dash.js - Overview and release planing

DASH-IF Call – 22.07.21

Daniel Silhavy - Fraunhofer FOKUS

Agenda

1. dash.js 4.0 (max 20 minutes)
2. dash.js release planing

dash.js v4.0.0

dash.js 4.0 – Release Information

- Release notes: https://github.com/Dash-Industry-Forum/dash.js/releases/tag/v4.0.0
- DASH-IF Hosted Player: https://reference.dashif.org/dash.js/v4.0.0/samples/dash-if-reference-player/index.html
- Samples Page: https://reference.dashif.org/dash.js/v4.0.0/samples/index.html
- Migration guide: https://github.com/Dash-Industry-Forum/dash.js/wiki/Migration-to-dash.js-4.0

DASH Industry Forum

dash.js 4.0 - Overview

- Released 25th June 2021
- Major improvements for multiperiod playback (stability, enable seeking between periods for multiperiod livestreams, improved gap handling, fix baseURL resolution, fix race conditions)
- Rework buffer and scheduling logic (e.g. DVR window and AST/AET calculation)
- Enable audio track switch with different codecs (different AdaptationSets)
- Rework sample Section and provide additional samples
- Low latency ABR improvements
  - Improvements for LoL+
  - AAST based approach for measurement and estimation of throughput calculation in LL CTE streaming
- Reworked text track handling
- Development improvements (replace Grunt with Webpack, replace JSHint with ESLint)
- Various bugfixes and improvements

Multiperiod Improvements – Timing model

- DVR window no longer limited to period boundaries. Enables seeking between periods in live streams
- Example: https://tinyurl.com/dashjsMp , five periods 60 seconds each, TSDB = 300sec

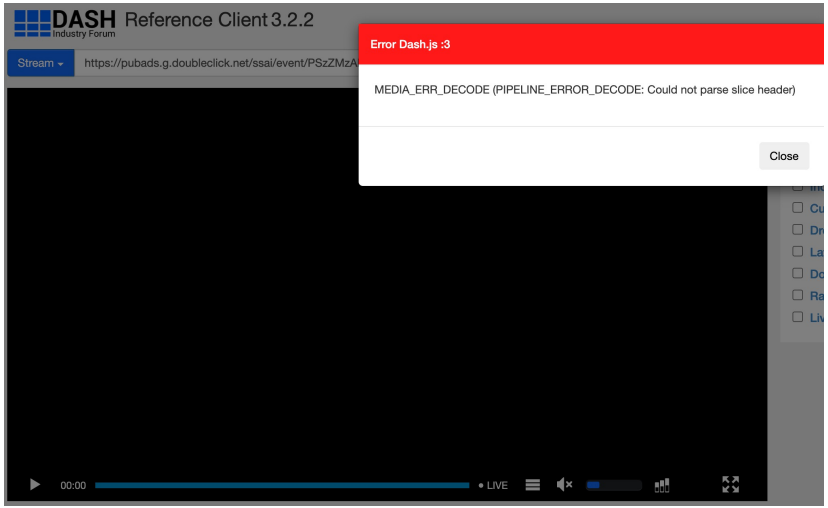
dash.js 3.2.2
Seeking limited to one period (60 sec)

