Table of Contents

Millennial Media Android SDK - Version 4.5.0

Thank you for joining mmDev and becoming a Millennial Media publisher! Included here are all of the resources for integrating Millennial Media advertising into your Android application.

Contents of the Zip File

- Readme.pdf (this file)
- License.txt
- Changelog.txt
- MMAdView.jar
    - ♦ The library containing Millennial Media classes and resources to include in your application
- SampleApp/
    - ♦ A sample application that demonstrates basic MMAdView integration, different ad types, and advanced features
- Javadocs/
    - ♦ HTML files containing Javadocs documentation for public classes and methods

Quick Start

For a short overview, or if you have used the Millennial Media Android SDK in the past, you can reference the quick start instructions below. If you would like more detailed information please see the detailed documentation in this document and the Javadocs folder.

1. Place the MMAdView.jar into the libs directory of your Android project.
2. Add MMAdView.jar to your project's build path.
3. In your `AndroidManifest.xml` file add the following:
    - ♦ Permissions for `android.permission.INTERNET`, `android.permission.WRITE_EXTERNAL_STORAGE`, `android.permission.READ_PHONE_STATE`, and `android.permission.ACCESS_NETWORK_STATE`.
    - ♦ Activities for `com.millennialmedia.android.MMAdViewOverlayActivity` and `com.millennialmedia.android.VideoPlayer` with attribute: `android:configChanges="keyboardHidden|orientation|keyboard"`.
4. In your code, create an instance of MMAdView with your APID. If you want to show banner ads add the MMAdView to your view hierarchy in XML or programatically. Be sure that your instance of MMAdView has a unique valid view id.
5. Optional implementations
    - ♦ (Optional) Implement an MMAdListener class to receive events about your MMAdView.
    - ♦ (Optional) Specify additional metadata such as age and gender.
    - ♦ (Optional) Specify location metadata to make your application available to more monetization opportunities.

What's New In This Version

- Added a MMAdCachingCompleted(MMAdView adview, boolean success) as a new required listener method
- Fetch, check, and display functions now allow better control over interactive video interstitials and other cached ads

- Interactive video cache sharing among other Millennial Media publishers on the same device
- Application retargeting from within creatives using third party scheme urls (For more information see Millennial Media's Branded App SDK)
- Improved location-based tracking using the new updateUserLocation method
- Added precaching capabilities
- Added click to interactive video support
- Added new overhauled creative javascript bridge for showing more immersive advertising
- Better conversion tracking support for multiple goal ids
- Misc. improvements and bug fixes

Requirements

New to 4.5+: Declarations for the supporting Millennial Media activities has changed please see the documentation for details. `MMAdCachingCompleted(MMAdView adview, boolean success)` is a required listener method for objects implementing the MMAdListener interface.

New to 4.2+: Versions 4.2 and above require a new listener method `MMAdRequestIsCaching` and require the `android.permission.WRITE_EXTERNAL_STORAGE` permission. Additionally, all MMAdView objects must be assigned a unique, valid view id, either in the XML layout or programmatically using the `setId()` method.

Documentation

Introduction

The Millennial Media Android SDK is a Java-based library that allows publishers to incorporate advertisements into their applications. Developers use this SDK to simplify monetization, abstract ad display, and provide richer ad experiences. This document covers integrating the SDK into your application, placing ads, and configuring them. It is written for developers with the assumption they are familiar with Android development, Eclipse, Java, the Android Development Tools (ADT), and Android Virtual Devices.

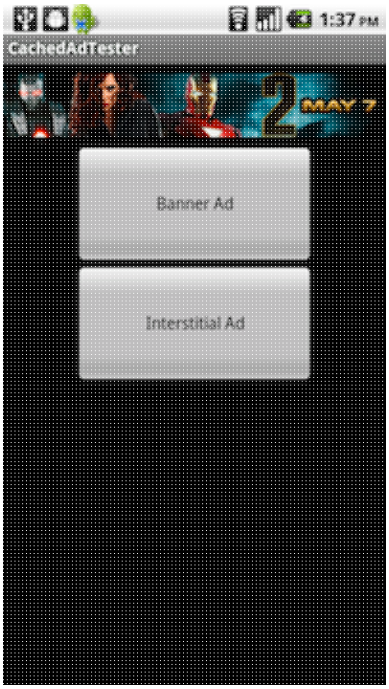
Ad Placements

There are 2 general types of ad views served: banner ads and interstitial ads.

