Google Analytics AIR Native Extension

http://www.milkmangames.com

For support, contact support@milkmangames.com

Before you begin:

To View full AS3 documentation, see 'docs/as3docs/index.html'.

This extension requires the AIR 16.0 SDK or higher, which you can get at http://www.adobe.com/devnet/air/air-sdk-download.html ***.***

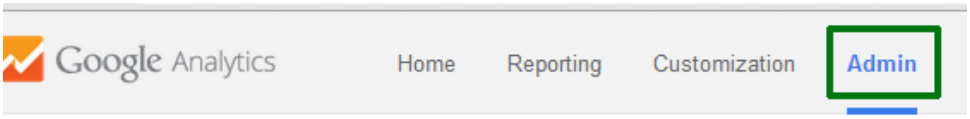
Review 'example/GAnalyticsExample.as' for a sample application.

Note that GAnalyticsExample.as is a Document Class. If you're a Flash Professional user and don't know how to use a document class, see the appendix "Using the Example Class in Flash CS6" at the end of this guide.

1. Setup Your App on the Google Analytics Website

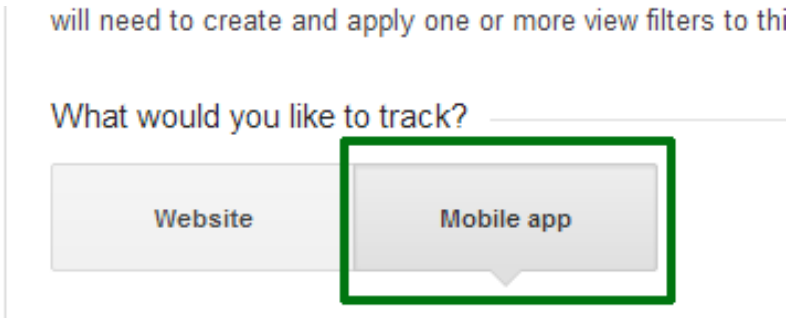
Before you can track your application with Google Analytics, you must set it up on the Google Analytics website (free).

1. Go to http://analytics.google.com/ and login to your account, or create a new account.
2. From the main page, Select the Admin tab.