dash.js 4.0.0
Seeking enabled for whole DVR window (300 sec), five periods

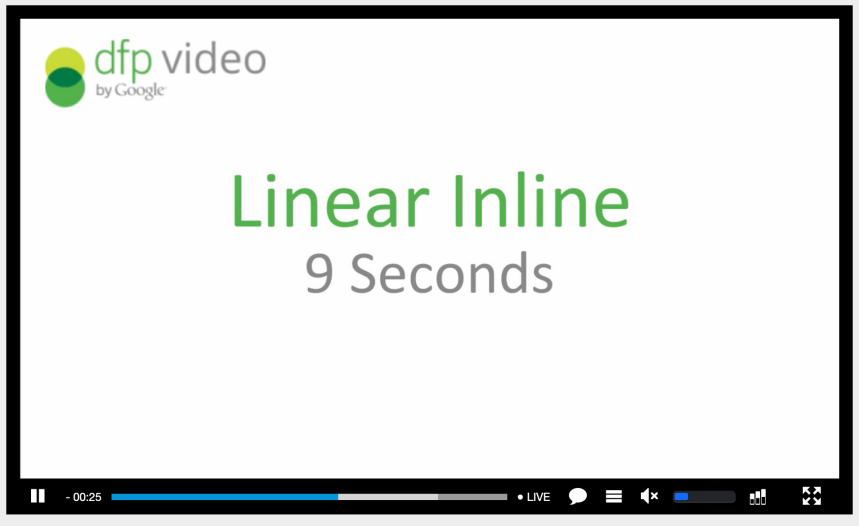
Multiperiod Improvements – Stability

- Fixed major race conditions due to parallel events
- MPD updates lead to inconsistencies in scheduling and BaseURL resolution
- Rewrote gap handling
- Introduced promises to make asynchronous code easier to maintain

dash.js 3.2.2



dash.js 4.0.0

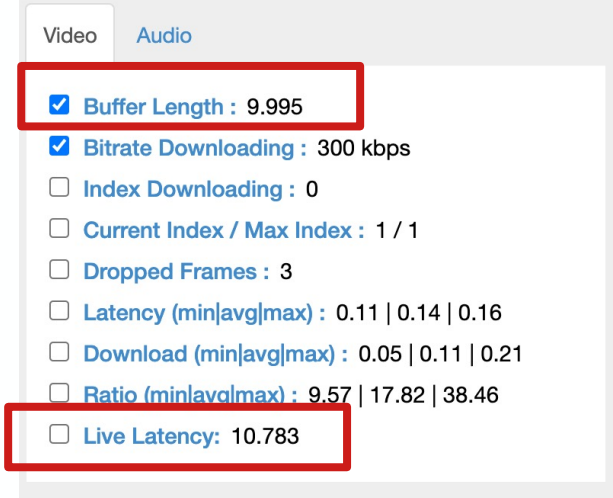


Timing/Scheduling Improvements - ATO

- Support for availabilityTimeOffset
- Example: https://tinyurl.com/dashjsAto - ATO set to 10seconds
- Discussion item: Correct? Useful for ad-insertion?

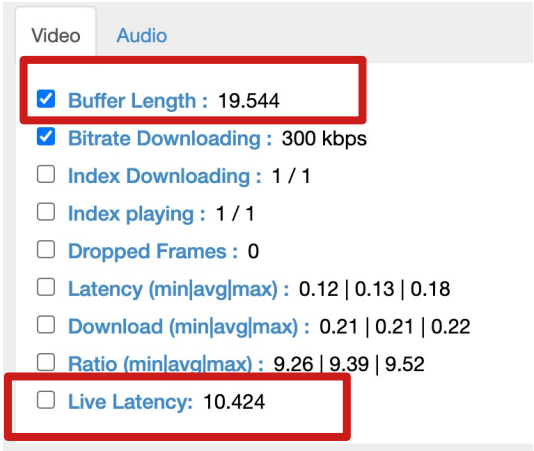
dash.js 3.2.2

Buffer level is always below live latency



dash.js 4.0.0

Buffer level can be higher than live latency if ATO is set accordingly

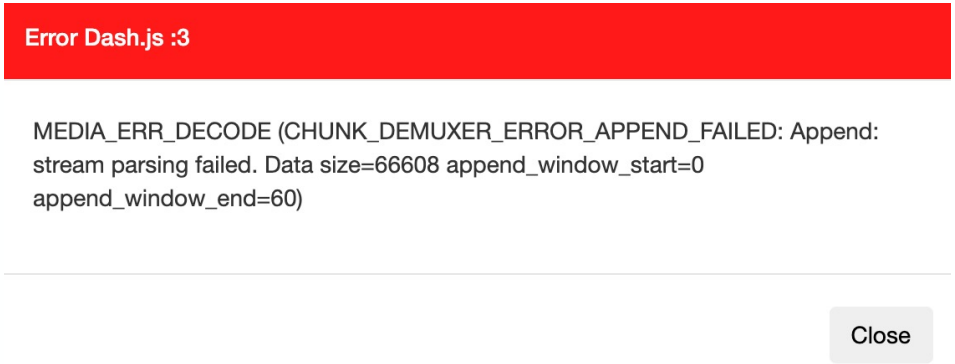


Multi Codec support – Audio/video track switch

- Enable audio/video track switch with different codecs (different AdaptationSets)
- Similar to period switches: MSE function changeType() is used
- Example: https://tinyurl.com/dashjs40 - Select Stream>DRM(modern)>Angel One