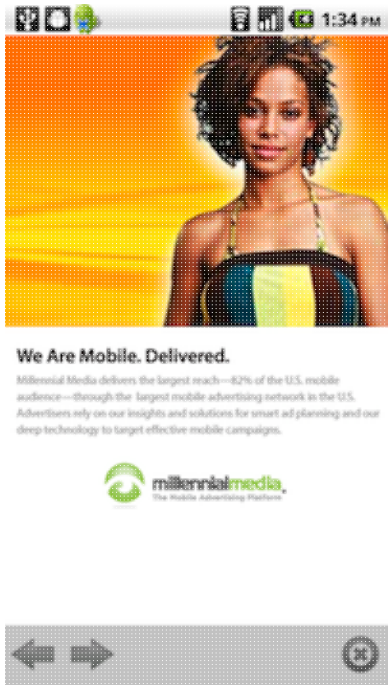
A banner ad is a view that spans the screen width and is visible alongside content. The two ad types are `MMBannerAdTop` (Top of Page) and `MMBannerAdBottom` (Below the Fold). It is recommended that you make sure your MMAdViews are always the width of the screen and at least 53 pixels high. Whenever an Ad Placement is on-screen, you should make sure it is fully visible. By default, ad placements are refreshed every 60 seconds while the screen is active unless changed in the initialization. The SDK will not refresh ads if the screen is in the background or if the phone is locked.

An interstitial ad, also known as a full-screen ad, covers a large part of or all of the screen for a short time period. The two ad types are `MMFullScreenAdLaunch` (Launch Prestitial) and `MMFullScreenAdTransition` (Transition Interstitial). The MMAdView object handles all the networking and display of the ad. There is no need to add the MMAdView to your view hierarchy for displaying full page ads but the MMAdView object must be initialized.

An MMAdView can be placed anywhere in an application or on a page, but selecting the correct ad placement type is critical for receiving optimal ads.



Banner Ad



Full Page Overlay or Interstitial Ad

Options

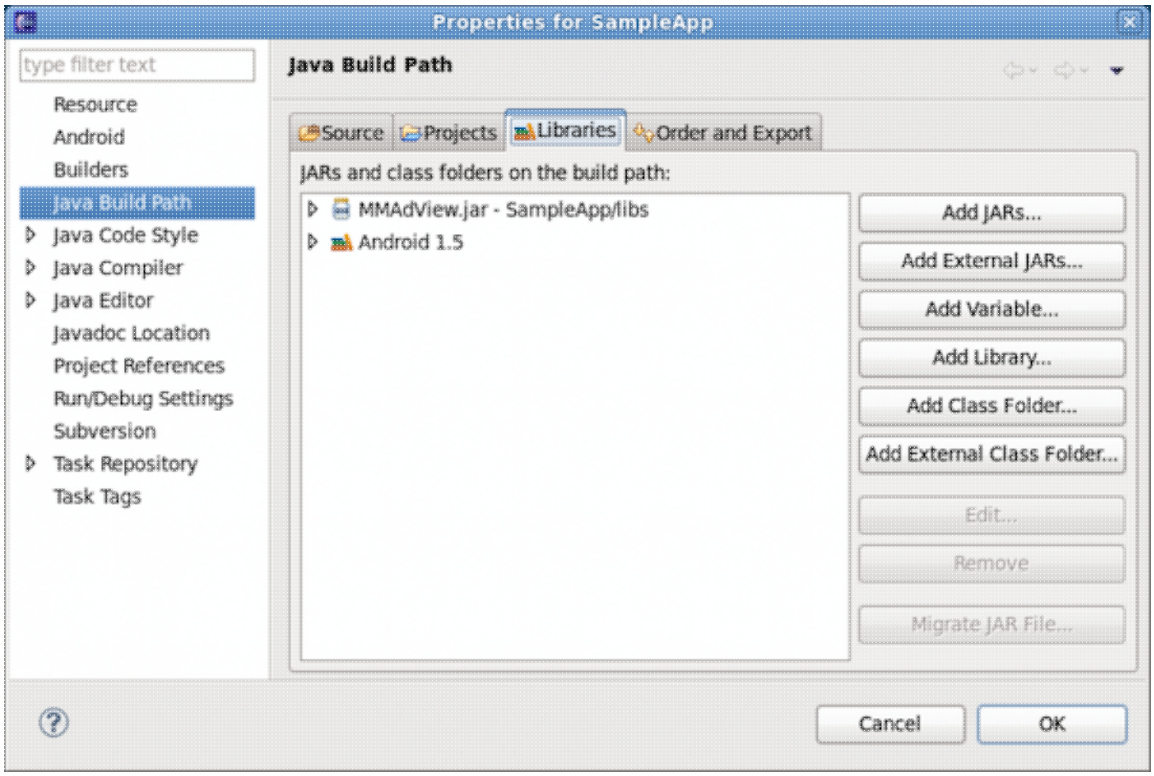
Ad Placements use several parameters to retrieve relevant ads from Millennial Media's servers. The APID and ad type are the primary parameters that help our system select an appropriate ad to deliver. It is also possible to specify demographic metadata such as age and gender in order to receive more relevant ads. If your application is already making use of location functionality, the device's location can also be included as a parameter. It is also possible to include arbitrary parameters, both globally across your app and for specific Ad Placements.

