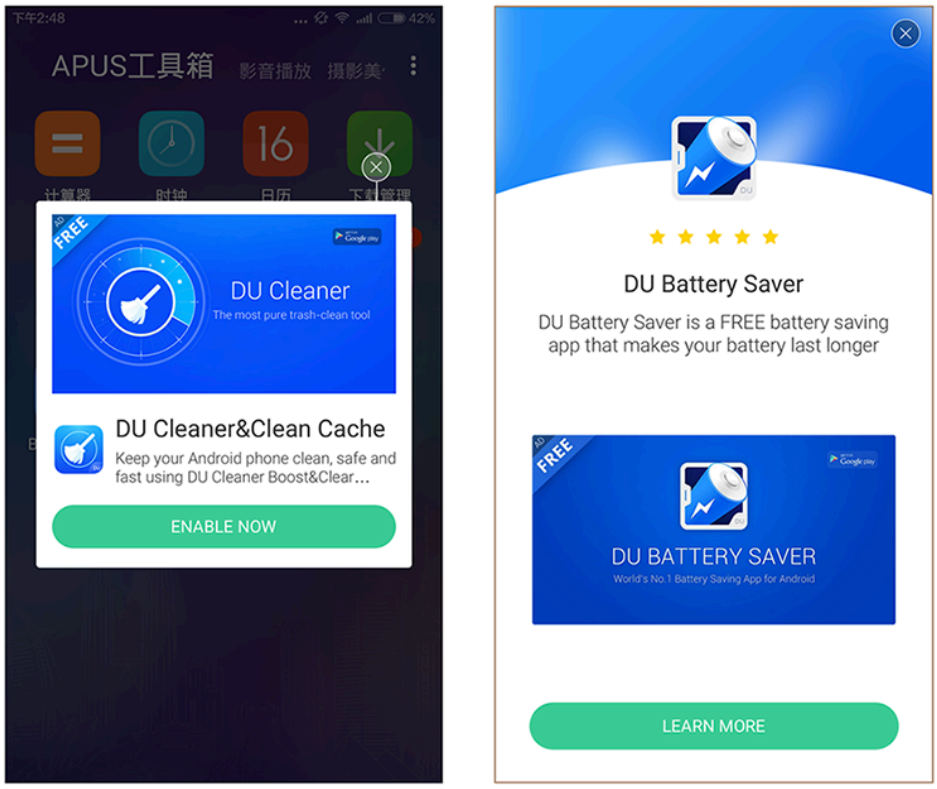
kotlin Code Sample:

```
var offerWallIntent = Intent(this, OfferWallAct::class.java)
var bundle = Bundle()
bundle.apply {
    putInt("pid", 61709);
    putInt(OfferWallAct.KEY_TITLE_ID, R.string.app_name); // 可选
    putString(OfferWallAct.KEY_TAB_BACKGROUND_COLOR, "#EDEDED"); // 可 选
    putString(OfferWallAct.KEY_TAB_INDICATOR_COLOR, "#1C86EE"); // 可选
    putString(OfferWallAct.KEY_TAB_TEXT_COLOR, "#000000"); // 可选
}

offerWallIntent.putExtras(bundle)
startActivity(offerWallIntent)
```

12. Request Interstitial Ad

A sample of interstitial ad:



12.1 Constructor of Interstitial Ad

Interface Instruction:

> public InterstitialAd(Context context, int pid, int type)

Parameters	Description
Context context	ACTIVITY CONTEXT
int pid	DAP placement ID, this pid must declared on Json's **native** array
int type	`InterstitialAd.Type.SCREEN` for fullscreen ad `InterstitialAd.Type.NORMAL` for half screen ad The default value is half screen

12.2 Set listener for Interstitial Ad

Interface Instruction:

> public void setInterstitialListener (AbsInterstitialListener adListener)

Parameters	Description
AbsInterstitialListener adListener	Callback function returns: ad error, ad data, and ad click event.

```
public interface AbsInterstitialListener  {
  //Retrieve ad failed
  public void onAdFail(int errorCode);

  //Retrieve ad successfully
  public void onAdReceive();

  //Ad destroyed event
  public void onAdDismissed();

  //Ad impression event
  public void onAdPresent();

  //Ad click event
  public void onAdClicked();
}
```

12.3 Retrieve Interstitial Ad

Interface Instruction:

> public void fill()

Use the `fill()` to pre-cache ad in advance for faster loading the ad when using `load()`. Recommend using the `fill()` at the page before the ad showing page.

Note: Ad data will be cached in client device's memory. Since SDK only caches the image's URL address not the image data, the cache size is small.

> public void load()

Acquire advertising object data **asynchronously**, making an ad request when there is no cache.

Recommend using `fill()` after `load()` to pre-cache again.

> public void show()

Show interstitial ad, Please use this interface in `onAdReceive()`.

> ~~public void close()~~

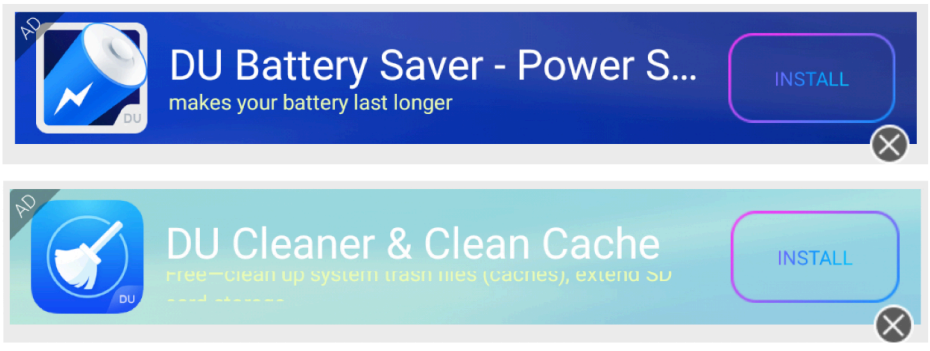
Close interstitial ad, **this interface is disabled in this version.**

> public void destroy()

Destroy interstitial ad, Recommend to destroy your ad object when exiting the ad showing page.