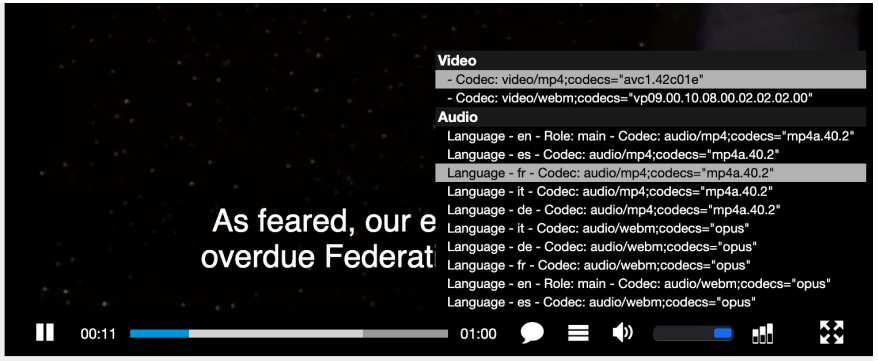
dash.js 3.2.2

Change of codec leads to MSE error



dash.js 4.0.0

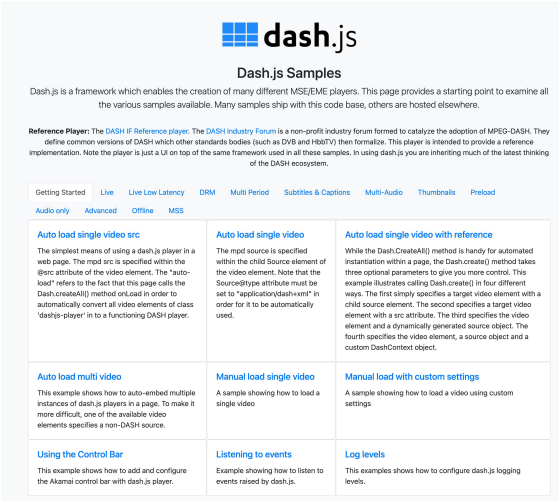
Change between avc/vp9 and aac/opus



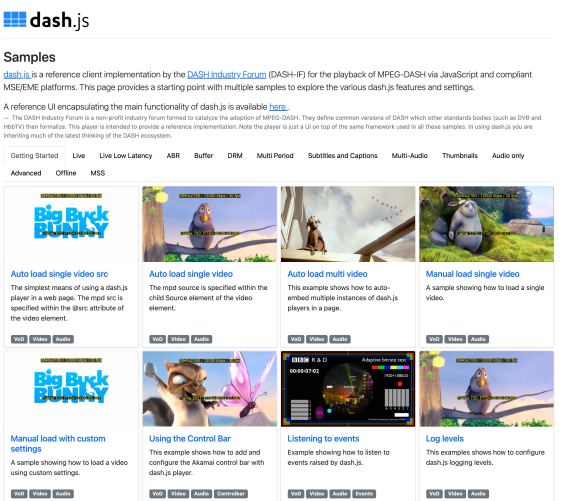
Reworked sample section

- Reworked/Redesigned existing samples
- Added various new samples related to buffer management, ABR handling, advanced features etc.
- Every sample includes sample code and a description
- Example: https://tinyurl.com/dashjs40Samples