Integrating the Library

Before you get started, make sure you have your Millennial Media Ad Placement ID (APID). You can create a new one or find an existing one in the developer portal under "Manage Applications."

Add the Millennial Media Library

- Place the MMAdView.jar file in the libs directory of your project.
- If you are using Eclipse, add MMAdView.jar to your application's build path.
- Highlight your project in the package explorer.
- Press ALT+ENTER to bring up the properties dialog.
- Click "Java Build Path".
- Select the Libraries Tab.
- Click "Add External JARs…".
- Browse to the location of the MMAdView JAR file.
- Highlight MMAdView.jar and select the "Open" button.
- Select "OK"



Set the Permissions and Configurations in the Manifest File

The SDK relies on having an active Internet connection to fetch and show ads. Before the SDK can make these requests, necessary permissions must first be added to your `AndroidManifest.xml` file. If you are in Eclipse follow the instructions below; otherwise just open your `AndroidManifest.xml` file in a text editor.

- Highlight your `AndroidManifest.xml` file in the project explorer.
- Double click on the file or press F3 to open it.
- Click on the `AndroidManifest.xml` tab to reveal the raw XML source.

```
        <activity android:name=".ImageViewActivity"
            android:label="ImageViewActivity"
            android:theme="@android:style/Theme.Translucent.NoTitleBar">
            <intent-filter>
                <action android:name="android.intent.action.VIEW" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity android:name="com.millennialmedia.android.MMAdViewOverlayActivity"
                  android:theme="@android:style/Theme.Translucent.NoTitleBar">
        </activity>
        <activity android:name="com.millennialmedia.android.VideoPlayer"
                  android:theme="@android:style/Theme.NoTitleBar.Fullscreen"
                  android:configChanges="keyboardHidden|orientation|keyboard" >
        </activity>
    </application>
    <uses-sdk android:minSdkVersion="3" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
</manifest>
```

Inside the `<manifest></manifest>` element add the following code:

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

The SDK implements two activities for displaying different types of overlay ads. Overlays allow users to focus and engage with advertisements while staying inside the application. Inside the `<application></application>` tags add the following activity code:

```
<activity android:name="com.millennialmedia.android.MMAdViewOverlayActivity"
      android:configChanges="keyboardHidden|orientation|keyboard" >
</activity>
<activity android:name="com.millennialmedia.android.VideoPlayer"
          android:configChanges="keyboardHidden|orientation|keyboard" >
</activity>
```

Create an MMAdView Object

There are two ways to add an MMAdView to your project:

1. In a combination of XML and Java

2. Entirely in Java

XML Layout

In your layout file, declare the Millennial Media namespace:

```
xmlns:app="http://millennialmedia.com/android/schema"
```

Then add the MMAdView and associated meta data you wish to include.

```
<com.millennialmedia.android.MMAdView
      android:id="@+id/mmadview"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      app:apid="28911"
      app:adType="MMBannerAdTop"
      app:refreshInterval="30"
      app:accelerate="true"
      app:ignoreDensityScaling="false"
      app:age="46"
      app:gender="female"
      app:zip="90210"
      app:income="85000"
      app:keywords="moms,shopping,groceries"
      app:ethnicity="hispanic"
      app:orientation="straight"
      app:marital="married"
      app:children="2"
      app:education="college"
      app:politics="libertarian"
      app:height="60"
      app:width="480" />
```

Java Layout

```
public void onCreate(Bundle savedInstanceState){
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    Hashtable<String, String> map = new Hashtable<String, String>();
    map.put("age", "45");
    map.put("zip", "90210");
    map.put("income", "50000");
    //include additional metadata or settings in the hashmap

    MMAdView adView = new MMAdView(this, MYAPID, MMAdView.BANNER_AD_BOTTOM, 30, map);
    adView.setId(MMAdViewSDK.DEFAULT_VIEWID);
    FrameLayout adFrameLayout = (FrameLayout)findViewById(R.id.adFrameLayout);
    LayoutParams lp = new LayoutParams(LayoutParams.FILL_PARENT,LayoutParams.WRAP_CONTENT)
    adFrameLayout.addView(adView, lp);
}
```

Note: You must add the view to your layout for banner ads. Failure to do this will prevent you from showing any ads. Also, the MMAdView refreshes ad and performs other actions on a timer (Handler) that requires it to be within a message loop. Make sure to create MMAdView objects and perform all method calls to MMAdView objects on your UI thread.

