



Android SDK v 4.0
SDK Integration Guide
Publishers



Android SDK v4.0

SDK Integration Guide

Publishers



Introduction

This document outlines the Android SDK v4.0 that allows displaying mobile ads, mobile editorials and interstitials (**simple** and **HTML5** version) inside android applications.

The current version of the Android SDK is compatible with Android 2.3 and above.

Today users expect the same high-quality experience regardless of device and if they don't get it, they blame your brand. This is why we use **Akamai number 1 CDN world wide** to display your ads.

The Android SDK v4.0 supports now the **Google's Advertising Identifier**.

The new Android SDK v4.0 allows you to handle ads events.

1. Download Android SDK v4.0

Download the Ad4game Android SDK for publishers. Decompress the zip file and extract the files to your development computer. The Android SDK is provided as a single java JAR file (**AndroidSDKPublisher4.0.jar**), making it easy to include in your Android project.

2. Configure AndroidManifest.xml

The SDK requires the following permissions. Put these in before closing the manifest tag.

Internet Permission (Mandatory):

The Internet permission is required to connect to publishing servers.

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

Wifi State Permission (Recommended):

This permission enable the SDK to access information about whether you are connected to a Wi-Fi network and obtain the device's MAC address.

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

Connection Type Permission (Recommended):

This permission enable the SDK to detect if you are connected from a Wi-Fi network or 3G network.

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

Tasks Permission (Mandatory):

Allows the SDK to get information about the currently or recently running tasks to set the [refresh](#) value.

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

Google's Advertising Identifier (Mandatory):

Allows the SDK to get Google's Advertising Identifier.

Open your manifest file and add the following tag as a child of the [`<application>`](#) element:

```
<meta-data android:name="com.google.android.gms.version"  
        android:value="@integer/google_play_services_version" />
```

3. Install the Android SDK

Add **AndroidSDKPublisher4.0.jar** to your Android project's build path.

If you are using Eclipse, right click on your **project folder -> Properties -> Java Build Path -> Libraries -> Add JARs**. (As of ADT 17.0, you can simply place **AndroidSDKPublisher4.0.jar** in a folder called “libs” in your project directory and Eclipse will automatically include the jar as part of the build under the classpath container “Android Dependencies”).

4. Display mobile ads

To display mobile ads inside your android application, you need to set the value of “[refresh](#)” variable to refresh banners. You'll then need to instantiate the A4GPublisher class to allow you to call the functions from the AndroidSDKPublisher.

Generally, the code is placed inside your main activity's `onCreate(Bundle)` method.

You will need to pass the layout ID to the **findViewById** method, and then call **loadZoneId("ZoneID")** method.

```
public class MyClass extends Activity {  
    public static int refresh = 30000; //30 seconds  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_a4g);  
        A4GPublisher widget = (A4GPublisher)  
findViewById(R.id.yourLayoutID);  
        widget.loadZoneId("ZoneID");  
    }  
}
```

In your **layout.xml** file, you need to include the properties of the layout where you want to display the banners as follows

```
<com.a4gpublisher.A4GPublisher  
    android:id="@+id/yourLayoutID"  
    android:layout_width="350dp"  
    android:layout_height="250dp"  
    ...  
/>
```

Note: You can add many zones as you want, just instantiate the A4GPublisher object to create many widgets.

Ex:

```
A4GPublisher widget1 = (A4GPublisher) findViewById(R.id.yourFirstLayoutID);  
widget1.loadZoneId("ZoneID");
```

```
A4GPublisher widget2 = (A4GPublisher) findViewById(R.id.yourSecondLayoutID);  
widget2.loadZoneId("ZoneID");
```



4. 1 Handling Events

If you want to handle ads events, you need to call A4GLListener interface that includes **onAdLoaded**, **onAdFailed** and **onAdClicked** methods as follow :

```
A4GPublisher widget = (A4GPublisher)findViewById(R.id.a4glayoutid);  
widget.loadZoneId("ZoneID");  
widget.setA4GLListener(new A4GLListener() {  
    @Override  
    public void onAdLoaded() {  
        //Do your treatment  
    }  
    @Override  
    public void onAdFailed() {  
        //Do your treatment  
    }  
    @Override  
    public void onAdClicked() {  
        //Do your treatment  
    }  
});
```

5. Display Android Editorials

To display Android Editorials inside your android application, you need to instantiate the A4GEeditor class to allow you to call the functions from the AndroidSDKPublisher.

you need to use one of the three methods (you can use all of them) bellow:

- 1- **showA4GEedito** : If you want to display the editorials once the activity's started.
- 2- **showA4GEeditoWithStartTime** : If you want to display the editorials after certain time.
- 3- **showA4GEeditoWithStartAndEndTime** : If you want to display the editorials after certain time and close them automatically after certain time.

