



# **AdView**

## **Android SDK Specification**

**v1.01**

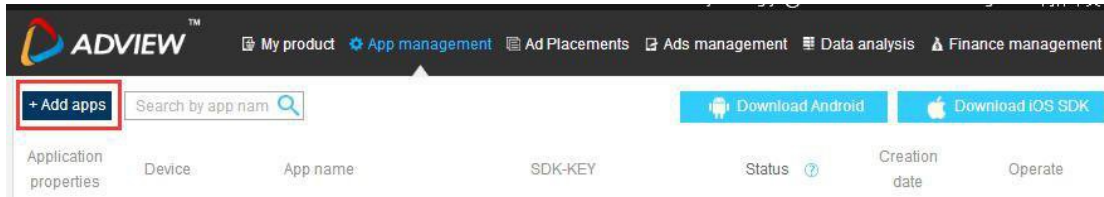
# Content

I .Register and get the SDK	
.....	
3	
II . Add the SDK	
.....	
5	
III. Add permissions and declare code	
.....	
5	
IV. Integrated SDK	
.....	
6	
1. Banner ads code to create (Single request mode)	
.....	
6	
2. Insert ads code to create	
.....	
7	
3. Open screen ads code to create	
8	
4. Native ads code to create	
.....	1
0	
5. Video ads code to create	
.....	1
1	
V.FAQ	
.....	1
3	
1. What should I do if the app wants to be confused (ProGuard)?	
.....	1
3	
2. Who do I contact if I have questions?	
.....	1
3	



## I . Register and get the SDK key

1. Sign up at AdView website <http://www.adview.cn> or <http://www.adview.com>
2. Add apps



3. Fill in the required information, please select the right ad format, otherwise you cannot get any ads.

4. Select AdViewBID and turn on the switch, enter 100% in capacity, Save and click Next

Advertising platform	Scoring	Setting	On/off	Capacity	Priority	Operate
AdViewBID	5.70	pending review	<input checked="" type="checkbox"/>	100%		

## 5. Wait for AdView's approval to get formal ads

The screenshot shows a progress bar at the top with three steps: 1. Add app information, 2. choose ad platform, and 3. Download SDK/Get links. Below the progress bar, a grey box displays the SDK-KEY. The main content area shows a congratulatory message, the SDK-KEY, a 'CopyID' button, and instructions to save the key and use an offline profile file. A link for downloading the offline configuration file is also present.

1 Add app information 2 choose ad platform 3 Download SDK/Get links

SDK-KEY

Congratulations, you have successfully added 123

AdView SDK-KEY of the app: **SDK20171520031256r9og9u7kf3n** [CopyID](#)

Please save the AdView SDK-KEY of your app, you need to copy it to the application code.

In order to guarantee your ad is not affected when the network is broken or AdView server is down and other extreme circumstances, AdView recommends you add an "offline profile file" while integrating AdView SDK. Please export your offline configuration file below, the configuration files are only available for banners, and add it to app source code according to the explanation in the SDK documentation.

[Offline configuration file download >>](#)

## 6. If you need video or native ads, go to ad placements model to create ad placement ID.

The screenshot shows the 'Create Ad Placement' form in the AdView dashboard. The dashboard header includes the AdView logo and navigation links: My product, App management, Ad Placements (active), Ads management, Data analysis, and Finance management. The form has fields for 'App Name' (a dropdown menu), 'Name' (a text input with an example), and 'Type' (radio buttons for Banner, Interstitial screen, Open screen, Native/Information, and video). A note at the bottom states that Banner AD space is temporarily allocated by the AdView system with a size of 320x50 and that one application temporarily supports only one banner AD space.

ADVIEW™

My product App management **Ad Placements** Ads management Data analysis Finance management

Create Ad Placement

App Name: \*

Name: \*  example: app\_name+position+style

Type: ☒ Banner ☐ Interstitial screen ☐ Open screen ☐ Native/Information ☐ video

Banner AD space is temporarily allocated by AdView system, with a size of 320x50, and one application temporarily supports only one banner AD space

