W3C Media WG Meeting

TPAC 2024, 26 / 27 September 2024

Safety reminders

While attending Media WG meetings at TPAC, please follow the health rules:

- Masks are not mandatory, and left to individual choice
- COVID tests are available
- Please be aware of and respect the personal boundaries of your fellow participants
- If you feel unwell or have tested positive, please do not attend the meeting in-person

https://www.w3.org/2024/09/TPAC/health.html

Code of conduct

- Appreciate and accommodate our similarities and differences, be inclusive
- Have empathy when discussing sensitive issues
- Treat everyone with respect
- Be honest, be truthful
- Be aware of how much time is taken up
- Be sensitive to language differences
- Respect confidentiality and privacy

Refer to https://www.w3.org/policies/code-of-conduct/

Meeting logistics

- We use IRC for minute taking and to manage the speaker queue. Please join https://irc.w3.org/?channels=#mediawg
  - Type **`present+ Your_Name`** to record your attendance
  - Type **`q+`** to join the speaker queue
- Please volunteer to scribe
  - See IRC Guide: https://www.w3.org/wiki/IRC
- Remote participants, please mute unless you're speaking
- Please avoid side conversations in the meeting room (which get picked up by microphones)

Agenda

- Thursday 26 Sep 11:00 - 12:30
  - WG intro and current status
  - Media Source Extensions
- Thursday 26 Sep 14:00 - 16:00
  - Joint meeting with WebRTC
- Thursday 26 Sep 16:30 - 18:00
  - WebCodecs
- Friday 27 Sep 09:00 - 10:30
  - Audio Session, Media Session
- See https://github.com/w3c/media-wg/wiki/TPAC-2024

Media Working Group: Introduction

- Mission
  - The media foundations for the web, such as **HTMLMediaElement** and **Media Source Extensions** have helped turn the Web into a major platform for media streaming and media consumption. The Media Working Group will extend those foundations with new standardized technologies to improve the overall media playback experience on the Web
- Charter
  - 26 June 2023 to 31 May 2025 (initial charter: 22 May 2019)
- Co-chairs
  - Chris Needham – BBC chris.needham@bbc.co.uk
  - Marcos Caceres – Apple marcosc@apple.com
- Team Contact
  - François Daoust – W3C fd@w3.org

Publication status

Spec	Status	FPWD	Implementation status	CR
Audio Session	ED		Chromium | Gecko | WebKit	?
Autoplay Policy Detection	WD	2022-03	Chromium | Gecko | WebKit	?
Encrypted Media Extensions v2	WD	2024-07	Chromium | Gecko | WebKit	?
Media Capabilities	WD	2020-01	Chromium | Gecko | WebKit	?
Media Playback Quality	ED	SOON HTML	Chromium | Gecko | WebKit	?
Media Session	WD	2020-01	Chromium | Gecko | WebKit	?
Media Source Extensions v2	WD	2021-09	Chromium | Gecko | WebKit	?
Picture In Picture	WD	2020-01	Chromium | Gecko | WebKit	?
WebCodecs	WD	2021-04	Chromium | Gecko | WebKit	?

Picture in Picture

Open issues

- #99 Should PiP video removed from the DOM leave PiP? and whatwg/html#6271
- #179 Who is responsible for the user interface for PiP?
- #182 Should include / require spoofing protections
- #184 disablePictureInPicture interoperability
- #191 Remove monkey patch sections
- #208 How to determine the correct video size?

- Editorial help needed to move these forward

Horizontal reviews

Tracking issue	https://github.com/w3c/picture-in-picture/issues/219
Accessibility	Completed (review feedback: #179)
TAG	Completed (review: #226)
Internationalization	Not requested yet
Privacy	Not requested yet (self review). Add Privacy Considerations section
Security	Not requested yet (self review)

Autoplay Policy Detection

Horizontal reviews

Document status	WD: https://www.w3.org/TR/autoplay-detection/
Accessibility	Requested (self review, review #39) Awaiting response
TAG	Completed (review #810) See also previous review and sync/async question
Internationalization	Completed (self review, review #192)
Privacy	Completed (self review, review #111) Feedback needs addressing: #42, #43
Security	Requested (self review, review #48)

Open issues

- Privacy review feedback needs addressing
  - #42 Document fingerprinting and cross-site information leak risk
  - #43 Notifying the site of a browser intervention can be user harming
- Implementation is currently Gecko-only
- #23 Should the API result be binding on HTMLMediaElement.play()?
  - Do we need more developer feedback?

