

AdMob Adobe AIR Native Extension

Copyright © 2011-2015 Milkman Games, LLC. All rights reserved.

<http://www.milkmangames.com>

For support, contact support@milkmangames.com

Before you begin:

The AdMob Extension requires Adobe AIR 16.0 or higher. You can download the latest version of AIR at <http://www.adobe.com/devnet/air/air-sdk-download.html>. AIR 16.0+ is required for iOS applications.

To View full AS3 documentation, see 'docs/as3docs/index.html'. Review 'example/AdMobExample.as' for a sample application.

Note that AdMobExample.as is a Document Class. If you're a Flash Professional user and don't know how to use a document class, see the FAQ "How do I use the AdMobExample.as class in Flash CS6?" at the end of this guide.

1. Install the AIR 16.0 (or later) SDK in your IDE

The extension requires the AIR 16.0 SDK or later. You can download the latest AIR SDK from <http://www.adobe.com/devnet/air/air-sdk-download.html>. If you haven't already installed the AIR 16.0 (or later) SDK for your Flash or Flash Builder IDE, follow the instructions below:

Enabling the AIR 16.0 or later SDK in Flash Professional CS6+:

1. Unzip the AIR 16.0 or later SDK package to a location on your hard drive.
2. Launch Flash Professional.
3. Select Help > Manage AIR SDK...
4. Press the Plus (+) Button and navigate to the location of the unzipped AIR SDK
5. Press OK
6. Select File > Publish Settings
7. Select the latest AIR SDK for iOS from the 'Target' Dropdown menu

Enabling the AIR 16.0 or later SDK in Flash Builder 4.6+- Windows:

1. Unzip the AIR 16.0 or later SDK package to a location on your hard drive.
2. Close Flash Builder.
3. Locate the Flash Builder SDK directory. On the PC, usually c:\Program Files\Adobe\Adobe Flash Builder 4.6\sdk .
4. Make a copy of the current Flex SDK directory, and give it a descriptive name. For instance, copy the "4.6.0" SDK folder inside /sdks and name the copy "4.6.0_AIR16".
5. Copy and paste the contents of the new AIR SDK on top of the 4.6.0_AIR16 directory. Accept all changes.
6. Edit the flex-sdk-description.xml file inside the new directory, and change the value of the <name> tag to 'Flex 4.6.0 (AIR 16.0)'.
7. Open Flash Builder and choose Project > Properties > Flex Compiler > Configure Flex SDKs.
8. Press 'Add' and navigate to the new folder location.

Enabling the AIR 16.0 or later SDK in Flash Builder 4.6+- Mac:

1. Copy the contents AIR 16.0 or later SDK package to a location on your hard drive.
2. Close Flash Builder.
3. Locate the Flash Builder SDK directory. On the Mac, it is usually /Applications/Adobe Flash Builder 4.6/sdks/. On the PC, c:\Program Files\Adobe\Adobe Flash Builder 4.6\sdks .
4. Create a new folder inside the SDK folder, called AIR16SDK and copy the contents of the SDK package into it.
5. Open the Terminal, and merge the AIR 16.0 or later SDK files into your current SDK directory:

```
sudo cp -Rp /Applications/Adobe\ Flash\ Builder\ 4.6/sdks/AIR16SDK/  
/Applications/Adobe\ Flash\ Builder\ 4.6/sdks/4.6.0/
```

6. Edit the flex-sdk-description.xml file inside the new directory, and change the value of the <name> tag to 'Flex 4.6.0 (AIR 16.0)'.
7. Open Flash Builder and choose Project > Properties > Flex Compiler > Configure Flex SDKs.
8. Press 'Add' and navigate to the new folder location. Press 'Add' and navigate to the new folder location.

2. Include the AdMob and GoogleServices Extension Library

Add the AdMob ANE files to your project.

