StreamhubAnalytics Android plugin for Kaltura Android applications

28th of September 2020	Programs tracking Ads tracking Sample app Sample logs
27th of October 2021	Revised QoS logic

This document describes the steps required to add and configure the StreamhubAnalytics plugin for Kaltura Android applications.

Download the plugin and sample application at
https://streamhub-static-content.s3-eu-west-1.amazonaws.com/plugins/kaltura/releases/kaltura-android-plugin.zip

The plugin is composed of the following files and components:

1. **StreamhubAnalyticsKalturaBridge.java** is the bridge between a Kaltura android player and StreamhubAnalytics data tracking abstraction layer
2. **StreamhubAnalytics.java** is the data tracking abstraction layer
3. **NetworkUI.java** is handling Http REST requests and responses

The downloadable .zip file contains:

1. The plugin files
2. A sample Android Kaltura player project that integrates with the StreamhubAnalytics plugin and shows how to initialise it.

Getting Started

Integrate the plugin in your own application

The integration process is straightforward, because the complexity of interacting with our Backend REST API has been abstracted in the plugin logic.

The plugin is written on top on the Kaltura Android SDK
https://developer.kaltura.com/player/android/getting-started-android/

Add the dependency for HTTP I/O requests

We will assume you are using Android studio for this document. If another IDE is being used please just adapt the steps.

1. Open ***build.gradle*** in the Module level
2. Add the following to the “dependencies”

```
implementation 'com.squareup.retrofit2:retrofit:2.9.0'
implementation 'com.squareup.retrofit2:converter-scalars:2.9.0'
```

3. Go to the Android Studio menu and click on File > Sync Project with Gradle Files, and sync the Gradle file

Add the plugin files

Locate the 3 plugin files StreamhubAnalyticsKalturaBridge.java, StreamhubAnalytics.java and NetworkUI.java in the archive and copy them into your Android project.

Initialize the StreamhubAnalytics plugin

In your application code java class, create a class member to hold an instance of the StreamhubAnalyticsKalturaBridge class.

Then, in an appropriate location of your code, initialize it:

Java

```
import uk.co.streamhub.kaltura.StreamhubAnalyticsKalturaBridge;

...
new StreamhubAnalyticsKalturaBridge(player, this, "my-player-id", isLive,
"logged-user-id",
"my-analytics-id", "my-player-id");
```

Kotlin

```
import uk.co.streamhub.kaltura.StreamhubAnalyticsKalturaBridge
...
StreamhubAnalyticsKalturaBridge(
   player,
   this,
   "my-player-id",
   isLive,
   "logged-user-id",
   "my-analytics-id",
   "my-player-id"
)
```

Parameters details

Parameter name	**Type**	**Description**
player	com.kaltura.playkit.Player	The instance of the Player that the plugin will track.
context	android.content.Context	The application Context object
playerId	String	Unique identifier of the player playing the video. It is a unique identifier for the player in both your system and ours. You can generate your own unique identifier or contact your Streamhub account manager to get one generated for you.
isLive	Boolean	Specify if the video content is either a Live or a VoD program.
userId	String	If your application users are logged, you can use this field to provide any user identifier. This will allow you to get analytics detailed at the user level of your audience.

analyticsId	String	The main tracking code that has been provided to you by your Streamhub account manager.
playerTitle	String	A user-friendly name associated with the playerId. This information might have already been shared with your account manager during the pre-integration phase.

Your Streamhub account manager can help you with the initialisation parameters, if needed.

Build and test your application.

We recommend that you capture the logs HTTP requests in a similar format as the ones in the **/sample.logs** folder and send those to us for review.