Media Playback Quality

Media Playback Quality

- Open issues
  - #19 Interaction with MediaStream
  - wicg/video-rvfc#92 Interaction with requestVideoFrameCallback
  - And more: https://github.com/w3c/media-playback-quality/issues
- Tests only cover WebIDL surface
  - https://wpt.fyi/results/media-playback-quality/idlharness.window.html
  - How to test droppedVideoFrames, corruptedVideoFrame
- Agreed to add to HTML spec (was originally part of MSE)
  - Propose to resolve issues first

Media Source Extensions

Horizontal reviews

Tracking issue	https://github.com/w3c/media-source/issues/356
Accessibility	Not requested yet (review feedback: #307)
TAG	Feature reviews: #298 (changeType), #576 (MSE for WebCodecs), #656 (MSE in Workers)
Internationalization	Not requested yet
Privacy	Not requested yet. See #261 Add Privacy Considerations section
Security	Not requested yet. See #261 Add Security Considerations section

Media Source Extensions: MSE in Worker

- WPT shows Chromium and WebKit support. Mozilla?
- Open issues:
  - #281 Track enable/selection change behavior needs clarity
  - #280 Expose {Audio,Video,Text}Track{,List} in DedicatedWorker
  - #278 Clarify window/worker coherency/communication of track instance creation/removal and track enabled/selected changes
  - #277 Consider (eventually) transitioning attached element to error upon termination of MediaSource's worker/what should media element do?
  - #276 Consider adding a "closing" readyState to explain new InvalidStateError exception when closing underway
  - #256 Compatibility with WebTransport

Media Source Extensions: Interop issues

- changeType tests results vary between engines - see WPT results: 1, 2, 3, 4, 5
- SourceBuffer.remove() - see WPT results
- SourceBuffer.abort() - see WPT results: 1, 2, 3, 4
- TrackDefault - see March 2015 MSE spec and WPT results: 1, 2
- Track changes and removal - see WPT results: 1, 2

Media Source Extensions: Managed MediaSource

- WPT test status?
- Chromium and Mozilla plans?
- Open issues:
  - #322 Add quality attribute

Media Source Extensions: Issues for discussion

- #357 Proposal: Have a detachable non-destructive MediaSource object

Joint Meeting: WebRTC Working Group and Media Working Group

Thursday 26 September, 14:00 - 16:00

Joint meeting: WebRTC WG and Media WG

- See meeting slides

WebCodecs, Encrypted Media Extensions, Media Capabilities

Thursday 26 September, 16:30 - 18:00

WebCodecs Horizontal reviews

Tracking issue	https://github.com/w3c/webcodecs/issues/784
Accessibility	Not requested yet. Needs self review
TAG	Completed (previous reviews: #433, #612)
Internationalization	Not requested yet. Needs self review
Privacy	Completed (self review)
Security	Completed? (review: #1)

WebCodecs: VideoFrame orientation

- Problem statement: video decoders and capturing pipelines sometimes produce video frames that need to be rotated before rendering. This rotation is usually expressed as extra data beside the frame. Currently Webcodecs API doesn't expose it, but developers ask for it on github and in emails.

- Images have the same issue. Let's get inspired by the CSS3 image-orientation syntax for explicit rotations.

- Transformation from the underlying media-resource to the intended rendering orientation is exposed as two new attributes on VideoFrame:
  - **rotation**: 0, 90, 180, or 270 (in degrees clockwise)
  - **flip**: false or true (whether horizontally mirrored after rotation)

More details w3c/webcodecs#351 VideoFrame orientation

WebCodecs: VideoFrame orientation details

1. **VideoFrameInit** and **VideoFrameBufferInit** will also need the **rotation** and **flip** attributes added to create a rotated frames from **CanvasImageSource** and buffers.