13. Request Banner Ad

A sample of banner ad (blue and green background)



Banner Size: 320 x 50 px

13.1 Constructor of Banner Ad

Interface Instruction:

> public BannerAdView(Context context, int pid, int cachesize, BannerType BType, BannerListener listener)

Parameters	Description
Context context	ACTIVITY CONTEXT
int pid	DAP placement ID, this pid must declared on Json's **native** array
int cachesize	Ad cache size.
BannerType	Use TYPE_CPM to get CPM offers
BannerListener listener	BannerAd listener (Only callback for the first time `Load()` result for each BannerAdView)

13.2 Set listener for Banner Ad

Interface Instruction:

> public BannerListener ()

```
public interface BannerListener  {
  //Retrieve ad failed
  public void onError(String msg);

  //Retrieve ad successfully
  public void onAdLoaded();
}
```

Java Code Sample:

```
BannerAdView mBannerAdView = new BannerAdView(this, PID, CACHESIZE, new
BannerListener() {
    @Override
    public void onAdLoaded() {
        Log.d(TAG, "onAdLoaded");
    }
    @Override
    public void onError(String msg) {
        Log.d(TAG, "onError:" + msg);
    }
});
```

Kotlin Code Sample:

```
mBannerAdView = BannerAdView(this, PID, 5, (object : BannerListener {
    override fun onError(p0: String?) {
        Log.d(TAG, "onError")
    }

    override fun onAdLoaded() {
        Log.d(TAG, "onAdLoaded")
    }
}))
```

13.3 Mandatory Parameter

> public void setBgStyle(int BannerStyle)

Parameters	Description
Int BannerStyle	Set background color `BannerStyle.STYLE_BLUE`: for blue background `BannerStyle.STYLE_GREEN`:for greeen background;

> public void setCloseStyle(int BannerCloseStyle)

Parameters	Description
Int BannerCloseStyle	Setthe position of close button: `BannerCloseStyle.STYLE_BOTTOM`: bottom right corner `BannerCloseStyle.STYLE_TOP`: top right corner

13.4 Add Banner Ad to Custom Layout

```
YourLayout.addView(mBannerAdView);
```

13.5 Retrieve Banner Ad

> public void load()

Acquire advertising object data **asynchronously**, making an ad request when there is no cache.

> public void destroy()

Destroy banner ad, Recommend to destroy your ad object when exiting the ad showing page.

14 Frequently asked questions

14.1 SDK integration

- Q: Where to find the latest SDK?

A: http://ad.duapps.com/en/sdk/

- Q: There are basic SDKs and extended SDKs, which ones should I use?

  A: Select according to the ad formats you need. Basic SDK is mandatory, extended SDK is optional.

  - Eg1: Only native ad needed, please use Basic SDK HW
  - Eg2: Native, interstitial and video are needed, please use basic SDK CW and extended SDK Video
- Q: Can I initialize json serveral times?

  A: Yes, you can. But last intialization will overwrite the previous one. So please make sure you put all necessary pids in the json file/string when you do initialization. If you use video ad also, DuVideoSDK.init() must contain all necessary pids, too.
- Q: Will SDK re-load ads automatcally when it returns error?

  A: No, it won't. Please retry when it returns error. But do NOT retry under onError() callback. Because it may cause a dead loop and crash the app.
- Q: Why the click rate of my native ad is very low?

  A: Click is user's behavior which we have no control of. We suggest to improve it by registering more ad elements to clickable. At the mean time, the design of you native ad card and the scenario of display also affect a lot the click rate. Please contact us for more optimization options.
- Q: Why the callback of native ad doesn't function?

  A: There is a one-to-one correspondence between native object instance and native listener. Please make sure you bind a new listener when you create a new native object. Even for the same ad placement, if the native object is changed, a new listener must be bound.

14.2 Developer dashboard

- Q: Why the status of my app is alway "Not activated"?

  A: "Not activated" means the app is already approved and able to receive ads. When the app successfully displayed an ad, the status will change to "Published" automatically after around 15 mins.
- Q: Is the data updated in real-time?

A: No, the data is not in real-time. Please check the data of previous days. In the calendar, we define 00:00-23:59 of UTC/GMT+08:00 as one day.

- Q: Why the data on the dashboard is different with my counting?

  A: For each request, SDK may cache several ads. Before the cached ads are expired/showed, SDK won't send new request, even load() is called. So the requests are lower than your counting if you consider load() as request. And the impressions could be more than fills.

- Q: Can I set filters for the ads?

  A: Yes, you can find the filter setting tab under the app detail page. You can set the filter by package name/genre/age.

14.3 Advertise

- Q: Return error code 1001 when loading the ad.

  A: Please ensure that the package name, app license and pid are configured right totally. Take **4. Load SDK and Configuration** as reference.

- Q: The ad can’t be redirected.

  A: Please ensure the test device has the whole GMS.

14.4 Others

- Q: Why the revenue is 0 or too low even though the app has the impressions?

  A: The ads from DAP are counted by CPA for now, so that the revenue will be brought up by user’s action after the impression, such as click, installation or activation. For the app who is accessed recently, we suggest increasing DAU to advance the optimization. After that, it will take a period of time for the system to make the optimization. Please contact us if you still have question of revenue.

- Q: Is the mediation supported?

  A: DAP doesn't support mediation. If the ads from other platform are needed, we suggest applying Admob as the mediation and check Du Ad Platform in dash board.