KANTAR SIFO

Developer's Guide

Kantar Sifo Mobile Analytics SDK for Android

Table of contents

Overview

The Kantar Sifo Mobile Analytics SDK helps you to measure usage of mobile application using Kantar Sifo's services. In order to measure traffic, your application needs to send HTTP requests to a server provided by Mobiletech or Codigo Analytics, using URL:s following a specified pattern with information about your application.

Each of these HTTP requests is called a "tag". To get a good measure of the usage, your application should send a tag every time a new view or page of content is shown to the user. This tag should contain information about what content the user is seeing. For example when the user opens the application and its main view is shown, a tag should be sent – telling the server that the main view is displayed. When the user shows a list, an article, a video or some other page in your application, another tag should be sent, and so on.

The SDK can help you with the whole process, both creating these URLs, as well as sending them to the server. The only thing it needs from you is for you to tell it when a view has been shown, and what content it has. It also needs some information about your application, which is specified later in this document.

What's new?

Version 3 of this SDK is a major update which has been refactored and adapted to use modern Android standards. For example:

- Deprecated libraries for cookie handling and http connections have been replaced.
- Data is by default sent using https, with an option to use http if you prefer.
- A new sample application is included as source code and compiled apk file.

For more detailed changes of each version, see release-notes.txt in the SDK root folder.

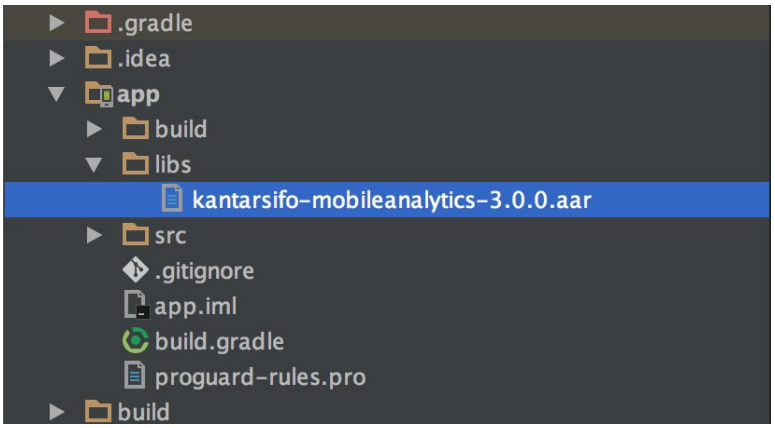
SDK contents

The SDK contains the following:

- **Framework**: The binary file that contains the API and functions to send tags in a correct way. The file is called *kantarsifo-mobileanalytics-3.X.X.aar*.
- **TestApp:** An Android application that uses the framework and prints out logs when sending tags. You can compare it to the logs printed by your own application to verify your implementation. See also Test Application below.
- **Javadoc:** API documentation of public methods in the framework. Open index.html in your web browser to view a complete specification of these classes and methods in the framework.
- **Developer's Guide.pdf**: This document.

Getting started

Follow the below steps to import the framework into your project.



Step 1: Copy *kantarsifo-mobileanalytics-3.X.X.aar* file to the *libs* folder.

```
dependencies {
    compile(name:'kantarsifo-mobileanalytics-3.0.0.aar', ext:'aar')
}
```

Step 2: Open *build.gradle* file and add a dependency to the .aar file

```
allprojects {
   repositories
     {
        flatDir { dirs 'libs' }
     }
}
```

Step 3: Add `flatDir {dirs 'libs'}` like above.

Basic parameters

- **Customer id** (cpid - 4 digits or 32 characters)
  This is your Customer ID provided by Kantar Sifo. This information will be included in the attribute called **cpid** in the tags sent to the server. Cpid can be one of two formats; Mobiletech IDs with 4 digits and Codigo IDs with 32 characters. The framework will automatically choose either Mobiletech or Codigo endpoints depending on the cpid.

- **Application name** (max 88 characters)
  The name of your application. Choose a name that are representative of the application. If you have both Android and iOS version of app, use the same name. The framework will add platform (e.g. “Android”) to URL parameters, so do not add platform to the name.

- **Category** (cat) - main (max 255 characters)
  The current category or name of the page the user is browsing. This value tell us what the user is doing. If the user is reading an article about sports in your application the value should be “sports”. If the user is browsing a set of football scores the value should be “sports/scores/football”. If the user is in the main view of the application, the value should be “main” or similar. If the user is watching streaming/web tv the value chain should start with “stream”. This information will be included in the attribute called “cat” in the tag. See also FAQ, and contact Kantar Sifo for more info regarding what structure of category values you shall use and what you can not use.