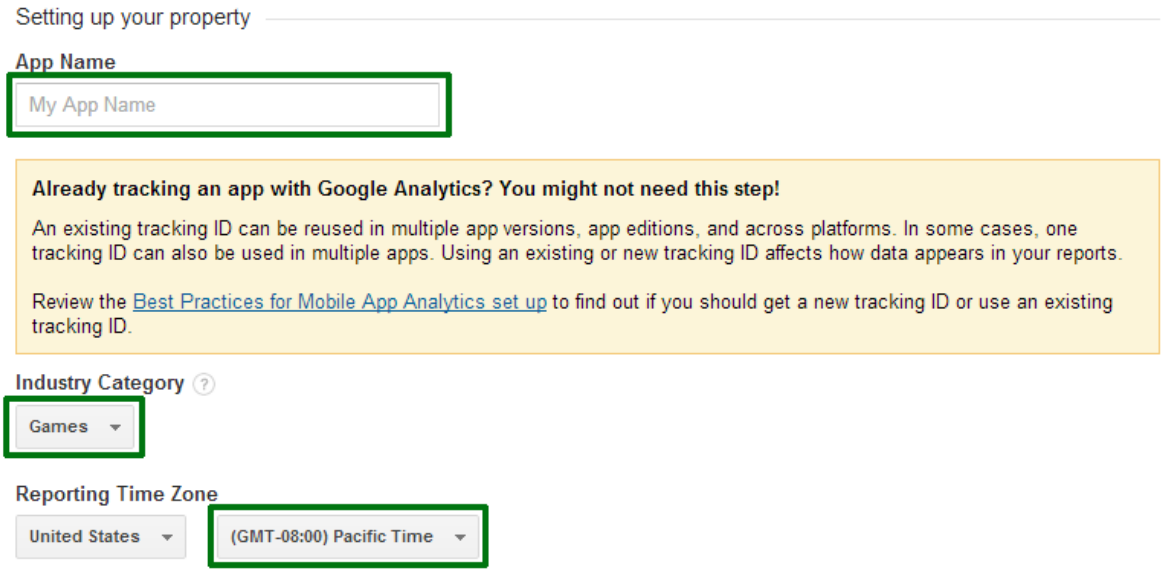


3. In the Property column, select "Create a New Property" from the dropdown menu.

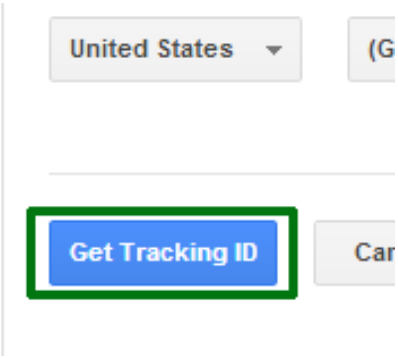
4. Choose "Mobile app" for "What would you like to track"?



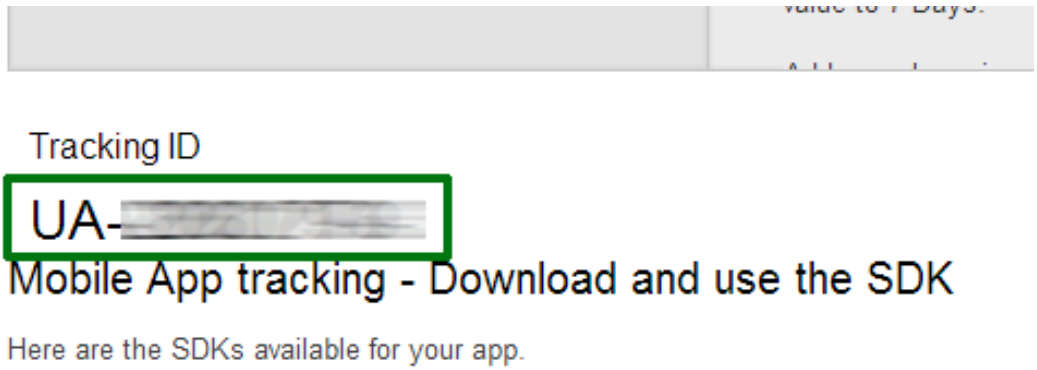
5. Fill in an App Name, Category, and Time Zone for your application.



6. Press "Get Tracking ID".



7. Copy and save your Tracking ID from the next page. You'll need it later to set up your app in ActionScript.



2. 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\sdks .
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.

3. Include the Analytics and GoogleServices Extension Library

Add the com.milkmangames.extensions.GAnalytics.ane AND com.milkmangames.extensions.GoogleServices.ane library to your project. This is located in the /extension folder of the extension package zip file.

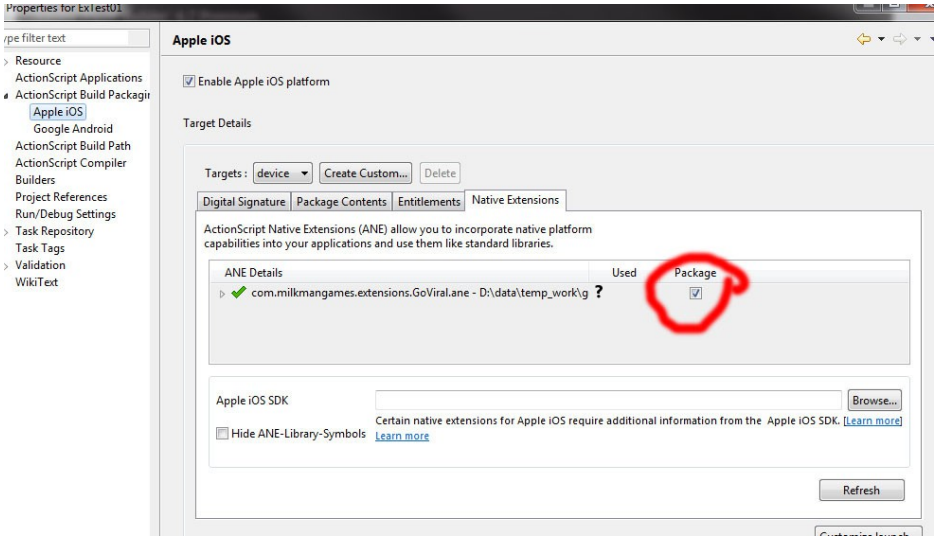
In Flash Professional:

1. Create a new project of the type 'AIR for iOS' (or AIR for Android)
2. Select File > Publish Settings...
3. Select the wrench icon next to 'Script' for 'Actionscript Settings'
4. Select the Library Path tab
5. Press the 'Browse for Native Extension (ANE) File' button and select the com.milkmangames.extensions.GAnalytics.ane file. Press OK.
6. Press the 'Browse for Native Extension (ANE) File' button and select the com.milkmangames.extensions.GoogleServices.ane file. Press OK.
7. Select the wrench icon next to 'Target' for Player Settings
8. *(For Android Only)* Select the 'Permissions' tab, and enable INTERNET and ACCESS_NETWORK_STATE permissions.
9. *(For Android Only)* **Check the 'Manually manage permissions and manifest additions for this app' box**.
10. Make sure you're using the AIR 16.0 SDK or higher per the instructions in Section 3.
11. Select File>Publish Settings...
12. For 'Target' Select 'AIR 16.0 (or higher) for iOS' (Or Android)

In Flash Builder 4.6+:

1. Go to *Project Properties*
2. Select *Native Extensions*
3. Choose *Add ANE...* and navigate to the com.milkmangames.extensions.GAnalytics.ane file
4. Choose *Add ANE...* and navigate to the com.milkmangames.extensions.GoogleServices.ane file

5. Make sure the 'package' box is checked for your target, under Actionscript Build Packaging:



6. Make sure you're using the AIR 16.0 SDK (or higher) per the instructions in Section 3.

4. Update Your Application Descriptor

You'll need to be using the AIR 16.0 SDK or higher, and include the extension in your Application Descriptor XML. For an example, see **'example/app.xml'**.

Set the AIR Version and Extension IDs

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 in the descriptor:

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

For Android Only: Update the Android ManifestAdditions

(For Android Only): The extension requires a few android-specific permissions and activities to be defind in your XML file. Edit your application.xml's '<android>' section to look like the one below (**NOTE: these settings have CHANGED between v1 and v2. Review below carefully or copy and paste from /example/app.xml!)**:

```
<android>
<manifestAdditions><![CDATA[
<manifest android:installLocation="auto">

<!-- These permissions are required by GAnalytics -->
```

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

<application>

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

  <!-- Used for Google Play Store Analytics Campaign Measurement -->
  <service android:name="com.google.android.gms.analytics.CampaignTrackingService" />
  <receiver android:name="com.milkmangames.extensions.android.ganalytics.CampaignTrackingReceiver"
android:exported="true">

    <intent-filter>
      <action android:name="com.android.vending.INSTALL_REFERRER" />
    </intent-filter>
</receiver>

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

5. ActionScript Initialization and Basic Setup

The Google Analytics extension can be up and running in a few simple calls. See '**example/GAnalyticsExample.as**' for a full example.

1. Import the API Classes:

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

2. First, check to make sure that Google Analytics is running on a supported platform, using the `isSupported()` method. Google Analytics only works on Mobile devices (Android and iOS). If it's supported, you can call the create() method, passing in your Google Analytics Tracking ID, retrieved in section 1, "Setup Your App on the Google Analytics Website":

```
if(GAnalytics.isSupported())
{
	// pass your GA Tracking ID to this function when your app starts up.
	GAnalytics.create("UA-123456-23");
}
else {
	trace("Google Anaytics only works on iOS or Android.");
}
```

6. Tracking Hits

When you track a "hit" or event with the extension, it gets recorded by Google Analytics. Google Analytics supports a variety of different hit types explained below.