## II. Add the SDK

1. Download **adview-v4.0-OVS-xxx.zip**
2. Add **adview-v4.0-OVS-xxx.jar** to your application project.

## III. Add permissions and declare code

```
<!-- AdView SDK mandatory or important permissions -->
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<!-- if need location can use this -->
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.REQUEST_INSTALL_PACKAGES" />
<!-- targetSdkVersion >= 26 must provide this -->
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS"/>

<!-- Must declare it for Adview SDK -->
<activity
    android:name="com.kuaiyou.video.AdViewVideoActivity"
    android:configChanges="keyboard|keyboardHidden|orientation|screenLayout|uiMode|screenSize|smallestScreenSize"
    android:hardwareAccelerated="true" >
</activity>

<service android:name="com.kuaiyou.utils.DownloadService" />
<activity android:name="com.kuaiyou.utils.AdViewLandingPage" />
<activity android:name="com.kuaiyou.utils.AdActivity" />
```

## IV. Integrated SDK

### 1. banner code generation (Single request mode)

```
// Initialize banner (Incoming parameters followed by context, sdkKey, whether to
refresh,and gdpr string )
AdViewBannerManager adViewBIDView = new
AdViewBannerManager (context,sdkKey, true, gdpr);

// Set whether to turn on the ads off
function
adViewBIDView.setShowCloseBtn(false);

// Set the ad switching time (seconds), if you do not set the default does not
automatically switch ads, if -1 means not changed automatically
adViewBIDView.setRefreshTime (15);
// Set the animation when the ads automatically switch
adViewBIDView.setOpenAnim(true);
// Set the monitor callback
adViewBIDView.setOnAdViewListener (this);
```

#### 1.1 banner ad interface description

```
public interface AdViewBannerListener{
    /**
     * This function is called when the
     ad clicks. */
    void onAdClicked();
    /**
     * This function is called when the ad is
     displayed. */
    void onAdDisplayed();
    /**
     * This function is called when the ad is
     received. */
    void onAdReceived();
    /**
     * This function is called when an ad
     request fails. */
    void onAdFailedReceived(String error);
    /**This function is called when the ad is closed. */
    void onAdClosed(); }
}
```

## 2. Interstitial code generation

```

// Instl initialization (incoming parameters followed by Context, sdkKey,
whether it can be closed, and gdpr string)
AdViewInstlManager adInstlBIDView= new AdViewInstlManager (context, sdkKey,
True, gdpr );

// Set the monitor callback
adInstlBIDView. setOnAdViewListener(this);

// Initialize the layout of the inserted screen when the ad is successfully received
@Override
public void onAdReceived() {
    // At this point you can call the show insert screen method
    adInstlBIDView.showInstl(context)
    // If you use a custom splash screen, such as the back-screen ads can use the
following
methods to get the ad view, custom display
    //adInstlBIDView.getDialogView();
}

```

### 2.1 Interstitial interface description

```

public interface AdViewInstlListener{
    /**
     * This function is called when the
ad clicks. */
    void onAdClicked();
    /**
     * This function is called when the ad is
displayed. */
    void onAdDisplayed();
    /**
     * This function is called when the ad is
received. */
    void onAdReceived();
    /**
     * This function is called when an ad
request fails. */
    void onAdFailedReceived(String error);
    /**This function is called when the ad is closed. */
    void onAdClosed();
}

```



### 3. Opening screen (Spread) code generation

```
// Initialize Instl (incoming parameters followed by Context, sdkKey, the
// need to display the outer layout of the open-screen ad, and gdpr string )
AdViewSpreadManager adViewBIDSpread = new AdViewSpreadManager
(context,sdkKey,parentView,gdpr);

// Set the top countdown notification method, the default does not notify
adViewBIDSpread.setSpreadNotifyType(
AdViewSpreadManager.NOTIFY_COUNTER_NULL);

// Set open screen advertising logo
adViewBIDSpread.setLogo(logoDrawable);

// Set the background color of the open screen ad
adViewBIDSpread. setBackgroundColor (backgroundColor);

// Set the open screen ads monitor callback
adViewBIDSpread. setOnAdViewListener (this);
```

The top notification settings of opening screen

Enum name	description	Constant value
NOTIFY_COUNTER_NULL	Do not show any countdown tips	0
NOTIFY_COUNTER_NUM	After setting the top shows countdown	1
NOTIFY_COUNTER_TEXT	After setting the top of the display as a skip button (after the specified show time will appear)	2
NOTIFY_COUNTER_CUSTOM	After setting will call onAdNotifyCustomCallback (), in which you can customize the notification style	3