In Flash Professional:

1. Create a new mobile project
2. Choose File>Publish Settings...
3. Select the wrench icon next to 'Script' for 'ActionScript Settings'
4. Select the Library Path tab.
5. Click 'Browse for Native Extension (ANE) File' and select the com.milkmangames.extensions.AdMob.ane file.
6. Click 'Browse for Native Extension (ANE) File' and select the com.milkmangames.extensions.GoogleServices.ane file.
7. Press OK.
8. Select the wrench icon next to 'Target' for Player Settings
9. Select the 'Permissions' tab, and select 'INTERNET' and 'ACCESS_NETWORK_STATE'.
- 10. Check the 'Manually manage permissions and manifest additions for this app' box.**
11. Press OK

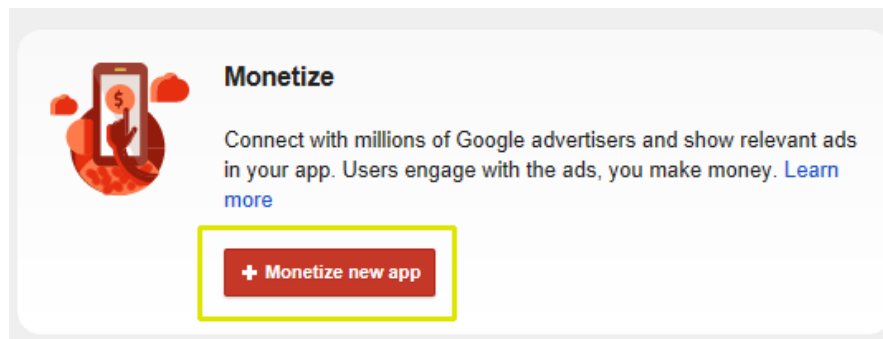
In Flash Builder 4.6 or higher:

1. Go to *Project Properties*
2. Select *Native Extensions* under *Actionscript Build Path*

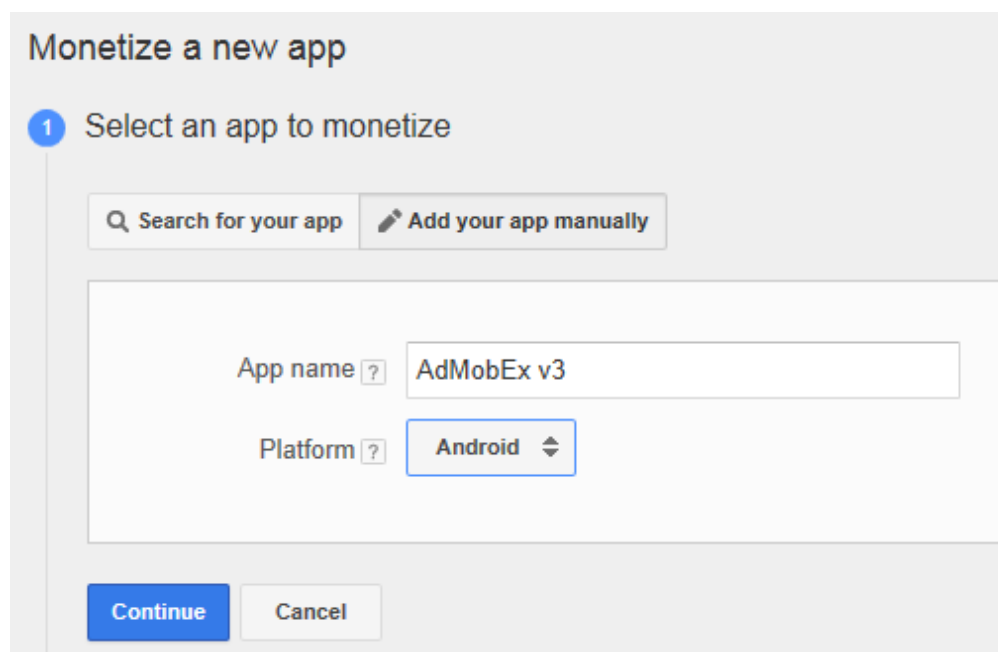
3. Choose *Add ANE...* and navigate to the `com.milkmangames.extensions.AdMob.ane` file
4. Choose *Add ANE...* and navigate to the `com.milkmangames.extensions.GoogleServices.ane` file
5. Select *Actionscript Build Packaging > Google Android (or iOS, if using the iOS Version)*
6. Select the *Native Extensions* tab, and click the 'Package' checkbox next to the extension

3. Register Your App with AdMob

1. Visit <http://www.admob.com> and sign up, or login to your existing account.
2. From the Home page, Select "Monetize New App".



