Trigger SDK for Android Access Guide

Du_Trigger-release-1.0

Contents

1. Obtain Identity 2
2. Load SDK and Configuration 2
    2.1 Load DU Ad Platform SDK 3
    2.2 Configure AndroidManifest.xml 3
    2.3 Obfuscate Code 5
3. Initialization 6
4. Request Trigger ad 7
    4.1 Trigger:AutoStart 8
    4.2 Trigger: Delay 8

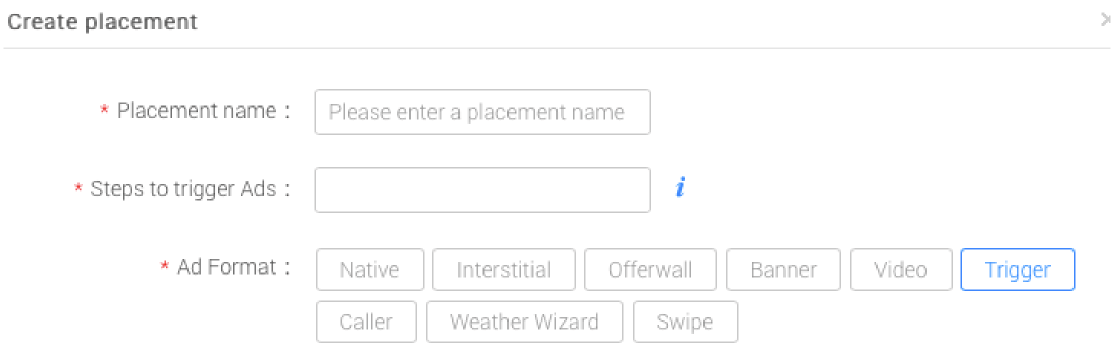
Precondition:

Currently DU Weather Wizard SDK must rely on DU Ad Platform_SDK CW or HW SDK.

1. Obtain Identity

Please refer to the chapter 3 in HW or CW version of DUADplatform SDK Access Guide to obtain necessary identities.

When applying for the DAP Placement ID , please make sure the app format you choose is 【Trigger】.


Create placement

* Placement name :	Please enter a placement name
* Steps to trigger Ads :	
* Ad Format :	Native, Interstitial, Offerwall, Banner, Video, [✓ Trigger], Caller, Weather Wizard, Swipe

2. Load SDK and Configuration

2.1 Load DU Ad Platform SDK

Please refer to the chapter 4.1 in HW or CW version of DUADplatform SDK Access Guide to load DU SDK.

For accessing Weather Wizard, the below operations are needed.

1) Copy the DuappsAd-Trigger-xxx.aar to your Android Project, under the *libs* directory in root directory
2) Then configure build.gradle：

```
repositories {
        flatDir {
                dirs 'libs'
        }
}
dependencies {
        compile fileTree(include: ['*.jar'], dir: 'libs')
        compile(name: 'DuappsAd-xW-xxx-release', ext: 'aar')      // Trigger SDK must
rely on DU AD SDK
        compile(name: ' DuappsAd-Trigger-xxx ', ext:'aar')
}
```

2.2 Configure AndroidManifest.xml

Please refer to the chapter 4.1 in HW or CW version of DUADplatform SDK Access Guide to configure AndroidManifest.xml

A. Add a user-permission element to the manifest. For accessing Weather Wizard, except the basic permissions:

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

B. Add a meta-data element to the application element, and fill your DAP App ID as the value of “app_license”.

```
    <application
        android:name=" Your_PackageName.MobulaApplication"
        android:label="@string/app_name"
        . . . >
        <meta-data
            android:name="app_license"
            android:value="@string/Your_DAP_APP_ID" />
   </application>
```

C. Declare the com.duapps.ad.stats.DuAdCacheProvider in the manifest. Replace the below packagename with your app's full package name. Please make sure the package name at here 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.

```
<provider
    android:name="com.duapps.ad.stats.DuAdCacheProvider"
    android:authorities="Your_packagename.DuAdCacheProvider"
    android:exported="false">
 </provider>
```

D．Register the BroadcastReceiver for receiving app install event.

Register the PACKAGE_ADDED Receiver in AndroidManifest.xml. Otherwise, it might affect your monetization efficiency.

```
    <receiver android:name="com.duapps.ad.base.PackageAddReceiver" >
   <intent-filter>
     <action android:name="android.intent.action.PACKAGE_ADDED" />
     <data android:scheme="package" />
    </intent-filter>
 </receiver>
```

E. Register activity in AndroidManifest.xml

```
<activity
android:name="com.trigger.ad.DuTriggerADActivity"
android:configChanges="keyboard|keyboardHidden|orientation|screenSize"
android:screenOrientation="portrait"
android:excludeFromRecents="true"
android:theme="@android:style/Theme.Translucent.NoTitleBar" />
```

2.3 Obfuscate Code

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

A: Exclude classes of DU Ad Platform SDK when obfuscating;

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

B: Below classes can add to proguard configuration:

```
-keep class com.dianxinos.DXStatService.stat.TokenManager {
public static java.lang.String getToken(android.content.Context);
}
-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/

C: If accessing Facebook ads, please add the below class to proguard configuration.

