W3C WebRTC WG Meeting

September 20, 2021
08:00 - 10:00 Pacific Time

Chairs: Bernard Aboba
Harald Alvestrand
Jan-Ivar Bruaroey

W3C WG IPR Policy

- This group abides by the W3C Patent Policy https://www.w3.org/Consortium/Patent-Policy/
- Only people and companies listed at https://www.w3.org/2004/01/pp-impl/47318/status are allowed to make substantive contributions to the WebRTC specs

Welcome!

- Welcome to the September 2021 interim meeting of the W3C WebRTC WG, at which we will cover:
  - Conditional Focus, getViewportMedia, Display Surface Constraints and Echo Cancellation

Upcoming Meetings

- October Virtual Interim
  - Focus on mediacapture-transform
  - Slides
  - Doodle Poll for week of 4 October or 11 October (closes Sept. 28)
- TPAC meeting schedule posted:
  - https://www.w3.org/wiki/TPAC/2021/GroupMeetings#WG.2FIG.2FBG_Group_Meetings_details
- Joint Meeting with MEDIA WG (2 hours)
  - 26 October 2021, 14:00 UTC (7:00 AM Pacific)
- WEBRTC WG Meeting (1 hour)
  - 29 October 2021, 15:00 UTC (8:00 AM Pacific)
- For details, see: https://www.w3.org/2011/04/webrtc/wiki/Main_Page

About this Virtual Meeting

- Meeting info:
  - https://www.w3.org/2011/04/webrtc/wiki/September_20_2021
- Link to latest drafts:
  - https://w3c.github.io/mediacapture-main/
  - https://w3c.github.io/mediacapture-extensions/
  - https://w3c.github.io/mediacapture-image/
  - https://w3c.github.io/mediacapture-output/
  - https://w3c.github.io/mediacapture-screen-share/
  - https://w3c.github.io/mediacapture-record/
  - https://w3c.github.io/webrtc-pc/
  - https://w3c.github.io/webrtc-extensions/
  - https://w3c.github.io/webrtc-stats/
  - https://w3c.github.io/mst-content-hint/
  - https://w3c.github.io/webrtc-priority/
  - https://w3c.github.io/webrtc-nv-use-cases/
  - https://github.com/w3c/webrtc-encoded-transform
  - https://github.com/w3c/mediacapture-transform
  - https://github.com/w3c/webrtc-svc
  - https://github.com/w3c/webrtc-ice
- Link to Slides has been published on WG wiki
- Scribe? IRC http://irc.w3.org/ Channel: #webrtc
- The meeting is being recorded. The recording will be public.
- Volunteers for note taking?

Understanding Document Status

- Hosting within the W3C repo does ***not*** imply adoption by the WG.
  - WG adoption requires a Call for Adoption (CfA) on the mailing list.
- Editor's drafts do ***not*** represent WG consensus.
  - WG drafts ***do*** imply consensus, once they're confirmed by a Call for Consensus (CfC) on the mailing list.
  - Possible to merge PRs that may lack consensus, if a note is attached indicating controversy.
  - Possible to run CfCs on Issues, PRs or sections of a document.

W3C Code of Conduct

- This meeting operates under W3C Code of Ethics and Professional Conduct

- We're all passionate about improving WebRTC and the Web, but let's all keep the conversations cordial and professional

Virtual Interim Meeting Tips

This session is being recorded

- **Type +q and -q in the Google Meet chat to get into and out of the speaker queue.**
- **Please use headphones when speaking to avoid echo.**
- **Please wait for microphone access to be granted before speaking.**
- **Please state your full name before speaking.**
- **Poll mechanism may be used to gauge the “sense of the room”.**

Status of Recent CfCs

- CfC on Re-Publishing Media Capture and Streams at CR
  - https://lists.w3.org/Archives/Public/public-webrtc/2021Sep/0004.html
  - Completed September 17, 2021
  - 4 (positive) responses
  - Summary to follow (Jan-Ivar)
- CfC on Transferrable MediaStreamTracks
  - https://lists.w3.org/Archives/Public/public-webrtc/2021Sep/0007.html
  - Two responses so far
  - Will Complete on September 27, 2021