Tracking App Views

Whenever the user views a new screen or location in your app, you should track it with the `trackScreenView()` function. **You should always track a view right when your application starts, as this will be used to designate the beginning of the user's session in Analytics.**

```
//  when the user goes to a new screen in your app, you can track it by name
GAnalytics.analytics.defaultTracker.trackScreenView("main menu");
```

Tracking App Events

Most of what you track in your application will likely be custom "events"- anything from the pressing of a button, to using a particular powerup in a game. The following examples show the use of the `trackEvent()` function to track custom events:

```
//get a notificaiton when the internet connection is lost or reestablished...
// events can be used to track any kind of action in your app or game.
// in this example, we record a "click" in the "uiEvents" category of
// a button labeled "Button 1"
GAnalytics.analytics.defaultTracker.trackEvent("uiEvents", "click", "Button 1");

// events can be used to track any kind of action in your app or game.
// in this example, we record a "magic" spelled used in the "fightingMoves"
// category with a label "Magic Spell" and 3000 magical power hit points
GAnalytics.analytics.defaultTracker.trackEvent("fightingMoves", "magic", "Magic Spell", 3000);
```

Tracking Social Network Events

You can track interactions with Social Networking features (such as those provided by the GoViral Native Extension) using the `trackSocial()` function:

```
// you can track interactions with social media services.  In this example, we report
// that the user has done the action "shareLink" with "Facebook", and that the relevant
// content was "http://www.milkmangames.com"
GAnalytics.analytics.defaultTracker.trackSocial("Facebook", "shareLink", "http://www.milkmangames.com");
```

Tracking In-App Purchase Events

Google Analytics can be used to track in-app purchases you make within your mobile applications, such as with the iOS In-App Purchase or Google Android Billing Native Extensions. You would typically call these functions inside the "onPurchaseSuccess" event listeners for such an extension, to indicate to Google Analytics that a purchase has taken place.

Tracking a purchase involves two calls: `trackTransaction()` with information about the transaction, followed by `trackItem()` with information about the particular item that was purchased.

The following examples show the use of the `trackTransaction()` and `trackItem()` functions to track the purchase of item, with a transaction id of "abc-def-ghi", from the "App Store", for $1.99. The item name is "Test Item" and its sku is "testitemsku":

```
// when an in-app purchase is completed, you can track it by calling trackTransaction
// followed by trackItem.  Both should use the same transactionId propert (in this
// example case, "abc-def-ghi") that is unique to this particular purchase.
GAnalytics.analytics.defaultTracker.trackTransaction("abc-def-ghi", "App Store", 1.99);
GAnalytics.analytics.defaultTracker.trackItem("abc-def-ghi", "Test Item", "testitemsku", 1.99);
```

Tracking Errors

When an error occurs in your application, you can track it in Google Analytics using the `trackException()` method. The following example causes an error to be thrown in Actionscript, and records it to Google Analytics inside the try /catch block:

```
// you can track errors and exceptions in your app with google analytics.
// in this example, we make call that which will cause an
```

```
// error in actionscript.  we the report that error to analytics.
// the 'false' indicates that it's not a fatal error.
try {
	var object:Object=null;
	object.functionDoesntExist();
} catch (err:Error) {
	GAnalytics.analytics.defaultTracker.trackException(err.message, false);
}
```

Tracking Custom Timings

Google Analytics can track periods of time measured in milliseconds, which can be useful for measuring performance on your user's devices. The following example records the time before the "`loadAssets()`" function, the time after its completed, then sends the time it took to load to Google Analytics using the `trackTiming()` function. (Note that the time is in milliseconds, and that there is no built-in ActionScript fucntion called "loadAssets" - this is just an example):

```
// get the time before the function we're testing
var before:Number=new Date().getTime();
// do something that might take awhile
loadAssets();
// get the time it took
var after:Number=new Date().getTime();
var timeItTook:Number=after-before;
// send the timing to Google Analytics, in the category "loadtimes", with the
// label "Load Assets" and the event name "loadAssets":
Ganalytics.analytics.defaultTracker.trackTiming("loadtimes", timeItTook, "loadAssets", "Load Assets");
```

7. Additional Options And Settings

The Google Analytics extension supports a number of additional options and parameters, described below.