- **Content id** (id) - optional ,contact sifo before use (max 255 characters)
  The value current article, resource or content within the category that is being browsed. If the current category does not provide different content, this value is not needed. This information will be included in the attribute called “id” in the tag.

- **Content Name** (name) - optional ,contact sifo before use (max 255 characters)
  The name of the current article, resource or content within the category that is being browsed.

Integration with Sifo Internet app

Kantar Sifo collects information about how apps and websites are used in order to create for example statistics and insights for the app/website publisher. The software and service needed is made available once an agreement with Kantar Sifo is in place. The usage information captured can be associated with individuals in the Sifo Internet panel. The Sifo Internet panel consists of several thousand persons from the Sifo panel who have opted in to participate in the panel and install the Sifo Internet app on their mobile devices. For more information about the Sifo panel please go to:
https://www.kantarsifo.se/undersokningsdeltagare/sifopanelen

Sifo Internet app integration is available to both WebView based apps and native apps. The purpose of this integration is to identify the user as a certain panelist. This is achieved by receiving a panelist Id string from the Sifo Internet app, and setting that Id as a cookie in your app’s shared CookieStore. This communication is handled by with:

- Install “Sifo Internet” app from Google Play Store.
- Login to app and stay logged in while running your app.

Code implementation

Basic functionality

To get started with the tagging, you only need 4 methods.

- Use **`createInstance`** to create an instance of the framework and specify application name and customer ID.
- Use **`getInstance`** to get the instance of the framework that you created at any time in your application.
- For native apps, use **`sendTag`** to send a tag to the server when a view is shown in your application.
- For hybrid apps, use **`activateCookies`** to make a WebView automatically send tags when visiting webpages that have implemented tagging.

Implementation steps

Initialize the framework by calling the **`createInstance`** method as shown below.

```
TSMobileAnalytics.createInstance(new TSMobileAnalytics.Builder(getApplicationContext())
        .setCpId("my_cpId") //required
        .setApplicationName("my_application_name") // required
        .setPanelistTrackingOnly(boolean) // optional, default value is false
        .setLogPrintsActivated(boolean) // optional, default value is false
        .useHttps(boolean) // optional, default value is true
        .build());
```

...where cpid is a String holding your Customer ID and applicationName is a String with the name of your application. The method **`getApplicationContext()`** is an Android specific method for getting the context of the current application. The framework needs the context to get a device identifier to separate unique users in the statistics.

It can also be a good idea to make sure that **`createInstance`** does not return null, that means that something was wrong with your input data. If this happens you will also get an error print in the log.

Now use the following methods to send tags:

Native apps

In the **`onResume()`**-method of any Activity or any other place where your application displays a view or page that should be tagged:

```
if (TSMobileAnalytics.getInstance() != null) {
    TSMobileAnalytics.getInstance().sendTag(category, contentID);
 }

or

if (TSMobileAnalytics.getInstance() != null) {
    TSMobileAnalytics.getInstance().sendTag(category, contentName, contentId);
}
```

...
The framework will now send a tag to the server with the information you provided, using a background thread.

If you want to use a category structure with subcategories, such as “News>Sports>Football”, you can use an array of Strings to specify your category structure:

```
String[] categoryArray= {"News", "Sports","Football"};

if (TSMobileAnalytics.getInstance() != null) {
    TSMobileAnalytics.getInstance().sendTag(categoryArray, contentName, contentID);
}
```

...where **`categoryArray`** is an array of Strings specifying the category structure. A maximum of 4 categories are allowed in the structure, and their names must not be longer than 62 characters each.

If **`getInstance()`** returns null, it means that **`createInstance`** has not been called. It would be a good idea to take some action if this happens. If you turned on debug prints in the framework using **`setLogPrintsActivated(true)`** as described above, you can see if tags are being sent and received properly in the Android LogCat.

Options

In addition to parameter category, you may use additional parameters. Please contact Kantar Sifo before doing that. **`contentID`** is a String with the identifier of the current content and **`contentName`** is a String with any other information that should be sent in the tag. Maximum length of these parameters are 255 characters.

