Player Side REST API

15th of December 2016	1.0	Removed Ad tracking additional parameter. We follow the standard VAST 4.0 spec.
30th of January 2017	2.0	Updated events table with QoS and metadata tracking. Added player events flow diagram.
24th of April 2017	3.0	Added a section on ‘implementation-free optional parameters’
30th of June 2017	3.1	Updated Event flow diagram section.
18th of January 2018	3.2	Changed `buffering` and `rebuffering` event names to `prebuffering` and `buffering` (Reporting events section)
6th of March	3.3	Additional examples for the addMediaMetadata API
26th of June	3.4	Updated media_metadata section with regards to how categories shoud be provided.

Overview

The Streamhub REST API provides a RESTful service for sending video usage reporting data to the Streamhub analytics platform.

Even though it is possible to report statistics by requesting the HTTP REST service directly, the effort to be compliant to the service methods would be significant.

Instead, we highly recommand to create a bridge plugin using our javascript library.

See Creating your own plugin for the Streamhub Analytics platform

API Endpoints and HTTP requests

Four categories of HTTP requests can be submitted to the Streamhub collector API: Ticks (api/player), Events (api/playerevent), Content (api/content) and Advertisements (api/advertisement).

note: Content and Advertisements apis are under development and not detailed in this document.

Sending ticks

API endpoint: //stats.streamhub.io/api/player

Ticks are the first class data that we collect and compute media views from.

The logic to report ticks to the collector does not require to know the media content duration.

Ticks should be sent to the collector:

- Every 5 seconds (0sec, 5sec, 10sec…) during the first minute of video playback

- Every minute after the first minute until the end of the playback

For example, if a video is 2 minutes long, 14 ticks should be sent at 0 seconds, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60 and 120 seconds.

Parameters

Property name	Required	Short description	Example values
startTime	X	current video playhead's position rounded downward to its nearest integer and divided by 60.	0 (0 seconds) 0.08333333333333333 (5 seconds / 60)
publicId	X	current video unique identifier in the OVP system.	4116821249001 (Brightcove typical video ID. Uses 13 digits identifier)
partnerId	X	identifier of the OVP partner. Provided to you by your Streamhub account manager	*Provided to you by your Streamhub account manager.*
analyticsId	X	A unique identifier in our system for each combination of a content owner and a partner.	*Provided to you by your Streamhub account manager.*
playerId	X	A unique identifier for a video player across your ecosystem of websites and application.	4355821241001 (Brightcove typical player ID. Uses 13 digits identifier). *Has to be generated by us or communicated to us.* **Important: player ids not communicated to us, will result in unprocessed views** (ticks will be stored and can be processed at a later time though)
isLive	X	indicates if the media currenlty played is a Vod content or a Live event	true false
randomSessionKey	X	UUID randomly generated for a particular video view session.	26A83674-7EEC-A662-A80C-71A7A3C3A553

sessionId	X	Used to identify a unique user (same computer, same web browser or application container). Should be created at the player plugin level, stored in a cookie and re-used.	8F4F55FA-CA09-B828-B23C-6DCA85EF4094
agent			Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.73 Safari/537.36&os=5.0 (Macintosh; Intel Mac OS X 10_10_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.73 Safari/537.36
bitrate		Integer value. Current video bitrate at the time the tick is sent.	1000 Expressed in kbps
refUrl		Used to compute referral traffic. Usually obtained with document.referrer.	http://www.google.com
locationUrl		Location of the video player	http://www.jwplayer.com/
userId		Should be ignored for now	

HTTP Request sample

http://stats.streamhub.io/api/player/?startTime=0.08333333333333333&publicId=unboxing6s&partnerId=demos&analyticsId=demos&playerId=player-demos&isLive=false&refUrl=&locationUrl=http://localhost/jw/index.html?html5&agent=Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.73 Safari/537.36&os=5.0 (Macintosh; Intel Mac OS X 10_10_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.73 Safari/537.36&bitrate=0&randomSessionKey=26A83674-7EEC-A662-A80C-71A7A3C3A553&sessionId=8F4F55FA-CA09-B828-B23C-6DCA85EF4094&userId=UNKNOWN

Reporting events

API endpoint: //stats.streamhub.io/api/playerevent

The playerevent api allows reporting of user interaction and media playback events, such as a user clicking on a pause button or a media reaching playback completion.

Parameters

Additionally to the property fields available to report ticks, the playerevent api accepts a property **event** which can be completed by one or more property providing details for the event.