4- **showA4GEeditoOnClick** : If you want to display the editorials when the user click on a button.

Note: The three first methods mentioned above are valid only if your project target android **3.0** or above, **showA4GEeditoOnClick** is valid from android **2.3** or above.

To use one of the three methods mentioned above, you need to define a View and assign it to the A4GEeditorial object like shown bellow:

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
    A4GEeditorial a4gEdito = new A4GEeditorial(getApplicationContext());  
    a4gEdito.setZoneid("zoneID");  
    View v = findViewById(R.id.yourView); // a textView or any composant that you use inside  
                                         // your activity  
    a4gEdito.setVue(v);  
    a4gEdito.showA4GEedito(getApplicationContext());  
    a4gEdito.showA4GEeditoWithStartTime(getApplicationContext(), 5000);  
    a4gEdito.showA4GEeditoWithStartAndEndTime(getApplicationContext(), 3000,  
10000);  
}
```

To use **showA4GEeditoOnClick** function, you need to define a button and assign it to the A4GEeditorial object like shown below:

```
Button myButton = (Button)findViewById(R.id.yourButton);  
a4gEdito.setButton(myButton);  
a4gEdito.showA4GEeditoOnClick(getApplicationContext());
```

5. 1 Handling Events

If you want to handle editorials events, you need to call A4GEeditoListener interface that includes **onEditorialLoaded**, **onEditorialFailed**, **onEditorialClicked** and **onEditorialClosed** methods as follow:

```
A4GEditional ag4Edito = new A4GEditional(getApplicationContext());
ag4Edito.setZoneid("43516");
ag4Edito.setVue(txt);
ag4Edito.setA4GEditionListener(new A4GEditionListener() {
    @Override
    public void onEditorialLoaded() {
        //Do your treatment
    }
    @Override
    public void onEditorialFailed() {
        //Do your treatment
    }
    @Override
    public void onEditorialClicked() {
        //Do your treatment
    }
    @Override
    public void onEditorialClosed() {
        //Do your treatment
    }
});
ag4Edito.showA4GEdition(getApplicationContext());
ag4Edito.showA4GEditionWithStartTime(getApplicationContext(), 10000);
ag4Edito.showA4GEditionWithStartAndEndTime(getApplicationContext(), 10000,
20000);
```

6. Display Android Interstitials

To display Android Interstitials (**simple** version or **html5** version) inside your android application, you need to instantiate the A4GInterstitials class to allow you to call the functions from the AndroidSDKPublisher.

you need to use one of the three methods (you can use all of them) bellow:

- 1- **showA4GInterstitials** : If you want to display the interstitials once the activity's started.
- 2- **showA4GInterstitialsWithStartTime** : If you want to display the interstitials after certain time.
- 3- **showA4GInterstitialsWithStartAndEndTime** : If you want to display the interstitials after certain time and close it automatically after certain time.

To use one of the three methods mentioned above, you need to define a View and assign it to the A4GInterstitials object like shown bellow:

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
    A4GInterstitials a4gInter = new A4GInterstitials(getApplicationContext());  
    a4gInter.setZoneid("zoneID", getApplicationContext());  
  
    View v = findViewById(R.id.yourView); // a textView or any composant that you use inside  
                                         // your activity  
    a4gInter.setVue(v);  
    a4gInter.showA4GInterstitials(getApplicationContext());  
    a4gInter.showA4GInterstitialsWithStartTime(getApplicationContext(), 5000);  
    a4gInter.showA4GInterstitialsWithStartAndEndTime(getApplicationContext(),  
            3000, 10000);  
}
```

6. 1 Handling Events

If you want to handle interstitials events, you need to call A4GInterListener interface that includes **onInterstitialLoaded**, **onInterstitialFailed**, **onInterstitialClosed** and **onInterstitialClicked** methods as follow:



```
A4GInterstitials a4gInter = new A4GInterstitials(getApplicationContext());
a4gInter.setZoneid("43517", getApplicationContext());
a4gInter.setVue(txt);
a4gInter.setA4GInterListener(new A4GInterListener() {
    @Override
    public void onInterstitialLoaded() {
        //Do your treatment
    }
    @Override
    public void onInterstitialFailed() {
        //Do your treatment
    }
    @Override
    public void onInterstitialClosed() {
        //Do your treatment
    }
    @Override
    public void onInterstitialClicked() {
        //Do your treatment
    }
});
a4gInter.showA4GInterstitials(getApplicationContext());
a4gInter.showA4GInterstitialsWithStartTime(getApplicationContext(), 10000);
a4gInter.showA4GInterstitialsWithStartAndEndTime(getApplicationContext(),
10000, 20000);
```