3. Choose "Add your app manually", and give a name for your application. (Or, if it's already on the app store, use 'Search for your app' to find it.
4. Select the platform (Android or iOS) for your application. If you are creating one AIR application for both iOS *and* Android, follow the steps in this section through once for each platform. From AdMob's perspective, they are two different applications.



5. Press Continue. In the next step, select either the Banner or Interstitial type. Complete the remainder of the format with your desired style, name, and other settings.

2 Select ad format and name ad unit

Banner **Interstitial**

i Ad type, size, and placement are specified when you integrate the code using the AdMob SDK.

Automatic refresh **?** ☐ No refresh
☒ Refresh rate: seconds (30-120 seconds)

Text ad style **?**

Ad unit name **?** **X**
 Example: "Top Banner on Home"

Save **Cancel**

6. Press Save.
7. You will be presented with 'Ad Unit ID' for your new Ad. Copy and paste this ID; you will need this string for later when setting up your ActionScript code.

✓ Select ad format and name ad unit

Ad unit ID **ca-app-pub-9347846278811882~1136402148**

Ad unit name: **banner1**

8. If you'd like to create additional ad units or types, press 'Create Another Ad Unit'.
9. If you are targeting both iOS and Android, start over a step 1, creating a second version of your AdMob app for the other platform. Save a copy of your ad unit IDs for each platform.

4. Update Your Application Descriptor

You'll need to be using the AIR 16.0 or higher SDK, include the extension in your Application Descriptor XML, and update the Android Manifest Additions with some AdMob specific settings. For a working example, see 'example/app.xml'.

1. Set your AIR SDK to 16.0 (or higher) in the app descriptor file:

```
<application xmlns="http://ns.adobe.com/air/application/16.0">
```

2. Include a link to the extension sin the descriptor:

```
<extensions>
<extensionID>com.milkmangames.extensions.AdMob</extensionID>
<extensionID>com.milkmangames.extensions.GoogleServices</extensionID>
</extensions>
```

For the AdMob for Android Extension Only: Update your Android Manifest Additions- you'll need to have the INTERNET, and ACCESS_NETWORK_STATE permissions; you also need to add a special Activity for AdMob, and set the <uses-sdk> flag. ***The line shown in green below should appear on a single line in your XML file. There should be no spaces or carriage return inside keyboard|keyboardHidden|orientation|screenLayout|uiMode|screenSize|smallestScreenSize .***

```
<android>
<manifestAdditions><![CDATA[
    <manifest android:installLocation="auto">
        <uses-sdk android:minSdkVersion="9" android:targetSdkVersion="19" />

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

<!-- this meta-data tag is required for AdMob -->
<meta-data android:name="com.google.android.gms.version" android:value="4452000"/>

<!-- this activity is required for AdMob -->
<activity android:name="com.google.android.gms.ads.AdActivity"
    android:configChanges="keyboard|keyboardHidden|orientation|screenLayout|uiMode|
    screenSize|smallestScreenSize"
    android:theme="@android:style/Theme.Translucent.NoTitleBar"/>
</application>

</manifest>]]>
</manifestAdditions>
</android>
```

5. Add ActionScript Code

The AdMob extension can be up and running in a few simple calls. See 'example/AdMobExample.as' for a full example class, including error capturing, destroying and refreshing ads, setting ad visibility, and capturing events when ad status changes. Note that you need to change the ad unit ID string values to your own IDs!

Initialize the AdMob Extension

1. Import the API Classes:

```
import com.milkmangames.nativeextensions.*;
import com.milkmangames.nativeextensions.events.*;
```

2. Initialize the API with the Ad Unit ID for your main banner ad. You can retrieve your ad unit ID(s) from the AdMob web site control panel, as described in the section above, "Register Your App with AdMob."