Hybrid apps

In the **`onCreate()`** method of any Activity with a WebView that should utilize the tags:

```
if (TSMobileAnalytics.getInstance() != null) {
    TSMobileAnalytics.getInstance().activateCookies();
}
```

This method will allow third-party cookies.

Debugging

There are several more advanced functions in the framework. The purpose of them is to help out if there are any problems or issues with the tagging that needs to be traced. It is possible to get information about pending requests, and to subscribe on notifications when a tag request has completed or failed, and more. To see some examples on how to use these functions, have a look at the source code for the test application.

Threads

The framework will take care of the threading for you, so you do not need to think about running the code in a separate thread.

String encoding

Strings sent to the server will be encoded using UTF-8. If the string given to the framework contains characters that are not supported by this encoding, these characters will not be stored correctly in the statistics.

FAQ

- **How do I test and verify that the framework is correct integrated?**
  See Test Application and Validation test with Kantar Sifo below.

- **Native app - how do I integrate the framework**?
  You have to both initialize the framework and send “page view” events in a correct way. This is a difference to “hybrid app” where page tagging is implemented separate to the framework.

- **Hybrid app - how do I integrate the framework**?
  First you have to check if pages shown in web view already are tagged with MobileTech or Codigo scripts (contact Kantar Sifo regarding this). Then you need only to implement **`activateCookies()`** on the framework instance.

- **How does Framework sync with Sifo Internet app**?
  If the device has the Sifo Internet app installed and is logged in, Sifo Internet writes panelist cookies as a json file to the internal storage, read by the framework.

- **Do I need to grant permissions to the framework to read files**?
  No, there is no need to explicitly grant permission.

- **Web TV / streaming - how shall we implement that**?
  Regarding web TV / streaming, it important to distinguish between measuring “preroll” and “start of clip”. Regarding what parameters to use, “preroll” shall not have

the same category value as **“Start of clip**”. As “start of clip” shall have **“streaming”** as top category value, preroll can have e.g. “preroll” or “play”.

Update strategy

Updates of the framework will be necessary to fix bugs, add features, handle changes on the servers or platform etc. For this reason we want to make updating of the framework as easy and seamless as possible.

When upgrading from version 2.X to 3.X, you need to do quite a lot of code changes, e.g. use the new package name **`se.kantarsifo.mobileanalytics.framework`** and use the new Builder constructor. Later on when a new version of the framework is delivered, it should be possible to simply replace the old .aar file with the new one.

To keeping updating as seamless as possible it is important that you:

- Inform us as soon as you find something that should be changed or improved
- Inform us as soon as you find a bug, memory or performance issue

Example files

Below is a Java code example of how you can use the framework for Android.

```
import se.kantarsifo.mobileanalytics.framework.TSMobileAnalytics;

public class MainActivity extends Activity { //...
public void onCreate(Bundle savedInstancestate) {

       TSMobileAnalytics.createInstance(
             new TSMobileAnalytics.Builder(getApplicationContext())
                 .setCpId(R.string.cpId)
                 .setApplicationName(R.string.application_name)
                 .setPanelistTrackingOnly(true)
                 .setLogPrintsActivated(true)
                 .useHttps(true)
                 .build());
      //...
}
//...

public void onResume() {

       if (TSMobileAnalytics.getInstance() != null) {
               TSMobileAnalytics.getInstance().sendTag(
               getString(R.string.mainPageName),
               "Contentname",
               "Content ID");
       }
       //...
```

```
}
//...
}

class OtherActivity extends Activity { private String articleID;
//...
public void onResume() {

        if (TSMobileAnalytics.getInstance() != null) {
                TSMobileAnalytics.getInstance().sendTag(
               getString(R.string.mainPageName),
               "Contentname",
               "Content ID");
        }
        //...
}
//...
}
```

For a more detailed example, please see the code for the test application.

Test application

The Android framework comes with an example application for you to test it with. The application will display a catalog structure with fake native content and real “hybrid content”. When a page or a category is browsed, the application will send a tag to the server with information. The application will print logs in the print log console called LogCat, which is included in the Android SDK, to show what information is sent and when.

If necessary, one could use tools to monitor network traffic, such as Charles Proxy or Fiddler, to verify that the network calls look ok.

Validation test with Kantar Sifo

Final step, contact Kantar Sifo (see contact info below) to verify that the sent tags have been correctly stored in the analytics databases.

Contact information

Technical questions:
peter.andersson@tns-sifo.se

Other questions:
claes.bostrom@tns-sifo.se
info@tns-sifo.se
08-50742000