Setting User Opt-Out

In some cases, the user may wish to "opt out" or "opt in" to Google Analytics tracking. You can set whether they are "opted out" using the `setOptOut()` method, and get their current opt-in state using the `getOptOut()` method:

```
// if the user wants to opt out of being tracked, you can call setOptOut(true)
Ganalytics.analytics.setOptOut(true);
log("Opted out status:"+GAnalytics.analytics.getOptOut());
```

Setting Dry-Run Mode

If you are running your application repeatedly for testing, you may wish to disable tracking temporarily in the Google Analytics dashboard, such that you or your team's own debugging does not affect your statistics. Setting "Dry Run" mode on prevents any hits you send from affecting the dashboard. (Just remember to set it back when you're done!) You can set Dry Run mode using the `setDryRun()` method, and get the current Dry Run state using the `getODryRun()` method:

```
// turns on Dry Run mode, so any hits won't show up on the Google Analytics website
Ganalytics.analytics.setDryRun(true);
log("Is dry run mode?:"+GAnalytics.analytics.getDryRun());
```

Forcing Dispatch of Cached Hits

By default, the extension will cache the hits that you track together, and send them over the network periodically to conserve bandwidth usage. If you want to force pending hits to be dispatched immediately, use

the `forceDispatch()` method:

```
// by default, Google Analytics will batch events and send them
// to the server perioidically.  this method forces all queued
// hits to be sent right away.
GAnalytics.analytics.forceDispatchHits();
```

Setting Custom Tracker Fields

You can set custom field values that will be used with every subsequent hit event you make. One use for this is associating a User ID with the user, as shown below. (Make sure you are familiar with Google's User ID Policy before using this example):

```
// set a field that will be used by all hits you track.
// in this example, we're setting the user ID.
// Make sure you are familiar with Google's user id policy
// (https://developers.google.com/analytics/devguides/collection/protocol/policy)
// before using this in an app.
GAnalytics.analytics.defaultTracker.setTrackerField("&uid", "bob@gmail.com");
```

Tracking Custom Dimensions and Custom Metrics

If you've defined a Custom Metric or Custom Dimension in the Google Analytics web panel, and been assigned an index number for the dimension or metric, you can set it on on any function call with an 'extraParams' object parameter, using the key format of "&cd1" and "&cm1", as the keys for Custom Dimension with Index 1 and Custom Metric with Index 1, respectively. For instance, the following example adds a Custom Dimension with the index 1 and a Custom Metric with the index 2 to a screen view tracking:

```
GAnalytics.analytics.defaultTracker.trackScreenView("homeScreen", {"&cd1":"my metric", "&cm2":"39"});
```

Enabling Ad ID Collection

You can enable Advertising ID collection for more detailed demographic information with the `setAdvertisingIdCollectionEnabled()` function.

VERY IMPORTANT NOTE FOR IOS USERS: You can't set this to true on iOS unless your app also serves AdMob or other types of ads- otherwise, Apple will reject your application. Additionally, you must also include GAIDFAAccess.ane to enable this feature on iOS. On android, do whatever you want, Google don't care.

```
// enables ad id collection - read above before using on iOS!
GAnalytics.analytics.defaultTracker.setAdvertisingIdCollectionEnabled(true);
```

8. Campaign Referral Tracking

As long as you've setup your application.xml file correctly, the extension will automatically track Install Referrals from the Android Google Play Store, and link referrals on iOS. If you are working with an affiliate who will be sending custom campaign referrals to your app on iOS, ask your affiliate for any changes required your iOS app's Plist file. (These changes will go into the <InfoAdditions> block in application.xml).

If your application was installed from the Google Play Store on Android, and had a referrer value, you can retrieve that value using the getInstallReferrer() method:

```
// example result: "utm_source=someAdCompany"
```

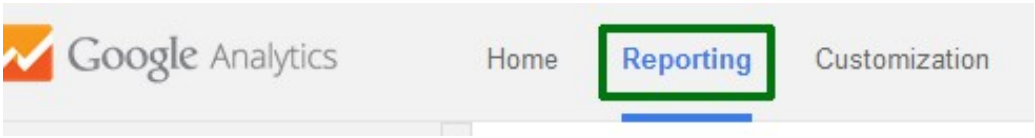
```
var referrer:String=GAnalytics.analytics.getInstallReferrer();
```

9. Testing Your Implementation

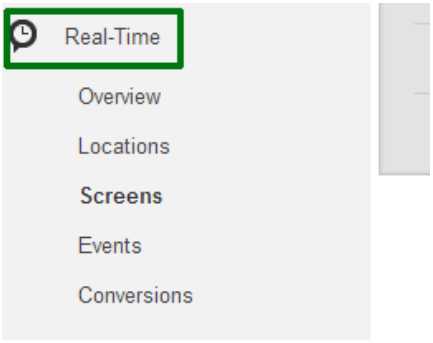
The Google Analytics dashboard's data is updated once a day, so you won't see the results of your work immediately in the main dashboard.