Additional Features

Listening for Ad Events

The MMAdView provides an interface that you can use to implement a class that listens for MMAdView events. The MMAdListener Interface gives you a way to know when an ad has successfully loaded or has failed to load. It also notifies you of when an ad has been clicked on and when overlays are shown. You can set the listener by calling the `setListener()` method of your MMAdView object. To implement an MMAdListener: Create a new class that implements MMAdListener. Define the required methods in your new listener class. Create an instance of your new MMAdListener class and set the listener of the MMAdView.

```
public class MyAdListener implements MMAdListener {
    public MyAdListener() {}

    public void MMAdFailed(MMAdView adview)
    {
        Log.i("My App", "MMAdView failed to load ad." );
    }
    ... other required methods here
}
...
MMAdListener listener = new MyAdListener();
adView.setListener(listener);
```

The MMAdListener interface requires the following methods:

```
public void MMAdFailed(MMAdView adview)

public void MMAdReturned(MMAdView adview)

public void MMAdClickedToNewBrowser(MMAdView adview)

public void MMAdClickedToOverlay(MMAdView adview)

public void MMAdOverlayLaunched(MMAdView adview)

public void MMAdRequestIsCaching(MMAdView adview)

public void MMAdCachingCompleted(MMAdView adview, boolean success)
```

If you are not interesting in listening for all of the events listed above, you may opt to have your listener extend com.millennialmedia.android.BasicMMAdListener. The BasicMMAdListener implements all of the required and MMAdListener methods and allows you to override only the methods you are interested in.

Note: Any call to your listener methods is not guaranteed to be running on the main thread. If you need to make changes to any view or activity in these methods, use the `runOnUiThread()` method.

Tip: You may wish to implement your activity as the listener. Otherwise if needed, you should use a WeakReference to your activity from your listener object.

Location-based Ads

If your application implements location-based services and you wish to pass the user's location to us, you can implement this by calling the `updateUserLocation()` method inside of your `onLocationChanged()` method. Setting the location using the `updateUserLocation()` method will

override any longitude and latitude set using the older `setLongitude()` and `setLatitude()` methods.

```
public void onLocationChanged(Location userLocation) {
    adview.updateUserLocation(userLocation);
}
```

Interstitial Ads

In addition to banner ads, you may choose to show interstitial ads, which are full screen ads that are typically shown as users transition in between different states of your app. To create an interstitial ad view follow the same steps as creating a banner ad and specify "MMFullScreenAdTransition" as the type. Interstitial ads do not need to be added to a layout. Instead use the below constructor and `callForAd()` method.

```
static MMAdView interAdView = null;
if(interAdView == null)
    interAdView = new MMAdView(activity, MYAPID, MMAdView.FULLSCREEN_AD_TRANSITION, true, null);
interAdView.callForAd();
```

Cached Ads

As of SDK version 4.1+, one type of interstitial ad that could be returned by the server is a cached ad. Cached ads are meant to provide a better user experience by completely downloading all of the required assets in the background before it is presented to the user. Currently, the only type of cached ads that will be served are interactive video ads with rich content overlays, and static interstitials when using the `fetch()` function. The interactive video feature can be managed in the developer portal.

The implications to you as a developer is that if the server is returning cached ads you may see an MMAdRequestIsCaching callback after an ad request because a cached ad was returned and is now downloading. After the ad has completely cached you will get an MMAdReturned callback. On your next ad request after this point you are now ready to display the cached ad.

Other information that you should be aware of is where the cached ads are stored. If an external SD card is present the SDK will opt to store the cached ad assets into a shared folder named millennialmedia off of the root. The benefit of caching on the SD card is that any assets cached will be shared among other apps that have also integrated the Millennial Media SDK. If no SD card is present then the cache will be located in the private cache directory of your app on the device's internal storage. There is also a small database managing cached ads that will always be stored in the cache directory of your apps private storage area.