WHATWG Streams Issues

- Issue 1063: Transferable streams: the double transfer problem
- Issue 1124: Permitting transferable stream optimisation
- Issue 1155: Is when pull() is called on an underlying source deterministic?
- Issue 1156: Allow web authors to tee() with cloned chunks
- Issue 1157: Allow web devs to synchronize branches with tee()?
- Issue 1158: Piping to writable streams with HWM 0

Issues for Discussion Today

- 08:10 AM - 08:40 AM Conditional Focus (Elad Alon)
  - Slides (08:10 - 08:25)
  - Discussion (08:25 - 08:40 AM)
- 08:40 AM - 08:50 AM getViewportMedia (Elad Alon)
  - Slides + Discussion (8:40- 8:50)
- 08:50 AM - 09:10 AM Display Surface Constraints (Elad Alon)
  - Slides (8:50 - 9:00)
  - Discussion (09:00 - 9:10)
- 09:10 AM - 09:40 AM Echo Cancellation (Harald)
  - Slides (09:10 - 09:25)
  - Discussion (09:25 - 09:40)
- 09:40 - 10:00 AM Wrap-up and Next Steps

Time control:

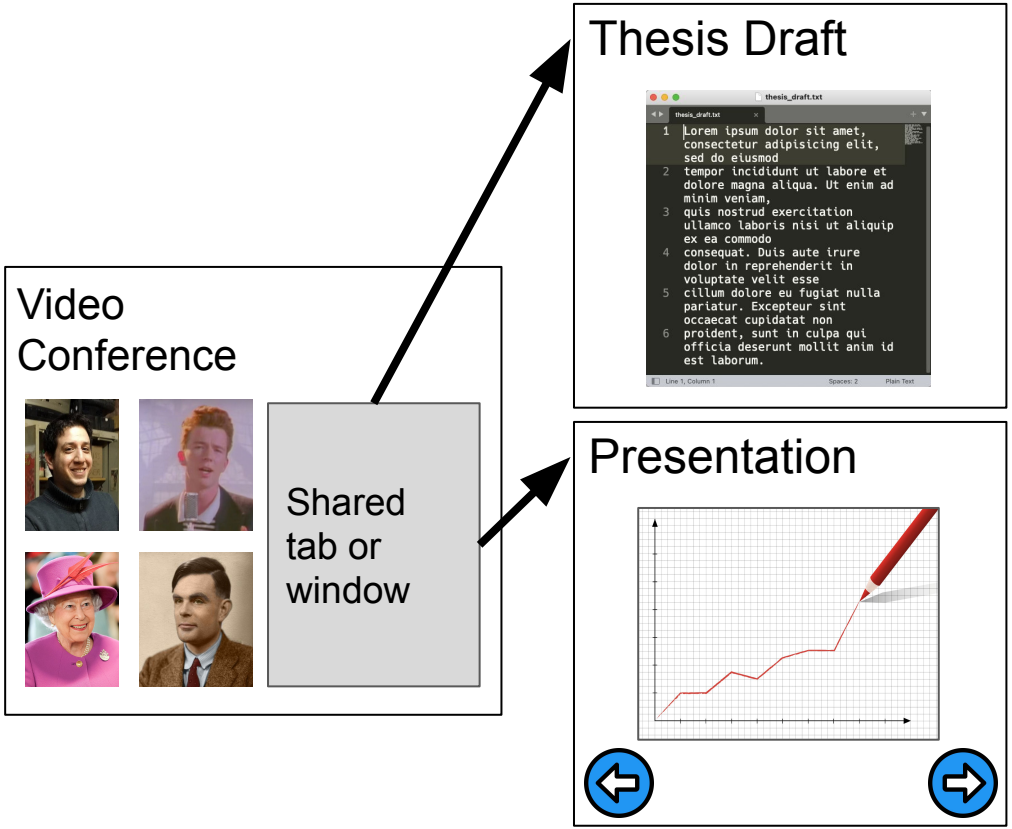
- A warning will be given 2 minutes before time is up.
- Once time has elapsed we will move on to the next item.

#190 Conditional Focus #1 (Elad Alon)

Question:
What do these all have in common? (The use-cases, that is.)

Answer:
When the video-conferencing application begins capturing either of these, the browser brings the captured tab/window to the forefront.