Property name	Required	Short description	Comments	Additional parameters
media_metadata		Should be sent as soon as metadata are available		title, duration. See reporting metadata section.
media_loaded		Should be the first tracking event sent by the player. The event is triggered by the first network request to download a video data segment.		
prebuffering		Account for the buffering period that occurs when the player downloads the first segment of video bytes data, prior to the first frame of the video is rendered in the player. The amount of buffering time tracked here is used to compute the average start time of the video over multiple sessions.	This event occurs **before media_loaded** and **player_start.**	bufferingTime
player_start	X	should be sent when a video playback starts for the first time. If there is a context of playlist where multiples videos are chained together, then a distinct **player_start** event should be sent for each videos in the playlist.		
buffering		Account for the buffering periods that occurs during playback when	This event occurs **after player_start.**	bufferingTime

		the video player buffer is empty and the playback is stalled.	The prebuffering and buffering events will necessarily be sent at the end of the buffering periods as they need to indicate how long the buffering took.	
player_play_completed	X	Should be sent at the end of a video playback (for each videos in a playlist)		
completion		see reporting completion rate section below	event=completion&completionRate=1	completionRate. see reporting completion rate section for details.

HTTP Request sample

http://stats.streamhub.io/api/playerevent/?startTime=0&publicId=unboxing6s&partnerId=demos&analyticsId=demos&playerId=player-demos&isLive=false&refUrl=&locationUrl=http://localhost/jw/index.html?html5&agent=Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.73 Safari/537.36&os=5.0 (Macintosh; Intel Mac OS X 10_10_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.73 Safari/537.36&bitrate=0&randomSessionKey=949BC36D-C275-359D-E8D8-72142090F7AD&sessionId=8F4F55FA-CA09-B828-B23C-6DCA85EF4094&userId=UNKNOWN**&event=click_pause_off**

Reporting completion rate events

The **completion** event enables reporting of the completion rate which corresponds to how much of a video has been really watched by the user.

The **completionRate** parameter in the event request (see example below) holds the actual completion rate value which can be one of the following: 1, 25, 50, 75 and 95 respectively for watching 1%, 25%, 50%, 75% and 95% of a video.

Technical notes

To evaluate if a completion rate should be reported, it is necessary to know the video duration.

A convenient way to make the comparison between the playhead's position value (expected in seconds) and the completion stop points (see below) is to round them both downward to their nearest integer and then make the comparison.

Example

A video is 68.9 seconds long.
Completion stop points happening at 1%, 25%..95% of the video duration, occur at 1, 17, 34, 51 and 65 seconds.
A playhead event is fired by the video player at 1.122 seconds. Then, the **completion** event should be reported as this:

http://stats.streamhub.io/api/playerevent/?startTime=0&publicId=...**&event=completion&completionRate=1**

Another playhead event is fired by the video player at 17.233 seconds. Then, the **completion** event should be reported as this:

http://stats.streamhub.io/api/playerevent/?startTime=0.25&publicId=...**&event=completion&completionRate=25**