The Millennial Media SDK will automatically manage the storage and purging of cached ads. All ads have an expiration date and get automatically removed after a certain point.

Fetching Cached Ads

Typically cached ads will be returned with one call to callForAd and displayed on a subsequent call when the ad has finished caching. This method leaves abiguity to the developer as to whether or not and ad will be cached or displayed. As of SDK version 4.5+, in efforts to improve user experience, you can now have greater control over your cached ads with three additional methods: `fetch()`, `check()`, and `display()`.

Using the `fetch()` method you can initiate a cached ad request. If no other caching is happening, then you should receive an MMAdRequestIsCaching call on your listener to let you know that caching has commenced. When caching has completed, you should receive an MMAdCachingCompleted call. At any point you may call `check()` on your MMAdView to check to see if a fetched ad is ready to display. After an ad has been cached its simply a matter of calling `display()` to show the cached ad.

Note: These methods apply to both interactive videos and static interstitials. Your MMAdView must be of type `FULLSCREEN_AD_LAUNCH` or `FULLSCREEN_AD_TRANSITION`.

SDK Reference

Using the MMAdView SDK is just as easy as creating any other Android view programmatically. Below is a quick run-down of the constructor and the type of data it expects. If you want more detail on each class and method please see the javadocs reference here.

MMAdView Class

```
MMAdView(Activity activty, String apid, String adType, int
refreshInterval, Hashtable<String, String> metaMap)
```

This constructor creates an MMAdView with the following parameters. Required parameters:

- activity: The Activity where the MMAdView will be placed.
- apid: Your ad placement id. This is the number assigned to you by Millennial Media that uniquely identifies your application, e.g. 15062.
- adType: This tells us where you are placing the ad and helps decide which ad to show.
    - `MMAdView.BANNER_AD_TOP` or "MMBannerAdTop"
    - `MMAdView.BANNER_AD_BOTTOM` or "MMBannerAdBottom"
    - `MMAdView.BANNER_AD_RECTANGLE` or "MMBannerAdRectangle"
    - `MMAdView.FULLSCREEN_AD_LAUNCH` or "MMFullScreenAdLaunch"
    - `MMAdView.FULLSCREEN_AD_TRANSITION` or "MMFullScreenAdTransition"
- refreshInterval: The ad refresh interval specifying how frequently a new ad is shown. This value can only be set once when the MMAdView is created. Minimum time interval is 15 seconds To turn off ad refresh set to 0. (Only load one ad and never refresh) To turn take control of ad calls, set to `MMAdView.REFRESH_INTERVAL_OFF` or -1 and use the `callForAd()` method as outlined below

Optional parameters:

- metaMap: A hash table of metadata values that may help Millennial provide more relevant ads to your users. All hash keys are optional and values can be any string you pass.

The available keys are:

- age: in years (e.g. 18), or "unknown"
- gender: "male", "female", or "unknown"
- income: approximate number in US dollars (e.g. 65000).
- zip: 5 digit US zip code, or "unknown"
- marital: Marital status, options are: "single", "divorced", "engaged", "relationship", "swinger".
- ethnicity: Options are: "hispanic", "africanamerican", "asian", "indian", "middleeastern", "nativeamerican", "pacificislander", "white", "other"
- orientation: Sexual orientation, options are: "straight", "gay", "bisexual", or "notsure".
- children: Boolean ("true" or "false"), integer (e.g. 2), or "unknown".
- politics: Political views or party affiliation, options are: "republican", "democrat", "conservative", "moderate", "liberal", "independent", "other", or "unknown".
- education: Highest level of education, options are: "highschool", "incollege", "somecollege", "associate", "bachelors", "masters", "phd", "professional", or "unknown".

- keywords: Any additional relevant description of the user or their preferences separated by commas. e.g. “soccer,scores,basketball”.
- width: The width of the ad if you have a limit on the size of ads you can display, e.g. 320. This should only be used if you have a specific limitation on the size of ads you can display. If not used, the ad returned will be optimized based on device specifications.
- height: The height of the ad if you have a limit on the size of ads you can display, e.g. 53. This should only be used if you have a specific limitation on the size of ads you can display. If not used, the ad returned will be optimized based on device specifications.

Methods

- `callForAd()` - Requests an ad when refreshing is disabled.
- `fetch()` - Requests a cached ad.
- `check()` - Check on the status of a fetched cached ad.
- `display()` - Display a fetched cached ad.