You'll also want to check that the current platform is supported by the extension before initializing (for instance, AdMob won't work on Windows):

```
if (AdMob.isSupported)
```

```

{
    AdMob.init("YOUR_AD_UNIT_ID_HERE");
}
else
{
    trace("AdMob won't work on this platform!");
    return;
}

```

If you're using the AdMob for iOS AND Android Native Extension, and you wish to build both versions of the app from one code base, AdMob.init() takes an optional second Publisher ID parameter. In this case, the first ID you enter will be the Android ID, and the second will be the iOS ID.

```

if(AdMob.isSupported)
{
    AdMob.init("YOUR_ANDROID_ID_AD_ID", "YOUR_IOS_AD_ID");
}
else
{
    trace("AdMob won't work on this platform!");
    return;
}

```

Showing Test Ads During Development

To avoid making ad impressions while testing, you can enable a list of test device IDs.

Be sure to REMOVE this method call before publishing your final application. (Otherwise, your end users will see test ads too!)

```

// remove this line when you're done testing!
AdMob.enableTestDeviceIDs(AdMob.getCurrentTestDeviceIDs());

```

Create and Display a Banner Ad

Show your first ad. The following call displays an ad that will automatically fill the current width of the screen, and remain top-center aligned:

```

AdMob.showAd(AdMobAdType.SMART_BANNER, AdMobAlignment.CENTER, AdMobAlignment.TOP);

```

Optional: Loading Interstitial Ads

Interstitial ads are a special, full screen ad type that can produce higher revenue for your application than traditional banners.

To display an interstitial ad, call the loadInterstitial() function. The first parameter is the Ad Unit ID for the interstitial (see "Register Your App With AdMob" above.)

```

// load interstitial, and show it when its ready
AdMob.loadInterstitial("your interstitial ad unit id", true);

```

The second parameter tells the extension to show the interstitial automatically as soon as it's loaded.

If you are targeting both iOS and Android with the same code, put the Android Interstitial Ad Unit ID as the first parameter, and the iOS Interstitial Ad Unit ID as the third parameter:

```

// setting different interstitial ids for android and ios.

```

```
AdMob.loadInterstitial("android interstitial ad unit id", true, "ios ad unit id" );
```

Important Notice for Starling/Stage3D Users on Android

Some Interstitial Ads may use GPU rendering. If you are using Starling or Stage3D directly, then your stage is also rendered with GPU, and in some cases, interstitial ads may not display as expected unless you handle context changes appropriately on Android.

If you are using Starling, this means you need to be using the latest version of the Starling framework, the latest version of the Adobe AIR SDK, and you should be setting `handleLostContext` to `true`:

```
Starling.handleLostContext=true;
```

You also should ensure that your application handles `NativeApplication`'s state events by properly stopping and starting the starling context, i.e.:

```
NativeApplication.nativeApplication.addEventListener(flash.events.Event.ACTIVATE,
onStageActive);
```

```
NativeApplication.nativeApplication.addEventListener(flash.events.Event.DEACTIVATE,
onStageDeactivate);
```

```
function onStageActive(e:flash.events.Event):void
{
    starling.start();
}
```

```
function onStageDeactivate(e:flash.events.Event):void
{
    starling.stop(true);
}
```

If you are using stage3D without starling, you need to ensure that your application correctly stops/starts rendering of stage3D in response to these events and handles `CONTEXT3D` create/loss events appropriately for Android.

Optional: Preloading Interstitial Ads

Optionally, you can begin loading an interstitial at the beginning of your app, but not display it until later. This is useful for giving the ad time to load, so it's ready to display during a quick transition such as a game over screen.

By setting the second parameter of `loadInterstitial` to `false`, the extension will not automatically show the ad when it's loaded, but instead hold it in memory.

