

Fraunhofer FOKUS Institute for Open Communication Systems

MSE based media playback on HbbTV using dash.js

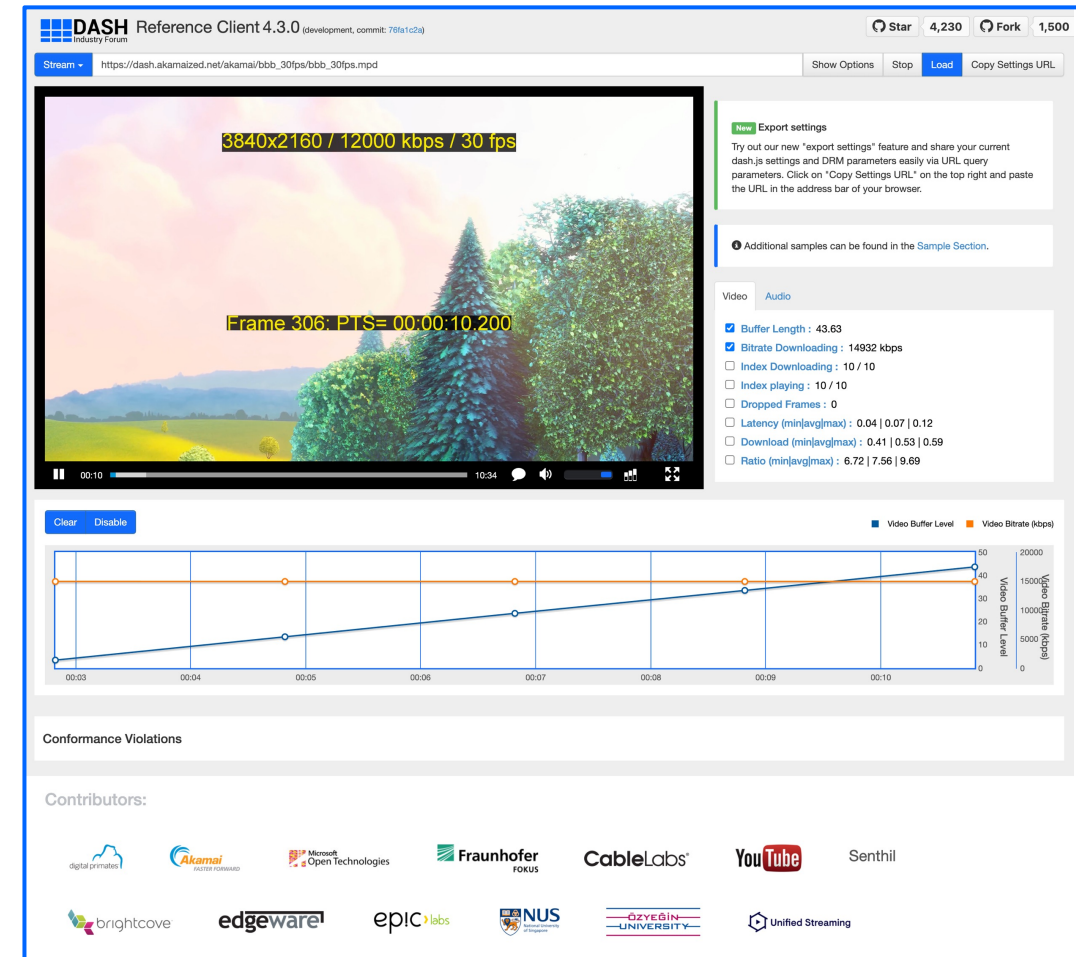
Daniel Silhavy, Louay Bassbouss

01

dash.js development and latest features

dash.js – Overview & Status

- Official [reference player by the DASH Industry Forum](#) for playback of MPEG-DASH content
- Maintained by Fraunhofer FOKUS, community driven development
- Open-source project on Github - <https://github.com/Dash-Industry-Forum/dash.js/> , last released version 4.4.0
- Used as a reference player, in production (Industry) and for academia and research purposes
- Written in JavaScript uses the W3C Media Source Extensions (MSE) and Encrypted Media Extensions (EME)
- Works on all MSE and EME based platforms including Desktop browsers, smartphones, SmartTVs, Set-Top Boxes.
- Various features including flexible ABR logic, multiperiod, DRM support, MPD patching, Gap handling, CMCD, CMAF low latency support, support for various subtitle formats (TTML, IMSC1, WebVTT) and many more.