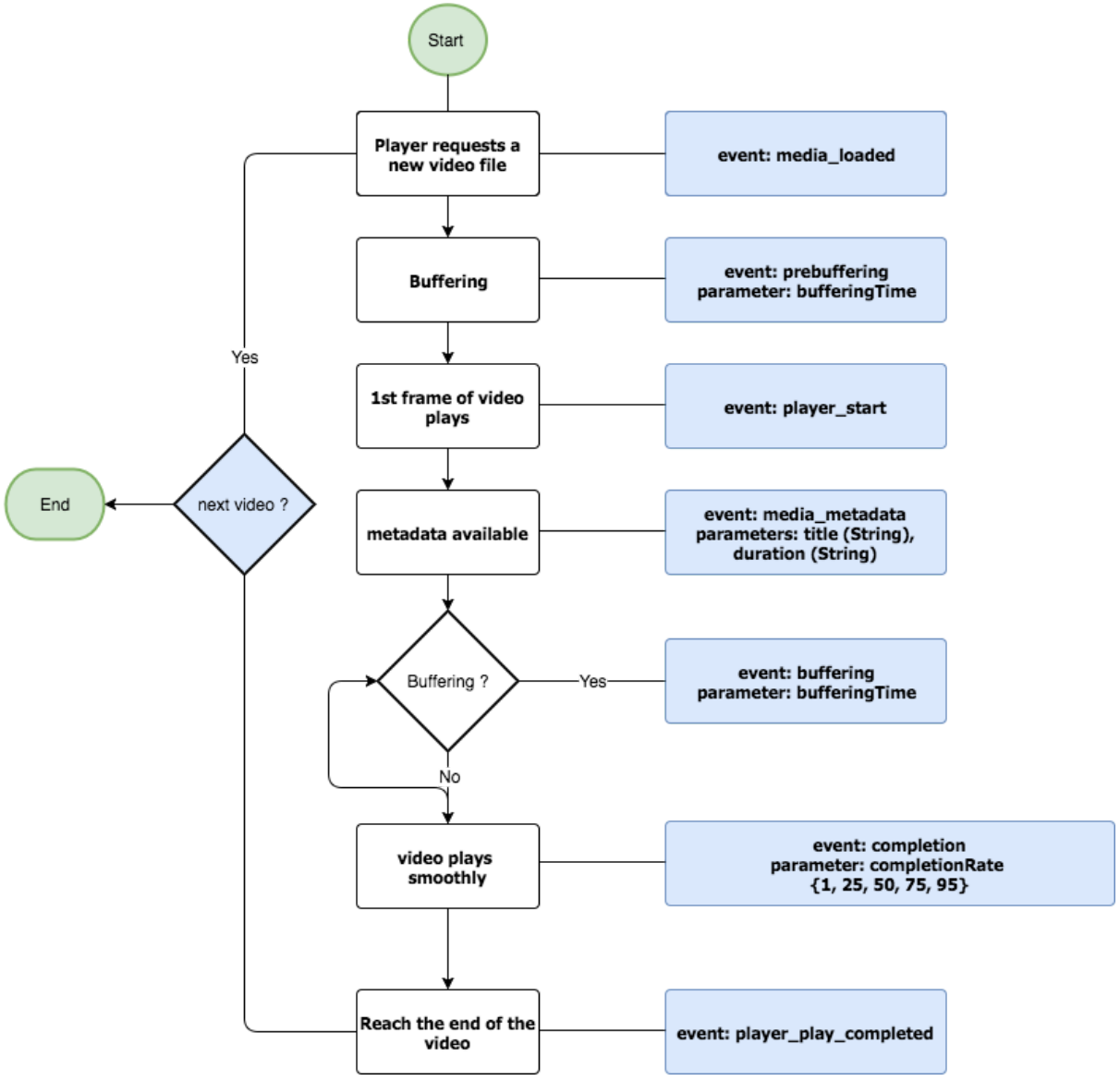
Reporting media medata events

The **media_metadata** event enables Front Loading of video metadata at the player level. We currently support 4 additional metadata properties that you can send with the **media_metadata** event: title, playerTitle, duration, categories.

Property name	Required	Type	Comments	Example
duration	yes	String	Must be provided in seconds as an **Integer**. Floating point value would be **rejected** by our collector.	"355"
playerTitle	yes	String	Need to be url encoded	"My%20awesome%20player"
title	yes	String	Need to be url encoded	"Screen%20Recording"
categories	no	String	Separated by a pipe | character	"some|meaningful|categories"

			provide d as comma delimited string of url-encoded categories.	

Player events flow diagram



Tracking Ads

API endpoint: //stats.streamhub.io/api/advertisement

Generally speaking, the ad tracking reporting is following the standard VAST specification (http://www.iab.com/wp-content/uploads/2016/04/VAST4.0_Updated_April_2016.pdf)
Not all the events available in the specification are available via our API though.

If you need to track Ads in your video player to get Ad specific reports from Streamhub, you can send 2 types of events to the **api/advertisement** endpoint to report ;

1. the viewing of ads in your video player (completion)
2. user engagement with the ad asset (**click**)

The **completion** event should be sent every time a 25% of the ad has been watched, e.g. at 0%, 25%, 50%, 75% and 100%.
The advertisement API accept a **percentile** parameter to hold the percentage watched.

The **click** event should be sent whenever the user clicks on the advertisement in the player.

Examples of ad tracking events are provided below.

Ad identifiers and video identifier

When you send a Ad specific event, the **publicId** parameter holds the ad id unique identifier which comes from the VAST xml response (e.g <UniversalAdId> element)
Please refer to:

- VAST (https://www.iab.com/guidelines/digital-video-ad-serving-template-vast-4-0/)
- AD ID (http://www.ad-id.org/).

Property	**Comments**
publicId	Equivalent of Ad ID. Required, String

The ad asset being identified, you still need to provide the video identifier (main content) for which you are tracking ads.
Use the **parentPublicId** parameter to provide the video identifier.

The table below summarizes the remaining parameters that you must send along with Ad tracking events. These are a subset of the parameters you have to send along with tick events.