2. **VideoDecoderConfig** will have the same **rotation** and **flip** attributes for passing orientation info from containers.

3. **VideoFrame.copyTo()** and **VideoEncoder** will not take the orientation into account.

4. Rendering must take orientation into account.

More details w3c/webcodecs#351 VideoFrame orientation

WebCodecs: Resource exhaustion signal

- Problem statement: GPU driver only supports 5 video encoding sessions and the user tries to use 6 instances of VideoEncoder.

- We have a Resource Reclamation section in the spec that currently only allows throwing QuotaExceededError for codecs that weren't used in a while.

- Unfortunately this doesn't cover the most common scenario when we face an out-of-resource situation when we've just trying to configure or start using the codec.

- Solution: the spec needs to be changed to consider newly created codecs inactive until they:
  1. produced first output
  2. something else? how can we be sure that codec has all resources and started working.

More details w3c/webcodecs#669 Signaling codec resource exhaustion

Encrypted Media Extensions

Test cases

- We still have tests for persistent-usage-record, which we decided not to specify
  - See https://github.com/w3c/encrypted-media/issues/480
- Add more MediaSessionClosedReason tests?
  - Only closed-by-application is covered. Also release-acknowledged, hardware-context-reset, resource-evicted?
- getStatusForPolicy tests are failing in Gecko and WebKit, and for clearkey in all implementations
- WebKit ClearKey support?
  - https://bugs.webkit.org/show_bug.cgi?id=231006

Horizontal reviews

Tracking issue	https://github.com/w3c/encrypted-media/issues/558
Accessibility	Not requested yet. Needs self review
TAG	Feature reviews: #671, #322, #323
Internationalization	Not requested yet. Self review complete
Privacy	Not requested yet. Self review needs input: #550
Security	Not requested yet. Self review needs input: #550

Issues

- #132 EME should support continuous key rotation per MPEG Common Encryption (ISO/IEC 23001-7)
- #563 MPEG-CENC version support
- #251 Clear Lead support
- New EME features are out of charter scope, current charter ends 31 May 2025

Media Capabilities

Media Capabilities: PRs to review

- #222 Fixup MIME usage
- #223 Make keySystemAccess nullable in MediaCapabilitiesDecodingInfo
- #218 Clarify meaning of spatialScalability
- #217 Add comments to privacy section to address issue #176
- #212 RTC capabillity negotiation
- #186 Add a new webrtcCodec parameter to MediaCapabilitiesInfo
- #78 Define the meaning for CBR and VBR more precisely

Media Capabilities: Issues ready for PR

- #220 Throw type error when the configuration in MediaDecodingConfiguration doesn't match the one in keySystemConfiguration
- #203 Browser interop issues
- #197 Provide an example using MediaCapabilitiesKeySystemConfiguration encrypted
- #184 Meaning of MediaDecodingType "file" is unclear
- #152 How to reconcile colorGamut & transferFunction with ISO 23001-8:2016
- #146 *Configuration needs to be subclassed into *DecodingConfiguration
- #95 Frame rate configuration

Media Capabilities: Privacy discussion

- w3c/media-capabilities/#176 General approach to capability negotiation
  - What's the privacy story?

Horizontal reviews

Tracking issue	https://github.com/w3c/media-capabilities/issues/226
Accessibility	Completed (review feedback)
TAG	Completed (review #218)
Internationalization	Not requested yet
Privacy	Requested (feedback: #170, #171, #172, #173, #174, #175)
Security	Not requested yet

Audio Session and Media Session

Friday 27 September, 16:30 - 18:00

Audio Session

- w3c/audio-session#4 What is the scope and defaulting rules of an AudioSession?
- w3c/audio-session#5 Should we allow linking AudioSessions to each HTMLMediaElement/AudioContext object?
- w3c/audio-session#6 Should AudioSession be able to specify the output speaker and/or route options (a la sinkId)?
- w3c/audio-session#3 In case an AudioSession is explicitly set, should some incompatible APIs start failing?
- w3c/audio-session#1 Define spec MVP