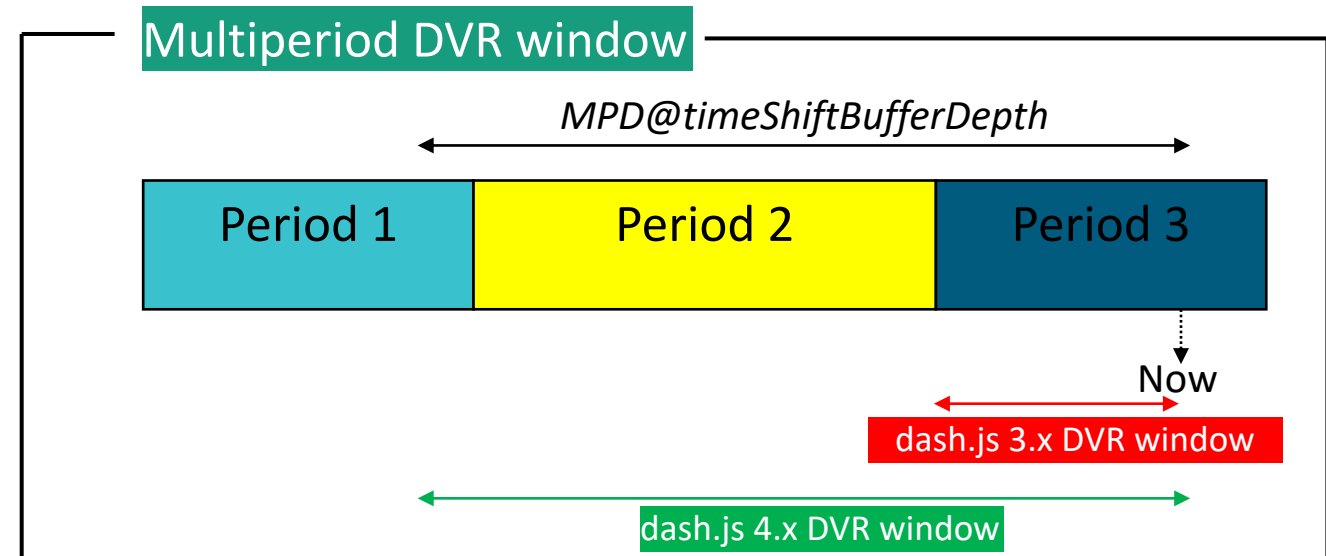
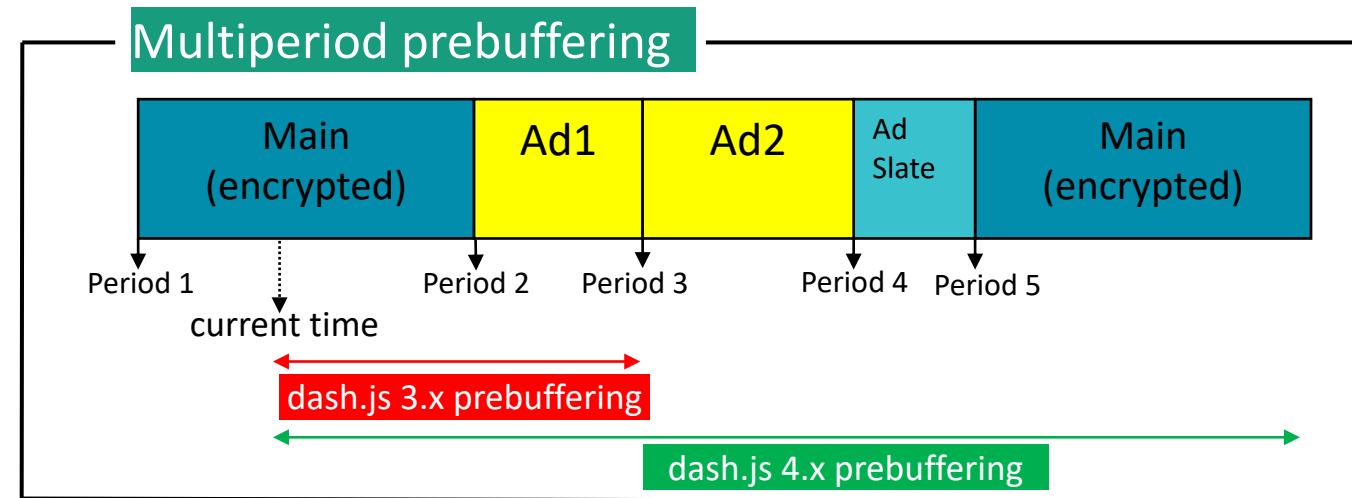
<https://reference.dashif.org/dash.js/nightly/samples/dash-if-reference-player/index.html>