### 3.1 Open screen advertising interface instructions

```
public interface AdViewSpreadListener{  
    /**  
     * This function is called when  
     the ad clicks. */  
    void onAdClicked();  
    /**  
     * This function is called when the ad  
     is displayed. */  
    void onAdDisplayed();  
    /**  
     * This function is called when the ad is  
     received. */  
    void onAdReceived();  
    /**  
     * This function is called when an ad  
     request fails. */  
    void onAdFailedReceived(String error);  
    /**  
     * This function is called when the ad  
     is closed. */  
    void onAdClosed();  
    /**  
     * This function is called when the show time is  
     about to close. */  
    void onAdSpreadPrepareClosed();  
    /**  
     * User cancels  
     display */  
    void onAdClosedByUser();  
    /**  
     * Custom  
     callbacks */  
    void onAdNotifyCustomCallback(int ruleTime,int delayTime);  
}
```

## 4. Native ads generation

```
// Initialization of native ads (incoming parameters followed by Context, sdkKey,  
advertising id, monitoring interface, and gdpr string (can be null))  
AdViewNativeManager adViewNative = new AdViewNativeManager (this, appld,  
posId, nativeAdCallBcak, gdpr);  
// Request advertisement (optional parameter: number of advertisement)  
adViewNative.requestAd();  
nativeBean = (HashMap) nativeAdList.get(0);  
// Report display  
adViewNative.reportImpression((String) nativeBean.get("adId"));  
// Click to report the incoming parameters for the current ad id -> corresponding map  
adIdfield  
adViewNative.reportClick((String) nativeBean.get("adId"));  
    (int)event.getX(), (int)event.getY());
```

### 4.1 Native ad interface instructions (Native ads only)

```
public interface AdViewNativeListener {  
    /**  
     * This function is called when the ad request is  
     * successful. */  
    public void onNativeAdReceived(List<HashMap> nativeAdList);  
    /**  
     * This function is called when an ad  
     * request fails. */  
    public void onNativeAdReceiveFailed(String errorCode);  
    /**  
     * Download progress  
     * status code. */  
    public void onDownloadStatusChange(int arg0);  
    /**  
     * This function is called  
     * when an ad closed. */  
    public void onNativeAdClosed(View var1);  
}
```

#### 4.2 Native ad field description

Field	Type	Description
<b>adIcon</b>	String	Icon picture link
<b>adImage</b>	String	Big picture data picture link
<b>title</b>	String	Ad title
<b>description</b>	String	Ad description content
<b>sec_description</b>	String	Supplementary description text
adFlagLogo	String	Logo
imageWidth	int	Big picture width
imageHeight	int	Big picture high
iconWidth	int	Icon wide
iconHeight	int	Icon high

\* The first four bold fields are the key ones

## 5. Video ads generation

```
/* Common non-patch advertising methods used:
// Initialize video ads (incoming parameters followed by Context, sdkKey,
advertising id, monitoring interface, is paste video, and gdpr string )
    AdViewVideoManager videoManager = new AdViewVideoManager(this, appld,
posld, adViewVideoInterface, false,gdpr );
// This method is called when the ad is loaded to show video ads
    videoManager.playVideo(context);
```

### 5.1 Video ad interface description (video ads only)

```
public interface AdViewVideoListener {
    /**
     * This function is called when the ad request is
     successful. */
    public void onReceivedVideo(String vast);
    /**
     * This function is called when an ad
     request fails. */
    public void onFailedReceivedVideo(String errorCode);
    /**
     * Called when the video ad is ready, after which video shows
     can be made */
    public void onVideoReady();
    /**
     * Called when video ads start
     playing */
    public void onVideoStartPlayed();
    /**
     * Called when video ads
     have ended */
    public void onVideoFinished();
    /**
     * Called when the video ad
     is closed */
    public void onVideoClosed();
    /**
     * Called when the video ad is playing
     incorrectly */
    public void onPlayedError(String error); }
```

## V.FAQ

### 1. What should I do if I want ProGuard app

AdView call ads dynamically, no need to make it ProGuard, the basic advertising company code has been independently ProGuard, if it's needed, you can add below at the beginning of proguard.cfg, more details please refer to the sample code (the code below can be copied in the sample):

```
-dontwarn
-keepclassmembers class * {public *;}
-keep public class    com.kuaiyou.**.* {*; }
-optimizationpasses 5
-dontusemixedcaseclassnames
-dontskipnonpubliclibraryclasses
-dontpreverify
-verbose
```

At present Adview SDK support proguard 4.6 above version, developers can download at proguard official website

<http://sourceforge.net/projects/proguard/files/proguard/>

If you want to upgrade, then replace the new version with the

"android-sdk-windows \ tools \ proguard"

### 2. Who should I contact with if any questions?

Log in AdView website, service email, customer service hotline, corporate QQ customer are available at the bottom of the page.