Audio Session: quick recap

- Audio can be played in different ways for different purposes
  - Is the audio being played music, notification, driving navigations?
- Mixing of audio sources depends on the application
  - Audio sources of the same page, different pages, different applications
- AudioSession to the rescue!
  - `navigator.audioSession.type = ’playback’
  - `navigator.audioSession.onstatechange = () => { // check whether session is interrupted; … }

Audio Session: model & defaulting rules (1/2)

- Each document has a default AudioSession
  - Exposed as navigator.audioSession, default type is ‘auto’
  - Audio sinks/sources of the document are attached to this AudioSession
  - Default behaviour should match what happens today for web pages
    - May be UA specific

- Activation of AudioSession tied to sink/source activity
  - Starting an AudioContext activates its corresponding AudioSession

Audio Session: model & defaulting rules (2/2)

- AudioSession type computation
  - Use document hierarchy in case AudioSession type is ‘auto’?
  - Should we document the expected default computation for each sink/source, when it starts?
- Interruption mechanism between different AudioSessions of the same page
  - `auto` AudioSessions do not tend to interrupt themselves
  - Should two audio sessions of type playback from two documents of the same page interrupt each other?
  - How much the spec should say about this?

Audio Session: links to audio sinks/sources

- Audio sinks/sources are by default tied to their document AudioSession
  - Can change for HTMLMediaElement
- Should we allow web pages to group audio sinks/sources on an AudioSession?
  - Should we allow JS constructed AudioSession?
  - AudioSession setters/getters on each audio sink/source?
- Use-case: GPS application with music
  - AudioSession A for long media playback
  - AudioSession B (transient) for navigation sounds

Audio Session: specify output speaker

- Assumption: AudioSession is used to group audio sinks/sources
- AudioSession becomes a natural place to control speaker routes
  - A la setSinkId to set the speaker
  - Exposing further options
- Should we do this?

Audio Session: type error mechanism

- What happens when trying to start and audio sink/source with an incompatible AudioSession type?
  - Only example so far: getUserMedia requires `auto` or `play-and-record`
- Proposal
  - Capture start: reject promise
  - During capture: fail capture except if track is muted
- Potential edge case
  - Capture happens in an iframe, AudioSession.type is set by the top level context.

Audio Session: MVP

- Minimal scope
  - Existing API: type, state, onstatechange
  - Describe the overall underlying model to prepare the ground for new features
- Additional potential features
  - Request/release focus API
  - AudioSession grouping API
  - AudioSession route control API
- Priorities?

Media Session

- w3c/mediasession#340 Make “skipad” action more generic
- w3c/mediasession#338 Disabling default action handlers
- w3c/mediasession#237 The freezing in the artwork getter is broken
- w3c/mediasession#176 The "convert artwork algorithm" is underdefined

Media Session: default action handlers

- w3c/mediasession#338 Allow websites to disable default action handlers

- Some actions have default handlers provided by the user agent
- Allow websites to disable the default handler but without providing their own (current requires assigning an empty handler function)

Media Session: skipad action

- w3c/mediasession#340 Make "skipad" action more generic
- Some implementations do not currently support "skipad"
- Can we replace with a more generic "skip" action?
- Could be based on ChapterInformation metadata?
- See skipad picture in picture demo page

Media Session: Updating MediaMetadata

- w3c/mediasession#237 How do we expect MediaMetadata to be updated?

  ```
  navigator.mediaSession.metadata = new MediaMetadata({artist:"???", artwork:[{src:"A.png"}]})
  ```

  Should the following change the assigned metadata?

  ```
  navigator.mediaSession.metadata.artist = "Prince"
  navigator.mediaSession.metadata.artwork[0].src = "B.png"
  navigator.mediaSession.metadata.artist // Prince
  navigator.mediaSession.metadata.artwork[0].src // "http://example.com/A.png
  ```

- Related PR: w3c/mediasession#243 - should we continue with this, or revisit?
- **Proposal:** Update by assigning a new MediaMetadata with an updated dictionary
  - MediaImage should be an interface, and add a constructor for initialization
  - Use ImageResource? Reuse existing image fetch behavior

Media Session: Updating MediaMetadata

- Issues:
  - Web apps have no way to know if fetching the artwork succeeded or failed
  - Replacing chapter metadata seems awkward for live streams where chapters are added dynamically
  - Need a security model for loading resources: If browser, fetch and CORS. If handed to the OS, we need to define security rules, e.g., type restrictions

Horizontal reviews

Tracking issue	https://github.com/w3c/mediasession/issues/336
Accessibility	Not requested yet
TAG	Completed (reviews: #149, #608)
Internationalization	Not requested yet
Privacy	Not requested yet (Split Security and Privacy Considerations sections) Security and privacy self review (from 2017)
Security	Not requested yet (Split Security and Privacy Considerations sections) Security and privacy self review (from 2017)

Extra time?