02

Multi-Period DASH and DRM in dash.js

dash.js – Multiperiod playback

- Multiperiod playback enables use cases such as server-side ad-insertion and transition between encrypted and non-encrypted content
- dash.js 4.x supports
 - Prebuffering of multiple upcoming periods
 - DVR window overlapping multiple periods
 - Support for transition between encrypted and non encrypted periods
 - Gap handling: MSE implementations stall if the buffer is not continuous. Reasons include
 - Unaligned Periods
 - Sample duration does not match segment duration
 - Negative/Positive *@eptDelta* or negative *@pdDelta*
 - Related blog post: <https://tinyurl.com/eptdelta>
- Multiperiod Example: <https://tinyurl.com/mp-dashjs>
- More details: <https://tinyurl.com/dashjspaper>



dash.js – Digital Rights Management (DRM)


- dash.js supports various configurations and settings related to DRM playback via simple API calls and predefined settings:
 - Support for [Playready](#), [Widevine](#) and (Clearkey)
 - Support for [multiple EME versions](#)
 - 0.1b – initial implementation of the EME, non-promised based uses prefixed events like “webkitneedkey”
 - 2014 – EME version state 2014: Implemented by Internet Explorer 11 (Windows 8.1).
 - 2015 - most recent EME implementation. Latest changes in the EME specification are added to this model
 - License server URLs via API and MPD
 - DRM system priority
 - In case multiple DRM systems are supported on the underlying platform
 - Key system string priority
 - For instance, use “*com.microsoft.playready.recommendation*” prior to using “*com.microsoft.playready*”
 - DRM specific headers
 - Add custom headers to your license request
 - Define robustness levels to be used to enable Hardware DRM
 - For instance, use “HW_SECURE_ALL” for L1 Widevine
 - Define promise-based callback functions to modify license request and license response
 - Preserve MediaKeys and MediaKeySessions during MediaPlayer lifetime to avoid new license requests
 - See DRM sample section and documentation:
 - <https://reference.dashif.org/dash.js/nightly/samples/index.html>
 - [https://github.com/Dash-Industry-Forum/dash.js/wiki/Digital-Rights-Management-\(DRM\)-and-license-acquisition](https://github.com/Dash-Industry-Forum/dash.js/wiki/Digital-Rights-Management-(DRM)-and-license-acquisition)

03

dash.js on HbbTV

MSE Support in HbbTV

- Native DASH Playback is supported since early versions of HbbTV, the **Media Source Extensions (MSE)** support was introduced in the last published **HbbTV 2.0.3** Version (as required by WMAS 2018).
- MSE works in combination with the HTML5 media element either via **srcObject** attribute if supported or via URL pointing to **MediaSource**
- **HbbTV 2.0.3** specification adds requirement for playback of **low-latency DASH** content using JavaScript DASH players. This must be tested via **unmodified releases** of widely deployed JavaScript **DASH players** (such as **dash.js**).
- Not only **HbbTV 2.0.3** specification requires MSE, but also **HbbTV Targeted Advertising (TA)** Specification requires it. HbbTV apps will have a full control on the **buffering** behavior of ad assets.
- Even MSE support was introduced in HbbTV 2.0.3, many terminals **with older HbbTV versions already support MSE**.

