W3C WebRTC WG Meeting

April 28, 2020
8 AM 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 interim meeting of the W3C WebRTC WG!
  - During this meeting, we will talk about new work and make progress on privacy and security concerns.

About this Virtual Meeting

Information on the meeting:

- Meeting info:
  - https://www.w3.org/2011/04/webrtc/wiki/April_28_2020
- Link to latest drafts:
  - https://w3c.github.io/mediacapture-main/
  - 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-stats/
  - https://www.w3.org/TR/mst-content-hint/
  - https://w3c.github.io/webrtc-nv-use-cases/
  - https://w3c.github.io/webrtc-dscp-exp/
  - 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.

Issues for Discussion Today

- Insertable Streams (Harald)
- Content-Hints
  - PR 40: speechRecognition content hint (Sam Dallstream)
- Media Capture & Streams
  - Issue 688: Clarify only fire devicechange event when devices physically added/removed (Jan-Ivar)
  - Issue 668: What happens when a machine suspends? (Harald)
  - Issue 669: "user-chooses": Does required constraints make any sense now? (Henrik)
  - Issue 672: Deprecate inputDeviceInfo.getCapabilities() for privacy (Jan-Ivar)
- Media Capture Output
  - Issue 87: Setting the audio output for a whole page (Youenn)

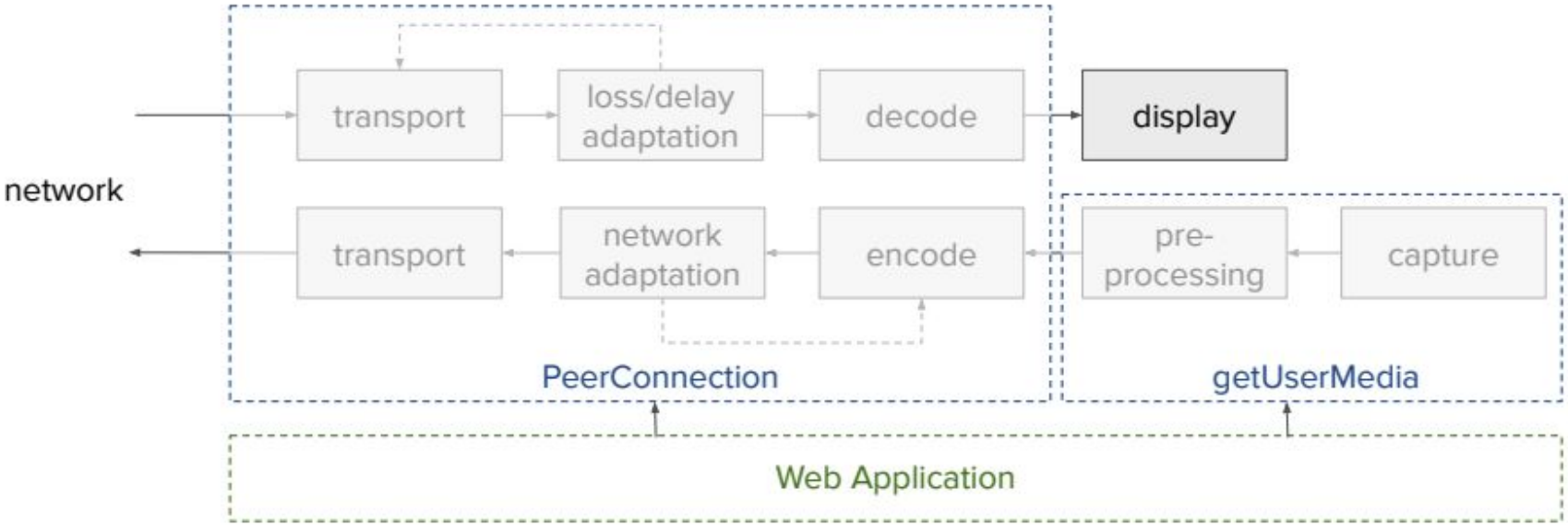
Insertable Streams (Harald)

- The WebRTC-NV Use Cases document derived several requirements from the "Funny Hats", "Machine Learning" and "Virtual Reality Gaming" use cases, including:

