StreamhubAnalytics Swift plugin for ThePlatform iOS applications

This document describes the steps required to add and configure the StreamhubAnalytics plugin for ThePlatform iOS applications.

Download the plugin and sample application at
http://streamhub-static-content.s3.amazonaws.com/plugins/theplatform/ThePlatform-ios-plugin.zip

The plugin is composed of the following classes:

1. StreamhubAnalyticsTPBridge.swift implements TPPlayerViewDelegate and is the only Swift class you will need to integrate with.
2. StreamhubAnalytics.swift
3. ShUserData.swift
4. NetworkUI.swift

The downloadable .zip file contains:

1. The 'StreamhubAnalytics plugin files' folder which points to the 4 classes mentioned above. These are alias and you need to take the actual files in the sample project provided
2. A sample iOS player project that makes use of the StreamhubAnalytics plugin and shows how to initialise the plugin (steps detailed below)

Prerequisites

The plugin is written is Swift 2.3. If your application uses a more recent version of the Swift language, then make sure the **Use Legacy Swift Language Version** setting is set to Yes under **Build Settings.**

The plugin has a dependancy on the AlamoFire library version 3.0 (https://github.com/Alamofire/Alamofire) to send HTTP requests to our collector.

To add the library you can rely on Cocoapods as we did in the sample project provided. Below is our Podfile:

```
source 'https://github.com/CocoaPods/Specs.git'
platform :ios, '8.0'
use_frameworks!
```

```
target 'PlayerDemoApp' do
        pod 'Alamofire', '~> 3.0'
end
```

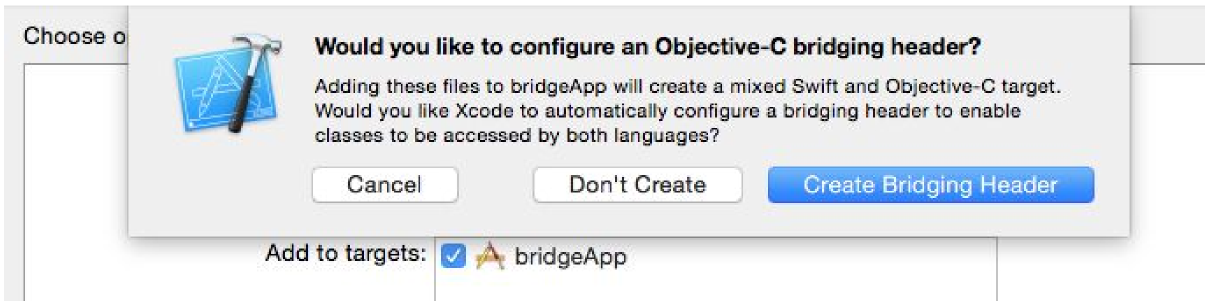
Then run ‘pod install’ in a terminal to install the library.
From now on, you have to open your application using the project file with extension .xcworkspace, no longer the .xcodeproj project file. This is a common practice in the Xcode ecosystem and as a Xcode developer you should be already familiar with it.

Getting started

The integration is very straightforward, as the complexity of interacting with our Backend REST API is being abstracted by the plugin.

Drop the plugin files into your player application that uses ThePlatform for iOS SDK (https://docs.theplatform.com/help/adk-for-ios)

As the result, XCode will prompt a window informing you that you are now creating a Objective-C / Swift hybrid app and as such need a bridging header. The role of this header is to actually let your Objective-C app code communicate with the Swift code of the plugin.


Would you like to configure an Objective-C bridging header?

Adding these files to bridgeApp will create a mixed Swift and Objective-C target. Would you like Xcode to automatically configure a bridging header to enable classes to be accessed by both languages?

You should now be able to import a ***{applicationModuleName}*-Swift.h** header file in your Objective-C own files and access the Swift function from your objective-c code.

For example, if you application is called ‘MyApp’ you are now able to add the following line in your objective-c files:
#import "MyApp-Swift.h"

Initialize the StreamhubAnalytics plugin

In you ViewController, add a property to hold a StreamhubAnalyticsTPBridge instance as such; @property StreamhubAnalyticsTPBridge * streamhub;

Then, in the *viewDidLoad* initialization function (or any other suitable function), create an instance as this;

```
self.streamhub = [[StreamhubAnalyticsTPBridge alloc] initWithPartnerId:@"mpx"
endPoint:@"http://stats.streamhub.io" playerId:@"ios-swift-tp-player" isLive:NO userId:@""
analyticsId:@"streamhub-5812d" userAgent:@"(iPad; U; CPU OS 9_2 like Mac OS X; en-us;
iPad5,3)"];
```

Pay attention to the userAgent parameter. This is the value that will be used to attribut the views to a specific type of device.
As for native apps you will not be able to retrieve a dedicated user-agent String programmatically, we recommend that you derive the user-agent from one of these examples, depending on which device your are targeting:

AppleTV
com.example.apple-samplecode.TVMLCatalog/1.0 iOS/9.0 model/AppleTV5,3 build/13T5365h.

iPad
iPad; U; CPU OS 9_2 like Mac OS X; en-us; iPad5,3

iPhone
iPhone; CPU iPhone OS 7_0 like Mac OS X

It is a good idea to request our validator tool for your integration (https://bitbucket.org/fivecool/streamhub-stats-collector-validator) to test the user-agent string you are providing. Contact tony@steamhub.co.uk for relevant informations.

Parameters details

name	type	comments
partnerId	String	provided to you by your Streamhub integration engineer
endPoint	String	`http://stats.streamhub.io` is the production endPoint for submitting stats. However, you could choose to use your own mockup server during the plugin development phase.
isLive	Boolean	specify if the video content is either a Live or a VoD program. If the content is Live, then, we usually require that you provide a channel identifier for the programs being played by your video player. See the API method **setChannelId** below.

		For any details, please contact your Streamhub integration engineer.
playerId	String	unique identifier of the player playing the video. It is a unique identifier for the player in both your system and ours. If any clarification is needed, please contact your Streamhub integration engineer.
userId	String	is used if you are able to track your user via a user identifier. If so, you will be able to see user-specific analytics (How old is the public watching the program XYZ ? Which gender ? Which location ? so forth...)
analyticsId	String	The main tracking code that has been provided to you by your Streamhub integration engineer.
userAgent	String	Hardcoded user-agent string used to attribute the views coming from this device.

Set the streamhub instance as your player's event delegate

After your instance creation, the next and final step is to assign that streamhub instance as your ThePlatform player event's delegate as in;

```
[self.player addDelegate: self.streamhub];
```

self.player designate an TPPlayerView * instance.
self.streamhub designate the StreamhubAnalyticsTPBridge * instance you just created.

By doing that, your ThePlatform player will be fully configured and ready to send events to our plugin for tracking.