Property name	**Required**	**Short description**	**Example values**
partnerId	X	identifier of the OVP partner. Provided to you by your Streamhub account manager	*Provided to you by your Streamhub account manager.*
analyticsId	X	A unique identifier in our system	*Provided to you by your Streamhub*

		for each combination of a content owner and a partner.	*account manager.*
playerId	X	A unique identifier for a video player across your ecosystem of websites and application.	4355821241001 (Brightcove typical player ID. Uses 13 digits identifier). *Has to be generated by us or communicated to us.* **Important: player ids not communicated to us, will result in unprocessed views** (ticks will be stored and can be processed at a later time though)
randomSessionKey	X	UUID randomly generated for a particular video view session.	26A83674-7EEC-A662-A80C-71A7A3C3A553
sessionId	X	Used to identify a unique user (same computer, same web browser or application container). Should be created at the player plugin level, stored in a cookie and re-used.	8F4F55FA-CA09-B828-B23C-6DCA85EF4094
agent			Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.73 Safari/537.36&os=5.0 (Macintosh; Intel Mac OS X 10_10_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/47.0.2526.73 Safari/537.36
refUrl		Used to compute referral traffic. Usually obtained with document.referrer.	http://www.google.com
locationUrl		Location of the video player	http://www.jwplayer.com/

Example scenario

Your video player is playing a preroll ad before the main video content.
The parameters are configured as:

- ➔ The ad has a unique identifier `1E2DFA89-496A-47FD-9941-DF1FC4E6484A`
- ➔ The main video content has a unique identifier based on 13 digits (like the Brightcove platform) 1996784390001
- ➔ The user has successively watched 0% (necessarily) and 25% of the ad
- ➔ The user click on the advertisement just after 25%

Ignoring the other parameters which are not specific to ad tracking, here are the 3 HTTP requests that should be send to the advertisement API endpoint ;

http://stats.streamhub.io/api/advertisement/?publicId=1E2DFA89-496A-47FD-9941-DF1FC4E6484A&parentPublicId=1996784390001&sessionId=501F9C1D-B7E0-D656-5509-5D7F740E1585&randomSessionKey=7EF7C2B4-40B3-8267-C34A-5D856AE2CE5B&refUrl=&agent=Mozilla/5.0%20(Macintosh;%20Intel%20Mac%20OS%20X%2010_10_5)%20AppleWebKit/537.36%20(KHTML,%20like%20Gecko)%20Chrome/51.0.2704.84%20Safari/537.36&partnerId=jwplatform&analyticsId=streamhub-5812d&playerId=player-demo&percentile=0&event=completion

http://stats.streamhub.io/api/advertisement/?publicId=1E2DFA89-496A-47FD-9941-DF1FC4E6484A&parentPublicId=1996784390001&sessionId=501F9C1D-B7E0-D656-5509-5D7F740E1585&randomSessionKey=7EF7C2B4-40B3-8267-C34A-5D856AE2CE5B&refUrl=&agent=Mozilla/5.0%20(Macintosh;%20Intel%20Mac%20OS%20X%2010_10_5)%20AppleWebKit/537.36%20(KHTML,%20like%20Gecko)%20Chrome/51.0.2704.84%20Safari/537.36&partnerId=jwplatform&analyticsId=streamhub-5812d&playerId=player-demo&percentile=25&event=completion

http://stats.streamhub.io/api/advertisement/?publicId=1E2DFA89-496A-47FD-9941-DF1FC4E6484A&parentPublicId=1996784390001&sessionId=501F9C1D-B7E0-D656-5509-5D7F740E1585&randomSessionKey=7EF7C2B4-40B3-8267-C34A-5D856AE2CE5B&refUrl=&agent=Mozilla/5.0%20(Macintosh;%20Intel%20Mac%20OS%20X%2010_10_5)%20AppleWebKit/537.36%20(KHTML,%20like%20Gecko)%20Chrome/51.0.2704.84%20Safari/537.36&partnerId=jwplatform&analyticsId=streamhub-5812d&playerId=player-demo&percentile=0&event=click

Implementation-free Optional Parameters

We provide 2 optional parameters, which you can use to send additional and custom informations to our collector API.
This mechanism provides a layer of flexibility and customisation in the integration phase.
These parameters must sent as **opt1** and **opt2** url parameters.
For instance, you could use them to send additional metadata about a program (like additional program's identifier, program's title, duration, category, tag...) that you want us to ingest or process in a particular way.

If you choose to use this facility, make sure you send those optional parameters with all the Http requests that you send to the collector (player/, playerevent/ and advertisement/ endpoints).

Example of a query to the player endpoints including opt1 and opt2 parameters:
http://stats.streamhub.io/api/player/?publicId=1E2DFA89-496A-47FD-9941-DF1FC4E6484A&parentPublicId=1996784390001&sessionId=501F9C1D-B7E0-D656-5509-5D7F740E1585&randomSessionKey=7EF7C2B4-40B3-8267-C34A-5D856AE2CE5B&refUrl=&agent=Mozilla/5.0%20(Macintosh;%20Intel%20Mac%20OS%20X%2010_10_5)%20AppleWebKit/537.36%20(KHTML,%20like%20Gecko)%20Chrome/51.0.2704.84%20Safari/537.36&partnerId=jwplatform&analyticsId=streamhub-5812d&playerId=player-demo&opt1=ABC&opt2=42

Please contact your integration engineer to use this feature.