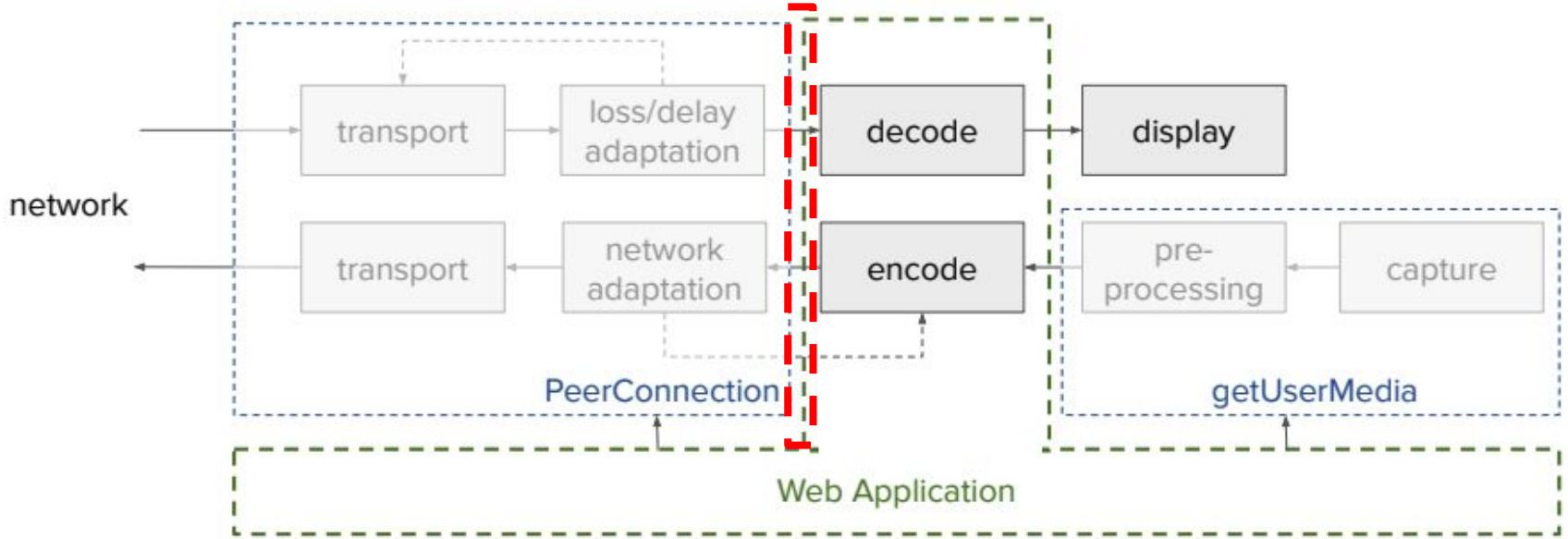
N18	The application must be able to obtain raw media from the capture device in desired formats.
N19	The application must be able to insert processed frames into the outgoing media path.
N20	The application must be able to obtain decoded media from the remote party.
N21	It must be possible to efficiently share media between the main thread and worker threads.
N22	It must be possible to do efficient media manipulation in worker threads by utilizing the GPU.
N23	The user agent must be able to send data synchronized with audio and video.

- The Insertable Streams API potentially addresses requirements N19 and N23.
- Transferable Streams (used by Insertable) addresses requirement N21.
- The Insertable Streams approach might address requirements N18 and N20 for some use cases. Is sufficient performance attainable without N22?

RTC Media Flow in WebRTC 1.0

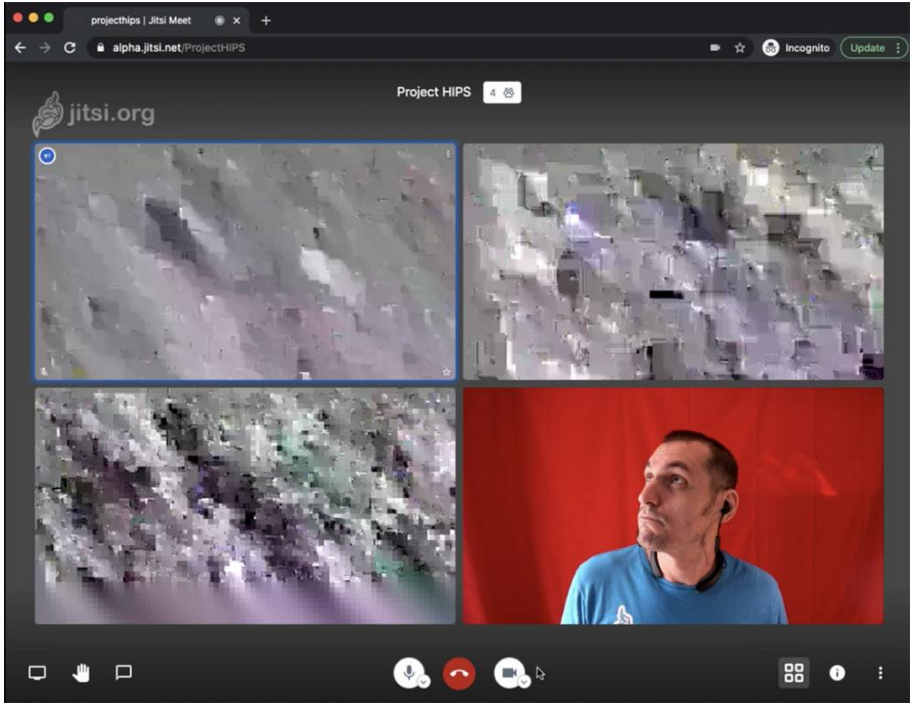


Insertable Streams Encode/Decode Flow



Insertable Streams Experiment

- In Chrome Canary M83
- Tested with:
  - Duo Web Group Calling
    - Pre-existing E2E crypto solution - web client must conform
  - Jitsi Meet (article)
  - Medooze for AR (article)
- Performance appears adequate
- Needs adoption