dash.js 3.2.2

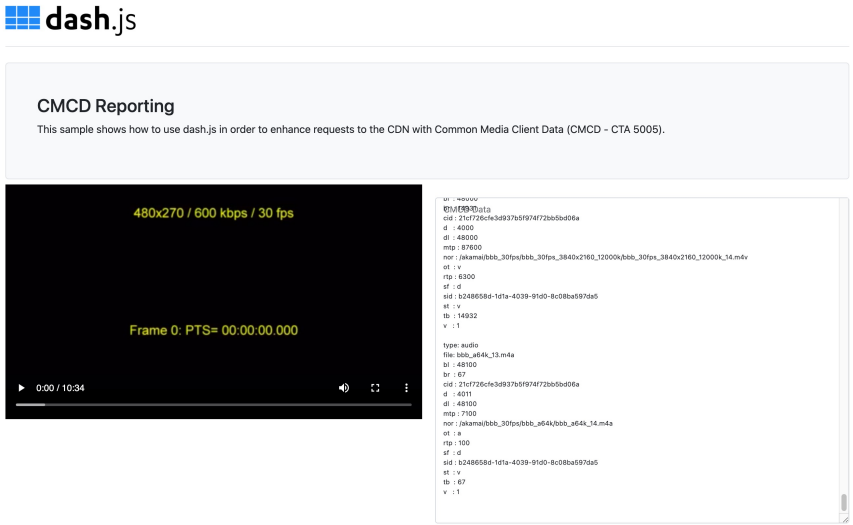


dash.js 4.0.0



Common Media Client Data

- Full support for CTA-5004 CMCD
- Report CMCD metrics like buffer level, bitrate, requested maximum throughput via query parameters or HTTP headers
- Example: https://tinyurl.com/dashjs40Cmcd



DASH Industry Forum

Improved developer setup & testing

- Replace grunt with Webpack
- Replace JSHint by ESLint
- Added additional unit and functional tests

dash.js – Release planing

Upcoming features / topics

- ABR & Throughput implementation
- Low Latency implementation
- Multiperiod
- API updates & General Improvements
- DRM improvements
- Ad-Insertion
- Xlink
- Supplemental & Essential Property
- Session based DASH (Watermarking)
- DASH + WebRTC
- Common Media Server Data (CMSD)

Note:

- After discussion in the DASH-IF IOP group some new items were added. These items are depicted in blue
- If test content is required, this is highlighted in red

DASH Industry Forum Testing

ABR & Throughput implementation

- Found issues with the current implementation.
  - Reacts slowly to bandwidth changes. One reason: Usage of arithmetic mean of latest four throughput samples.
  - Ideas: Harmonic mean, Assign weight on throughput samples, ML optimization problem
- Research on latest ABR algorithms including throughput calculation (ask Christian, Ali, others for recommendations).
- A testbed to test the various throughput and ABR combinations dash.js offers to identify the “best” combination. Something like https://github.com/cd-athena/CAdViSE or https://github.com/twitchtv/acm-mmsys-2020-grand-challenge . Potentially an open platform to compare.
- Additions to our functional test suite to check the ABR behavior.
- Better documentation on the throughput calculation and ABR algorithms in dash.js and how they interact.
- This is related to low latency activities.

DASH Industry Forum

Low-Latency implementation

- In general we should look at optimizations for LL. Most important aspect is stability.
- Support for Resync Representations - Currently quality change only possible at segment boundaries (reference streams required, completely defined in IOP5 and 23009-1 V5?)
- Improved low latency scheduling logic based on discussion in DASH-IF → Check with low latency specification first there might be edge cases.
- Improve ABR algorithms
  - L2A
  - Lol+
  - AAST based approach + PRFT
  - Requires sophisticated tests to figure out the best player configuration
- Add support for PRFT boxes
  - Encoder support(Elemental, Harmonic, Ateme)? Reference streams required
  - Dispatch to application
- Enable low latency based on MPD parameter: @availabilityTimeComplete = false

Multiperiod

- Further improve stability and buffer management → waiting for user feedback on potential problems
- Make use of MPD attributes like period connectivity and and period continuity. Potentially reference streams required
- Improve transition non encrypted and encrypted periods
  - Use initializationSet on MPD level to signal upcoming encrypted periods. Reference streams required
  - Check DASH profile for CMAF
- Improve buffer logic when the user seeks: Do not prune prebuffered segments.
- Thumbnail rendering limited to currently active period