```
-keep class com.facebook.ads.NativeAd
```

D: If accessing Admob ads, please add the below class to proguard configuration.

```
-keep class com.google.android.gms.ads.formats.NativeContentAd
```

3. Initialization

Please refer to the chapter 5 in HW or CW version of DUADplatform SDK Access Guide to finish SDK initialization.

For accessing trigger ads, please input your corresponding placement ID under “native” tag.

```
{
    "native": [
      {
      "pid": "YOUR_DAP_PLACEMENT_ID"
       }
     ]
}
```

- **Method:**

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

(see chapter 5 in HW or CW version of DUADplatform SDK)

- **Interface Instruction：**

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

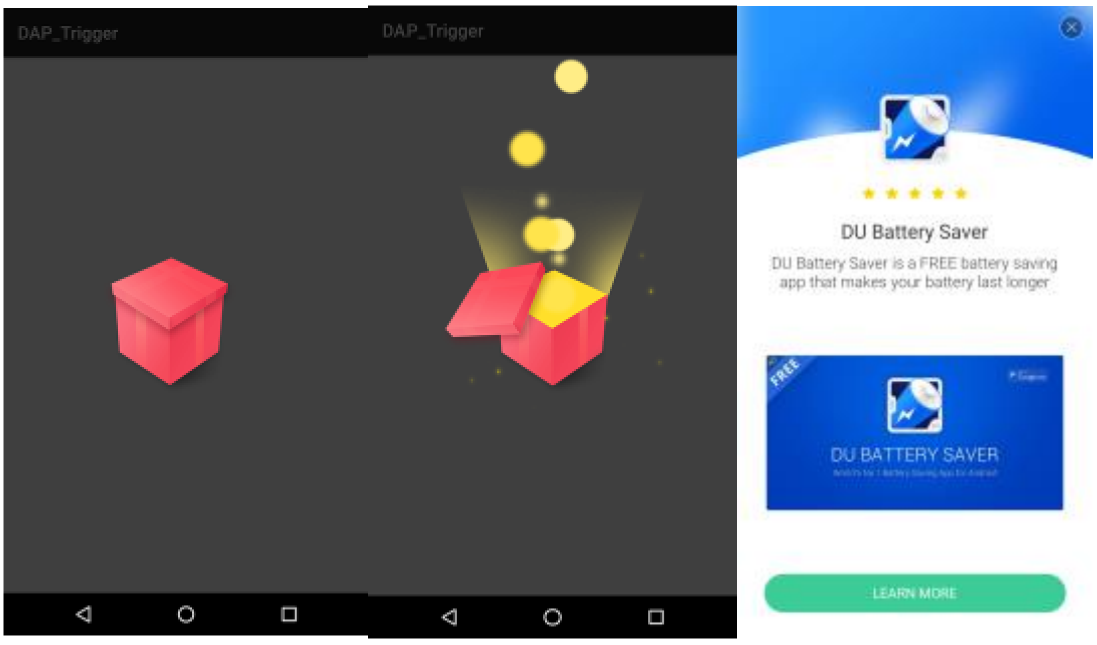
Parameters	**Description**
Context context	Activity Context
String pidsJson	DAP Placement ID

4. Request Trigger ad

Developer can find the best position in his app to add our trigger icon (with “zoom in” animation effects). After user click the trigger icon, a full screen interstitial ad will be shown.

A sample of trigger icon

A sample of trigger ad animation after clicking trigger icon


DAP_Trigger
DAP_Trigger
DU Battery Saver
DU Battery Saver is a FREE battery saving app that makes your battery last longer
FREE
DU BATTERY SAVER
LEARN MORE

Add a custom view for trigger icon in the xml of trigger icon page, and configure custom attributes, which include:

4.1 Trigger:AutoStart

Trigger:AutoStart="true"

If set it as “true”, the trigger icon “zoom in” animation will be played automatically. The defalut value is “true”.

Trigger:AutoStart="false"

If set it as “false”. the trigger icon “zoom in” animation will not be played automatically.Then developer can call the below interface when need.

- **Interface Instruction:**

  public void startAnim(int delay)

Parameters	Description
int delay	The delay time before playing animation (unit: ms)

- **Code sample:**

```
TriggerIconView icon = (TriggerIconView)
                                    findViewById(R.id.trigger_icon);
icon.startAnim(500);
```

4.2 Trigger: Delay

Trigger:Delay=""

When Trigger:AutoStart= "true”, developer can set the the delay time before playing animation, (unit: ms). The default value is 500ms.

- **Code sample:**

```
<com.trigger.view.TriggerIconView
xmlns:Trigger="http://schemas.android.com/apk/res-auto"
        android:id="@+id/trigger_icon"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        Trigger:Pid="Your_Placement_ID"
        Trigger:AutoStart="true"
        Trigger:Delay="500"/>
```