Important Note: You cannot preload a new interstitial until the last interstitial has been *dismissed* (the `AdMobEvent.SCREEN_DISMISSED` event with `isInterstitial=true`.)

```
// preload interstitial, but don't show it until you're ready
AdMob.loadInterstitial("your interstitial ad unit id", false);
```

When the preload is complete, an `AdMobEvent.RECEIVED_AD` event will be dispatched, and its `isInterstitial` property will be set to `true`:

```
AdMob.addEventListener(AdMobEvent.RECEIVED_AD, onReceiveAd);
```

```
function onReceiveAd(e:AdMobEvent):void
{
    if(e.isInterstitial)
    {
        trace("Interstitial has loaded");
    }
}
```

You can then use the `AdMob.showPendingInterstitial()` function to display the preloaded ad. (You may also check the `AdMob.isInterstitialReady()` function to check if a preload is complete at any time):

```
if(AdMob.isInterstitialReady())
{
    AdMob.showPendingInterstitial();
}
```

Optional: Handling Multiple Banner Ad Units

Some developers may wish to use more complex tracking by placing different banner ad units at different locations in the game. By default, the extension will use the ad unit ids for banners you passed to `AdMob.init()`. You can change these ad unit IDs at any time by calling `AdMob.setBannerAdUnitID()`:

```
// setting a new banner ad unit id (one platform)
AdMob.setBannerAdUnitID("YOUR_BANNER_AD_UNIT_ID");

// setting new banner ad unit ids for android and ios separately
AdMob.setBannerAdUnitID("YOUR_ANDROID_AD_ID","YOUR_IOS_AD_ID");
```

Optional: COPPA Support for Children's Apps

If your application is directed at children, you should call `AdMob.setChildDirected(true)`:

```
// for children's apps only (COPPA compliance)
AdMob.setChildDirected(true);
```

Optional: Hiding or Destroying Ads

You may wish to remove an ad from the screen- either by destroying it entirely, or temporarily making it invisible.

For instance, if your application is switching to a sub menu momentarily, and you do not want the ad to display, you might set it be invisible while the menu shows, and then visible again when the menu is complete.

```
// hide an ad temporarily
AdMob.setVisibility(false);

// to show it again...
AdMob.setVisibility(true);
```

Note that the extension can not set the visibility of an ad that does not exist; if your ad has been destroyed and you call `setVisibility()`, an error will be thrown. You can use a try/catch block to catch this error condition:

```
try
{
    AdMob.setVisibility(false);
}
```



```

}
catch(e:Error)
{
    trace(" no ad present to change visibility of!");
}

```

If your application is switching to a mode where the ad will not be seen again for a long time, or ever, you may prefer to destroy the ad entirely. In this case, the ad will stop making requests so you can avoid wasting memory and ad inventory:

```

// destroy an ad entirely.
AdMob.destroyAd();

```

Optional: Add Event Listeners

All event listeners are optional; however, they can be very useful for debugging if you're having trouble loading ads, or to fine tune your app's behavior when ads change state.

1. The `AdMobErrorEvent.FAILED_TO_RECEIVE_AD` event will fire if an error occurs reaching the ad network.

```

AdMob.addEventListener(AdMobErrorEvent.FAILED_TO_RECEIVE_AD,onFailedReceiveAd);

function onFailedReceiveAd(e:AdMobErrorEvent):void
{
    trace("Ad failed to load.");
}

```

The events `AdMobEvent.SCREEN_PRESENTED`, `AdMobEvent.SCREEN_DISMISSED`, and `AdMobEvent.LEAVE_APPLICATION` are optional, but can be useful for managing state in your application. For instance, you might want to pause your game or turn off the music between `SCREEN_PRESENTED` and `SCREEN_DISMISSED`, as the user will be viewing an ad during that time.

```

AdMob.addEventListener(AdMobEvent.RECEIVED_AD,onReceiveAd);
AdMob.addEventListener(AdMobEvent.SCREEN_PRESENTED,onScreenPresented);
AdMob.addEventListener(AdMobEvent.SCREEN_DISMISSED,onScreenDismissed);
AdMob.addEventListener(AdMobEvent.LEAVE_APPLICATION,onLeaveApplication);

function onReceiveAd(e:AdMobEvent):void
{
    trace("ad has loaded!");
}

function onScreenPresented(e:AdMobEvent):void
{
    trace("modal is showing...you might want to pause or stop sound here.");
}

function onScreenDismissed(e:AdMobEvent):void
{
    trace("modal closed. You might want to resume the game here if paused.");
}

function onLeaveApplication(e:AdMobEvent):void
{
    trace("ad took user out of app.");
}

```

5. Build the Application

Flash Builder 4.6 or higher:

Flash Builder 4.6 has full support for native extensions.