HbbTV +  dash.js

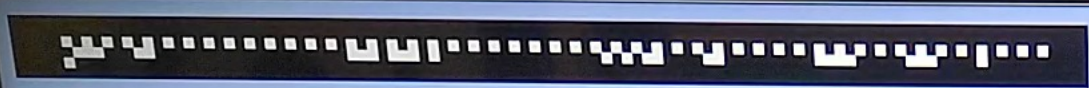


dash.js meets HbbTV at Fraunhofer FOKUS TV Lab

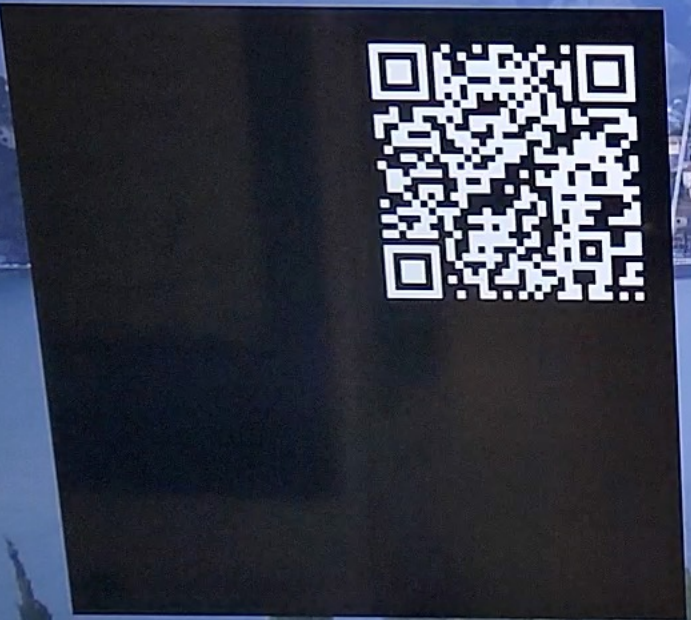
MSE & dash.js performance tests on HbbTV

TV	HbbTV Version	MSE	dash.js LL	Comment
Manufacturer1	HbbTV1.2.1	Supported	Supported	
Manufacturer2	HbbTV1.5.1	Supported	Supported	
Manufacturer3	HbbTV1.4.1	Supported	Supported	
Manufacturer2	HbbTV1.4.1	Supported	Supported	
Manufacturer4	HbbTV1.5.1	Supported	Supported	
Manufacturer5	HbbTV1.2.1	Supported but...	Supported but...	...QuotaExceededError is raised
Manufacturer1	HbbTV1.4.1	Supported but...	Supported but...	...Lot of artifacts for entire playback
Manufacturer2	HbbTV1.2.1	Supported	Supported	
Manufacturer4	HbbTV1.1.1	Not Supported	N/A	
Manufacturer5	HbbTV1.5.1	Supported	Supported	
Manufacturer6	HbbTV1.5.1	Supported	Supported	

<https://www.fokus.fraunhofer.de/en/fame/laboratories/smartmedia>



L1



00:00:03.960;0000099;25

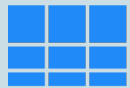


WAVE

WEB APPLICATION VIDEO ECOSYSTEM

<https://github.com/cta-wave/Test-Content>

<https://github.com/cta-wave/mezzanine>



dash.js

<https://github.com/Dash-Industry-Forum/dash.js>

(v.4.4.1)

HbbTV

<https://hbbtv.org>

HbbTV 2.0.2 Terminal

Contact

Daniel Silhavy
daniel.silhavy@fokus.fraunhofer.de



Louay Bassbouss
louay.bassbouss@fokus.fraunhofer.de



Fraunhofer FOKUS
Institute for Open Communication Systems
Kaiserin-Augusta-Allee 31
10589 Berlin, Germany
www.fokus.fraunhofer.de
<https://websites.fraunhofer.de/video-dev/>