API updates & general improvements

- Improve player robustness over
  - MSE errors
  - Segment download errors
- New reference UI: All settings should be accesible, Export and share settings
- MSE in webworkers & potentially additional MSE v2. features
- Improve settings handling. We need to distinguish between default settings, app settings and MPD settings.
- Improved dash.js XML parsing library
- Enhance livesim to support MPD patching (ask Zack). Reference streams required
- Update node modules
- Improve subtitle handling, current implementation is hard to maintain.
- Support for codec switch without changeType(). Reinit sourceBuffers when switching between audio/video AS.
- Decapsulate metric reporting from internal player metrics
- Improve/Enhance existing documentation. Might create a gh-page instead of putting everything in the Wiki.

DRM improvements

- Verify support for Enhanced Clear Key Content Protection (ECCP). Reference streams required
- Check for potential changes in the EME specification and what will be included in the next EME version
- Add support for
  - Different keys and different security levels in the Adaptation Sets. For instance HD uses a different key and security level than SD. We should not fail playback once one AdaptationSet fails to play. Reference streams required
  - Track filtering based on given output protection key status
  - HDCP check
  - PSSH naming as defined in MPEG DASH version 5. Reference streams required
- See also https://github.com/Dash-Industry-Forum/dash.js/discussions/3556

Ad-Insertion

- Align with DASH-IF ad-insertion guidelines
- Event reporting. We already support
  - MPD Reload events
  - Callback events
  - Dispatching of custom events
  - Anything else needed?
- MPD chaining or preroll element
  - Simple MPD chaining, pure playback of multiple playlists
  - Advanced MPD chaining: Optimize buffer usage
- Test content required, some is already in place: https://github.com/Dash-Industry-Forum/Test-Content/issues

DASH Industry Forum

XLink

- Dash.js currently supports Xlink:onload
- Add support for Xlink:onRequest
- Additional items?
- Testcontent required

Supplemental & Essential Property

- EssentialProperty: dash.js currently supports:
  - Filtering of unsupported EssentialProperty elements
  - Rendering of thumbnails
    - http://dashif.org/thumbnail_tile
    - http://dashif.org/guidelines/thumbnail_tile
- SupplementalProperty: dash.js currently supports:
  - DVB low latency property: urn:dvb:dash:lowlatency:critical:2019
- Additional items?

DASH Industry Forum

Session based DASH

- 23009-8 Session Based DASH
- Enables A/B watermarking

DASH + WebRTC

- New activity started in DASH-IF
- Integrating WebRTC Streaming into the DASH ecosystem.
- See: https://docs.google.com/document/d/1vwtJCgE95d2bPthwfKlvGxPiElgqJQZf7rtz_g_ZgZw/edit#

Common Media Server Data

- Specification is currently being drafted
- Consider doing an early adoption like with CMCD to increase visibility of dash.js
- Might also help to improve specific parts of the player like low latency throughput estimation.
- Future topic, specification will not be completed before Q4 2021. First implementation in Q1 2022

DASH Industry Forum

Testing

- Move to Karma based browser unit tests instead of running unit tests in node.js. Leads to more realistic results and browser APIs do not need to be polyfilled
- Add additional functional tests
  - ABR
  - Cover possible dash.js settings

Fraunhofer – Suggested development priorities

Major tasks

1. ABR / Throughput
   - Research
   - Testbed
   - Testing
   - Documentation
2. Multiperiod improvements
   - Bugfixing based on user feedback (continous task)
   - Buffer improvements
3. Low Latency
   - ABR/Throughput improvements (synergy to 1)
   - Improved low latency scheduling logic based on discussion in DASH-IF
   - PRFT support
4. DRM improvements
   - ECCP
   - (Different keys and different security levels)

General parallel activities

1. Improve player robustness over MSE and segment download errors
2. New reference UI: Export and share settings
3. Enable low latency based on MPD parameter
4. Improve settings handling (App, MPD, Default)
5. Improved dash.js XML parsing library
6. Update node modules

DASH Industry Forum

Development – Effort Estimation