1. Make sure you've included the ANE files as described in Section 2, '*Include the Library.*'
2. Make sure that the extension is enabled for your target configuration.
3. Build your application and deploy as usual.

Flash Professional CS6+:

Flash CS6 has full support for native extensions.

1. Make sure you've included the ANE files as described in Section 2, '*Include the Library.*'
2. Build your mobile application and deploy as usual.

6. Troubleshooting

Why aren't my ads showing?

- Ensure that your ad unit ID is correctly set in *AdMob.init()*.
- Ensure you have included both the AdMob and GoogleServices ANE files.
- Ensure you have ad inventory. You can guarantee an ad will display by setting your test devices IDs- see "Showing Test Ads During Development" above.
- Check for errors retrieving ads:

```
AdMob.addEventListener(AdMobErrorEvent.FAILED_TO_RECEIVE_AD,onFailedReceiveAd);  
...will allow you to receive notices of why ads are not displaying.
```
- On Android, Use 'android-sdk/platform-tools/adb logcat' to review full debug text from the AdMob plugin.
- Some functions may throw an error if the Ad is not in a valid state for the operation- for instance, trying to destroy an ad before creating one. Use a try/catch block around the call to catch any errors:

```
try  
{  
    AdMob.destroyAd();  
}  
catch(e:Error)  
{  
    trace("could not destroy ad:"+e.message);  
}
```

Why do I get "manifest" errors when attempting to compile my application?

- Make sure that your application.xml changes are correct, as per "Update Your Application Descriptor" above. Double check you do not have any extra spaces or line breaks.

Why do I get ADT / Java compiler errors when attempting to compile my application?

If your Windows system, or local java installation, are set to a non-english language that uses special characters, you may need to change the current system language to English and restart.

How do I use the AdMobExample.as file in Flash Professional CS6?

- First, create the application, add the extension, and update AdMob by following this guide, Sections 1-4.
- Copy and paste AdMobExample.as into the same folder as your .fla. Do not copy and paste its contents on to the timeline. That will not work.
- In Flash properties, under 'Document Class', type 'AdMobExample' (no quotes) and press OK.
- Edit the AdMobExample.as file, and change the "your_publisher_id_here!" property to your own publisher ID from the AdMob control panel.
- Build and install the application.

I'm not a programmer, what's some minimal code I can copy and paste on to my Flash timeline on Frame 1 to make this work?

The code below will work for that. You still need to set up your application on admob.com and set up the extension in Flash though (Sections 1-4 above!):

```
import com.milkmangames.nativeextensions.*;
import com.milkmangames.nativeextensions.events.*;
if(AdMob.isSupported)
{
    AdMob.init("you must type your ad unit id here");

    // remove this next when you're done testing!
    AdMob.enableTestDeviceIDs(AdMob.getCurrentTestDeviceIDs());

    AdMob.showAd(AdMobAdType.SMART_BANNER,AdMobAlignment.CENTER,AdMobAlignment.TOP);
}
```

Why do ads display incorrectly on Android, or having a missing close button?

- You must handle context changes correctly if you are using Starling or Stage3D – refer to page 6, "[Important Notice for Starling/Stage3D Users on Android](#)".

Why doesn't the extension work when I run my swf on my Mac / BlackBerry / Windows Computer?

- The AdMob platform relies of functions of the mobile operating system for the version of the extension you are using (iOS or Android). The extension will only work when you run it on an Android or iOS phone or tablet.

I still need help! How can I get technical support?

- Your purchase comes with free technical support by email – just send us a message at support@milkmangames.com . We're here to help! Please remember that...:
- We're open during United States business hours, excluding U.S. Holidays, Monday-Friday, Pacific Standard Time. We strive to answer all support email within 24 hours (not including weekends and holidays) but usually do so much faster. Remember that we may not be in the same time zone.
- Please remember to mention: which extension you're having a problem with, what IDE you're using (such as 'Flash CS6' or 'Flash Builder 4.6'), and what device you're targeting. If you're experiencing an error message, please specify what that message is.
- We don't provide tech support through blog comments, Facebook, or Twitter. Please email us and we'll be happy to help you out!