MIME type handling

- Our specs used “valid media MIME type”
- Recent work to update using mimesniff **parse a MIME type** algorithm
  - Media Capabilities, Issue #69 (interop issues) and PR #222 (algorithm cleanup)
  - Encrypted Media Extensions, Issues #511, #512, and PR #559
- Implementation differences in MSE isTypeSupported: See WPT results
- We should also update MSE isTypeSupported() spec to use **parse a MIME type**

MIME type handling

- Media Capabilities and EME use mimesniff **supported by the user agent**:

> A MIME type is **supported by the user agent** if the user agent has the capability to interpret a resource of that MIME type and present it to the user.
>
> Ideally this would be more precise. See w3c/preload #113.

- MSE isTypeSupported has:

> 3. If *type* contains a media type or media subtype that the MediaSource does not support, then return false.
>
> 4. If *type* contains a codec that the MediaSource does not support, then return false.

MIME type handling

- HTML defines **a type that the user agent knows it cannot render**

4.8.11.3 MIME types §

A media resource can be described in terms of its *type*, specifically a MIME type, in some cases with a `codecs` parameter. (Whether the `codecs` parameter is allowed or not depends on the MIME type.) [RFC6381]

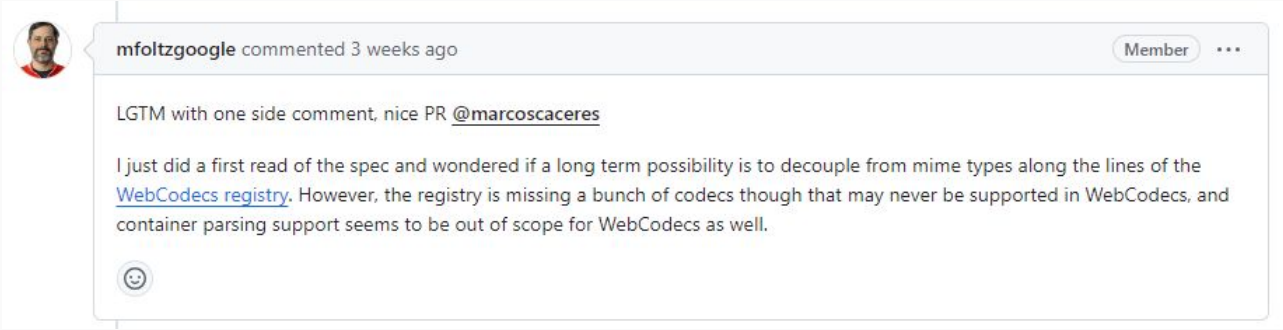
Types are usually somewhat incomplete descriptions; for example "`video/mpeg`" doesn't say anything except what the container type is, and even a type like "`video/mp4; codecs="avc1.42E01E, mp4a.40.2"`" doesn't include information like the actual bitrate (only the maximum bitrate). Thus, given a type, a user agent can often only know whether it *might* be able to play media of that type (with varying levels of confidence), or whether it definitely *cannot* play media of that type.

A type that the user agent knows it cannot render is one that describes a resource that the user agent definitely does not support, for example because it doesn't recognize the container type, or it doesn't support the listed codecs.

The MIME type "`application/octet-stream`" with no parameters is never a type that the user agent knows it cannot render. User agents must treat that type as equivalent to the lack of any explicit Content-Type metadata when it is used to label a potential media resource.

MIME type handling

- Can we be more precise than **supported by the user agent**?



- Create a registry? What would be its scope? Media playback, MSE, <audio> and <video>? WebCodecs? Encoding? RTC?

Thank you!