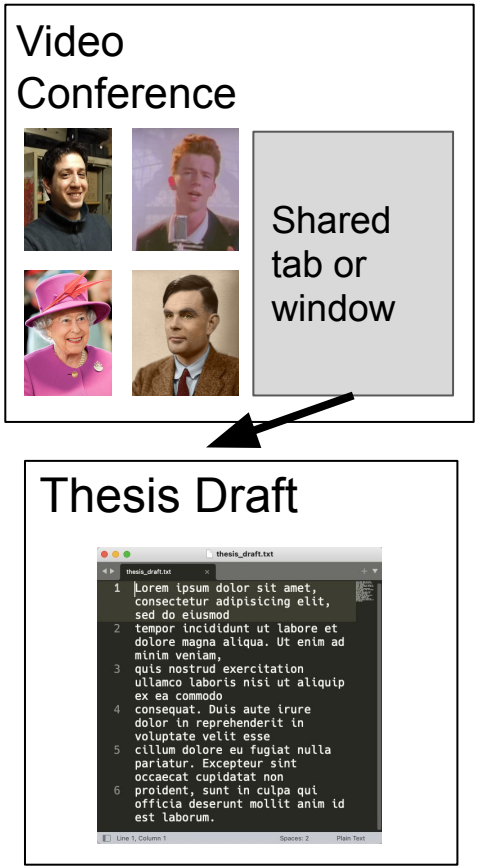
Is that really desirable?



#190 Conditional Focus #2 (Elad Alon)

Use-case #1:

Assume the user shares a document which they intend to co-author with remote participants. Switching focus to that document saves the user some time, allowing them to jump right into the action.



#190 Conditional Focus #3 (Elad Alon)

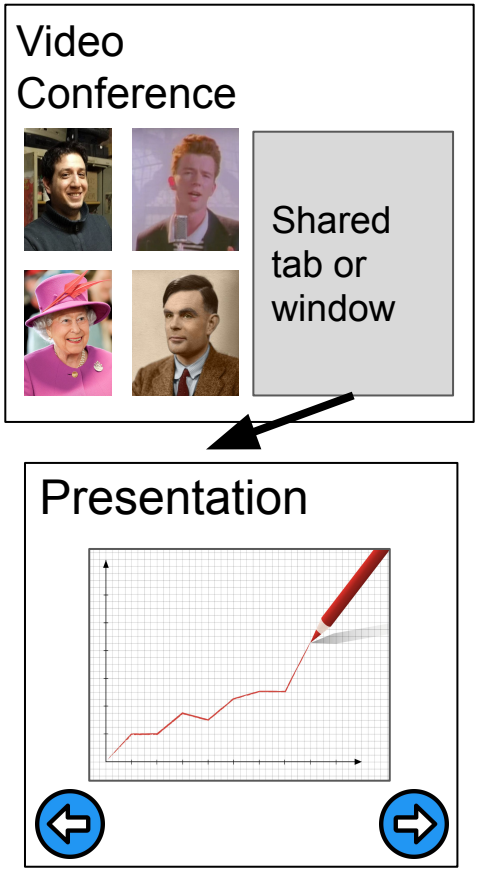
Use-case #2:

Assume the user shares a slide, a video, or something else with which they have minimal interaction.

Especially if Capture Handle is used, the user could be controlling the presentation remotely - see demo.

In such cases, the focus-change to the captured tab/window is an annoying distraction for the user and to remote participants. It adds friction. It risks confusing users as they try to navigate their way back to the video-conferencing tab while simultaneously talking to remote participants and trying to keep their train of thought from derailing.

Focus-change is problematic for non-VC applications, too: example.



#190 Conditional Focus #4 (Elad Alon)

The browser is mostly agnostic of the nature of both the capturing and captured applications.

- Is the capturing application video-conferencing, or is it a screen-recorder?
- Is the captured application interactive, like a text-editor, or non-interactive, like a static image?
- Can the capturing and captured applications collaborate, e.g. by exchanging messages to request the previous/next slide?

The browser cannot answer these questions.
But the capturing application can.
The capturing application knows itself. And using Capture Handle, it may recognize the captured application, too.

#190 Conditional Focus #5 (Elad Alon)

Suggested API:

- MediaStreamTrack is subclassed as FocusableMediaStreamTrack. (See later slide for more.)
- FMST gets a new method - focus().
  - It accepts either "`focus-captured-surface`" or "`no-focus-change`". Like a boolean, but more self-documenting.
- Non-trivial fine-print in next slide.