Other Options

`accelerate` - When set to false will allow you to maintain control of the accelerometer and no accelerometer-based ads will be served. This value can be set with a number of the MMAdView constructors or by specifying it in the XML layout. The default value for this option is true.

`ignoreDensityScaling` - This allows you to disable or enable any automatic scaling of the ad view content on high density devices. For example, on a high-density screen a 320x53 banner ad may appear to be stretched. Set this value to true to avoid the stretching and to show the banner ad in a smaller portion of the screen. You can set this value in the XML layout of the MMAdView or through the setIgnoresDensityScaling method. The default value is false.

Conversion Tracking

Conversion tracking is an optional feature of the SDK that can track app download conversions from advertising. To use this feature, ensure your account manager has provided a “Goal ID” and implement the static method below. Learn more here.

```
public static void startConversionTrackerWithGoalId(final Context context, final String goalId)
```

For example,

```
MMAdView.startConversionTrackerWithGoalId(this, "12345");
```

Classes and Methods

For more detailed documentation about each class and method, including deprecated methods, is located in the Javadocs folder.

Known Issues

SDK Versions prior to 4.5.0 should not pass null for the Hashtable parameter in an MMAdView constructor.

Frequently Asked Questions

How do I get started with putting Millennial Media ads inside my app?

Signup for a publisher account on the Developer page. From the mmDev portal you will be given an id number called an APID and can download Android advertising SDK.

What is an APID?

An APID is a placement id number, given to you by Millennial Media. This unique identifier that tells us which application is calling for ads. Implementing the APID is required and helps us keep track of your revenue and advertising statistics that you can view in the mmDev portal.

Where can I get an APID?

Visit out Developer page and create an account. After providing some details, your APID will appear on your "Manage Applications" page.

What versions of Android are supported?

The Millennial Media SDK supports applications running Android 1.5+ (API level 3 and above).

Why is my app crashing in the Android 2.3 emulator?

The MMAdView class relies upon the WebView's javascript to java bridge. Unfortunately, this bridge is broken in the 2.3 implementation of the Android emulator and will likely never be fixed. As an alternative you can do testing with other versions of the emulator or use a physical Android 2.3 device. http://code.google.com/p/android/issues/detail?id=12987

Why is my MMAdView not showing any ads?

Aside from the obvious problems of the device or simulator lacking active network connectivity, you may also be faced with the problem of not assigning your MMAdView a view id. As of version 4.2 it is required that each MMAdView has its own unique view id assigned either in the XML layout or by the setId() method. You may see in your log a message like, "MMAdView found without a view id. Ad requests on this MMAdView are disabled."

When you create an MMAdView you specify an ad type. Even though this ad type describes where the ad may be placed it does not mean the ad view will be added to your layout automatically. Make sure you are adding the view to your layout. Also make sure that an ad is being successfully requested and you are using the correct APID.

It does take some time to activate an APID in our system. If you are getting no ads or see "Zero length response" in your device logs, then wait 1-2 hours to make sure our system correctly identifies your new placement. If the problem persists, you can contact us here for more help.

Why can't I get interactive video interstitials working?

Be sure that interactive video interstitals are enabled in the developer portal. If you still can't get interactive videos working be sure that the SD card of your phone is not currently mounted on your computer through USB. Interactive video playback while the SD card is mounted is currently not supported. Additionally, interactive videos are disabled for 2.2 devices without external storage and for apps that do not declare the `WRITE_EXTERNAL_STORAGE` permission.

Why does my app crash with java.lang.RuntimeException: Can't create handler inside thread that has not called Looper.prepare()?

This exception occurs when your MMAdView was not created or used from within the app's UI thread. The MMAdView does ad refreshes and other actions on a timer (Handler) that requires it to be within a message loop. To fix this issue, put your MMAdView and all method calls to it on your UI thread.

What will an ad click do?

A click opens a new browser activity, creates a web overlay, opens the Android Market, opens the Phone application, plays a video, or other action. You can see examples of our creatives on our developer page.

What is the latest version of the SDK?

The latest version is Version 4.5.0 (released 10/11/2011).

Where can I download the SDK?

Go to the Developer page and create an account. After you login, select "Android" from the drop-down menu under "SDK Downloads".

What is Conversion Tracking and how do I get a Goal ID?

Conversion tracking allows publishers that promote their apps through Millennial Media's network to identify app downloads. You can get a goal ID and learn more by contacting us at http://developer.millennialmedia.com/contact.php.

I still have questions. How can I get help?

Use the contact page through the developer site located at http://developer.millennialmedia.com/contact.php.