However, you can use the “Real Time” view to make sure that your implementation is working:

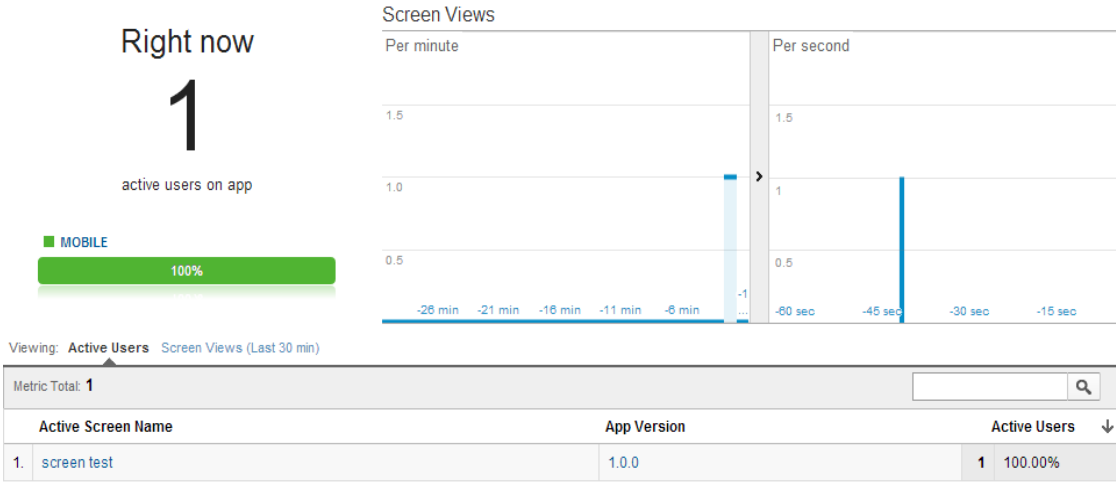
1. Goto https://analytics.google.com and login to your account.
2. Choose the Reporting tab.



3. From the left panel, Choose Real Time.



4. Cause some hits to occur in your app. You should see screenView hits appear in the 'Screens” panel, Events hits in the “Events” panel, and so on.



5.

10. Troubleshooting Common Problems

"Why is my app not working properly on Android?"

- *Carefully reread the 'Update Your Application Descriptor' instructions and make sure you've updated the descriptor with the correct android data. These changes are required for Android.*
- *If you're using Flash Professional version CS6, there is a bug in the IDE can cause your XML changes to be overriden. To avoid this, make sure you publish your app by Selecting File>Publish or Debug>Debug Movie, instead of File > Publish Settings... > Publish to avoid this.*

"What iOS/Android versions does Google Analytics Extension support?"

- *Google Analytics will work on any Android device running Android 2.3 or higher, and any iOS Device running iOS 6 or higher.*

"Why don't I see data in real time reports?"

- *It may take a day or more after your initial publication before real time data is logged on the Google Analytics website.*
- *If you're still not seeing data, go to the "Admin" area, select your project, and create a new "View".*

- **"How do I use the GAnalyticsExample.as file in Flash Professional?"**
- First, setup the ANE and descriptor in your IDE, as shown in Sections 1-4 above .
- Copy and paste the GAnalyticsExample.as FILE 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 'GAnalyticsExample' (no quotes) and press OK.
- In GanalyticsExample.as, change YOUR::TRACKING_ID to your actual Google Analytics tracking ID from the Google Analytics website.
- GAnalyticsExample.as is a complete application. It draws a user interface that lets you test everything Google Analytics can do using buttons. It is not meant to be used "on top of" an existing Flash app (unless you want your game to be covered in gray example buttons!)
- Build and install the application.

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

- *The extension uses features that are built into the iOS and Android operating systems. The extension will only work when you run it on a 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!