Code sample:

```
const stream = await navigator.mediaDevices.getDisplayMedia();
const shouldFocus = ShouldFocus(stream);
track.focus(shouldFocus ? "focus-captured-surface" : "no-focus-change");
```

Where ShouldFocus() is some arbitrary code making a decision.

- It could hard-code to always/never switch-focus to the captured tab. For example, an application performing a recording, may wish to never switch focus.
- It could base its decision on displaySurface. Maybe focus tabs, but not windows.
- It could base its decision on Capture Handle. For example, avoid focusing remote-controllable tabs.

#190 Conditional Focus #6 (Elad Alon)

Fine details:

- Recall that getDisplayMedia() returns a Promise. This Promise is resolved on a microtask. Name that specific microtask MT.
- Calling focus():
  - Before MT resolves - not possible; the app has no reference to the track.
  - On MT - legal and has the expected effect. (Caveat below.)
  - After MT - raises an exception. Prevents late-focus cross-tab clickjacking.
- Not calling focus():
  - After MT completes, if focus() was not explicitly called, the UA makes it own decision.
    - (Chrome intends to keep the existing behavior and default to switching focus.)
  - Regardless of MT's completion of lack thereof, if focus() is not called within 1s of the capture starting, the UA makes its own decision. (Avoids late-focus attacks involving busy-waiting.) Silent failure in this case.
- Note that MT-completion triggers the UA decision, thereby avoiding a needless 1s delay of focus-change by applications that do not explicitly call `focus("no-focus-change")`.
- Multiple calls to focus() - exception raised.
- Calling focus() on a clone - exception raised.

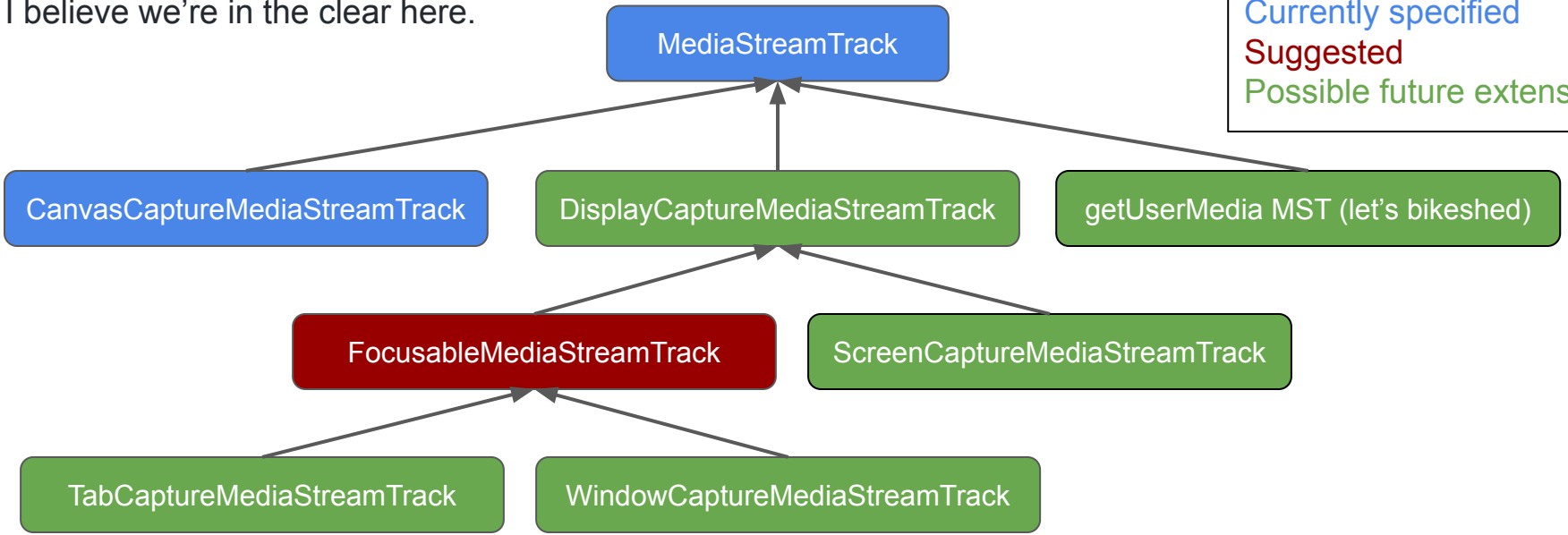
#190 Conditional Focus #7 (Elad Alon)

Vision for future MediaStreamTrack inheritance tree

We don't need to do most of this right-away, but it's good to make sure that by introducing FocusableMediaStreamTrack, we're not limiting our future options.

I believe we're in the clear here.

Legend
Currently specified
Suggested
Possible future extension



#155 getViewportMedia (Elad Alon)

Recap:

- We've been discussing getViewportMedia(), an API for capturing only the current tab. (Confirmation-only dialog; "take it or leave it.")
- Rough consensus from April on gating this API by a combination of:
  a. Cross-origin isolation (window.crossOriginIsolated a.k.a. COOP+COEP)
  b. Opt-in header (controlling whether embedded document may be captured)
  c. Permissions policy (controlling whether embedded document may capture)

Proposed:

- gVM always gets the current browsing context's viewport, even when called from an iframe. WG's thoughts?
- Opt-in shall use an off-by-default Document Policy, consisting of "Require-Document-Policy: viewport-capture" and "Document-Policy: viewport-capture".
  - WG's thoughts?

TBD:

- Name
- User activation - yes or no?

#184 Display Surface Constraints #1 (Elad Alon)

Recap

- When getDisplayMedia() is called, the user agent MUST offer the user an unlimited selection of all “available” display surfaces.
- The standard says “the user agent … MUST NOT use constraints to **limit** … choice.”
- The standard then presents an additional restriction: “constraints MUST be applied to the media chosen by the user, only after they have made their selection.”
  - The non-normative text explains how this prevents application from **influencing** user-choice.
- “Limiting” and “influencing” are not the same thing.
- The user agent is already influencing user-choice through the order in which it presents options.
- Influence can be wielded productively.
  - User agents can steer users towards safer options. (Mozilla has made multiple suggestions to this effect.)
  - Applications can steer users towards the most context-appropriate option.

#184 Display Surface Constraints #2 (Elad Alon)

Support and Rationale

- Large web-developers working on significant, legitimate applications, have been clamoring for the ability to influence the user's choice:
  - Examples: RingCentral, Pexip, Jitsi, Webex, Atos.
- Reasons cited include:
  - Save clicks on the journey towards the user's historic preferences.
  - Help users discover the surface type that supports audio.
  - Lead users to the option that supports high-fps capture.
  - Direct users to the app-relevant surface type.
  - Push users away from risky options.
  - Keep the user focused on options that the application would accept (apps that intend to terminate full-screen capture rather than share it with remote participants).

#184 Display Surface Constraints #3 (Elad Alon)

Proposal

- Allow constraints to influence the prompt presented to the user, but **not** to limit user-choice.
- The constraint serves as a hint to the user agent, instructing the UA as to what class of surfaces the applications believes to be the most appropriate. The constraint asks the user agent to present this class of surfaces most prominently - while still allowing the user to choose surfaces from another class.
- The UA is free to ignore this suggestion. For example, if the user agent believes this hint to be pushing the user towards a less safe option, such as the entire current screen.
- The UA is recommended to present additional warnings for especially dangerous options.
    - For example, when offering the user tabs, the UA is recommended to employ additional warning over the current tab, or place it last, or require additional confirmation.

#184 Display Surface Constraints #4 (Elad Alon)

Proposed Spec

getDisplayMedia

Prompts the user for permission to live-capture their display.

The user agent *MUST* let the end-user choose which display surface to share out of all available choices every time, and *MUST NOT* use *constraints* to limit that choice.

The user agent *MAY* allow the application to use the `displaySurface` constraint to signal a preference for a specific display surface type. The user agent *MUST* still offer the user unlimited choice of any display surface, but *MAY* order the sources offered to the user according to this constraint.

Other than the `displaySurface` constraint, *constraints MUST* be applied to the media chosen by the user only after the user has made their selection.

Open question - should the spec call out "ideal" as the only permissible value type?

#184 Display Surface Constraints #5 (Elad Alon)

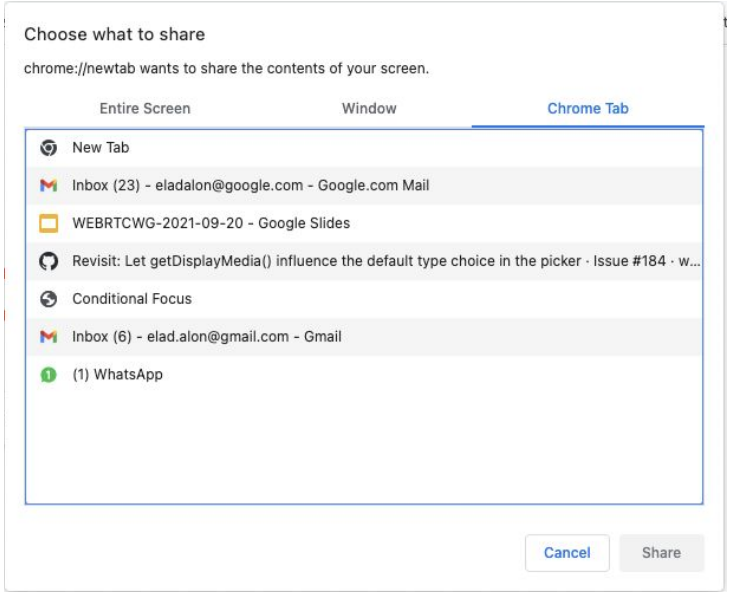
Sample Code and Illustration of UA-Implementation

For the example below, recall that when display surfaces are concerned, the word “browser” basically refers to a tab.

```
navigator.mediaDevices.getDisplayMedia({
  video: {
    displaySurface: {ideal: browser},
  },
});
```

In response to this code, Chrome preselects the requested display surface **category**, but:

- Does **not** preselect any member of that category.
- Does **not** stop the user from selecting another category.



Echo Cancellation: Choosing the Reference

(mediacapture-extensions issue #31, PR #32)

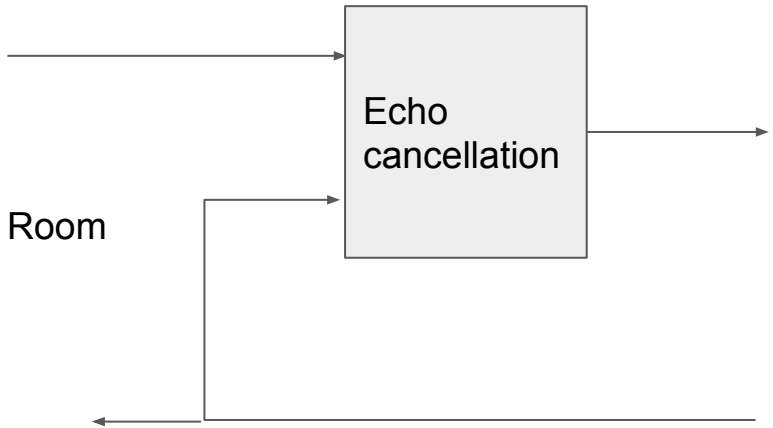
Echo cancellation consists of 2 inputs, 1 output.

- Input 1 is “signal + transform(feedback)”
- Input 2 is “feedback”
- Output is “signal with no feedback”

Echo cancellation is complex because the “transform” function is complex:

- Delay
- Distortion

Still, echo cancellation depends critically on knowing what it is supposed to remove.



Current implementation (Chrome)

Reference signal is “sum of all audio outputs from PeerConnection”

Not cancelling noise from other sources

Not able to deal with audio piped through processing elements

Not ideal.

More ideal situation: Cancel audio outputs

Requires access to system output. Sometimes possible; sometimes we have to fall back on “what the browser emits”.

Difficulty: In some cases, what’s on the speaker is not what’s on the feedback.

- Some headphones have strong acoustic or electrical coupling
- Some setups use an output different from the native speaker

The need is:

- Identify what output is being used for the most important thing to cancel
- Specify to the echo canceller that it should use this output

Proposal: echoCancellationReferenceSinkId

Uses an existing enumeration of output devices: SinkID from mediacapture-output

Can be specified as a constraint on a MediaStreamTrack

Special value “” means the default audio output of the system (or as close to it as we can get given implementation constraints)

Decision to be taken by WG:

- Accept this proposal, merge PR
- Reject this proposal, close PR

May or may not need ML Consensus Call

Discussion

Wrap-up and Next Steps (9:40 - 10:00)

Followup items:

Thank you

Special thanks to:

WG Participants, Editors & Chairs