Insertable Streams - WebIDL API

```
partial dictionary RTCConfiguration {
    boolean forceEncodedVideoInsertableStreams = false;
    boolean forceEncodedAudioInsertableStreams = false;
};

partial interface RTCRtpSender {
    RTCInsertableStreams createEncodedVideoStreams();
    RTCInsertableStreams createEncodedAudioStreams();
};

partial interface RTCRtpReceiver {
    RTCInsertableStreams createEncodedVideoStreams();
    RTCInsertableStreams createEncodedAudioStreams();
};
```

```
dictionary RTCInsertableStreams {
    ReadableStream readableStream;
    WritableStream writableStream;
};

enum RTCEncodedVideoFrameType {
    "Empty", "key", "delta",
};

interface RTCEncodedVideoFrame {
    readonly attribute RTCEncodedVideoFrameType type;
    readonly attribute unsigned long long timestamp;
    attribute ArrayBuffer data;
    readonly attribute ArrayBuffer additionalData;
};
```

Insertable Streams - example use

```
let senderTransform = new TransformStream({
    async transform(chunk, controller) {
      let view = new DataView(chunk.data);
      let newData = new ArrayBuffer(chunk.data.byteLength + 4);
      let newView = new DataView(newData);
      // Invert and pad the bits in the frame
      for (let i = 0; i < chunk.data.byteLength; ++i)
        newView.setInt8(i, ~view.getInt8(i));
      // Set the padding bytes to zero.
        newView.setInt8(chunk.data.byteLength + i, 0);
      chunk.data = newData;
      controller.enqueue(chunk);
    },
  });
```

```
let senderStreams =
videoSender.getEncodedVideoStreams();

// After ICE and offer/answer exchange.

senderStreams.readableStream
  .pipeThrough(senderTransform)
  .pipeTo(senderStreams.writableStream);
```

Insertable streams and Workers

- Uses Transferable Streams to connect a Worker into streams defined in a page
- Makes other activities less of a problem
  - Work off the main thread!
- Makes an easy separation of concerns
  - You can even use other people's workers
- Example code is available now

Insertable Streams - Next Steps

- Experiment with API in a real app under Chrome's "origin trial" mechanism
  - Origin trial in progress (Chrome Beta)
- Synchronize with WebCodecs as appropriate
- Investigate extension to raw media
- Propose (revised) API for standardization
  - Expect to call for consensus to adopt in May

Issues for Discussion Today

- Content-Hints
  - PR 40: speechRecognition content hint (Sam Dallstream)

PR 40: speechRecognition content hint (Sam Dallstream)

- Problem Statement:
  The specification, as it stands today, does not allow developers to differentiate streams for speech recognition and communication.

- Mst-content-hint only contains the ‘speech’ hint. Which seems to be geared for communication.
- Communications modifications generally hurt speech recognition and vice versa.

PR 40: speechRecognition content hint (Sam Dallstream)

Examples of areas where speech recognition streams differ from communications streams.

	Communications Audio	Speech Audio
Echo (Suppression)	Echo leakage is intolerable to human listeners >40 dB speakerphone, >46 dB handsets, ITU-T TS 26.131	Echo leakage is tolerable with sufficient speech level Typically >15-20 dB speech to echo
Echo (Switching)	Switching (slight loss of initial syllables) is used to avoid echo leakage, and slight impairments are not noticeable to human listeners ITU-T P.501, P.502, TS 26.131, P.1100, G.131	Any switching resulting in slight loss or attenuation of syllables impairs barge-in and introduces word error rate STQ63-250 Section 5.2
Ambient Noise	Focused on perceived quality/distraction from speech/noise. Some ambient noise is tolerable is it provides contextual cues in human perception without distraction Noise sources diffuse ITU-T TS 26.131, P.835, ETSI EG 202 396	Not concerned with human perception but preservation of source utterance and removal of background noise Noise sources diffuse + discrete in test to evaluate rejection nearby non-users and noises STQ 63-250 Section 4.2 Alexa Acoustic Testing V3.5.6

PR 40: speechRecognition content hint (Sam Dallstream)

Room Acoustics (Reverb)	Typically ok to have some amount of reverb, as audio provides contextual cues in human perception without distraction	Devices concerned with preservation of source utterance. Room reflections introduce loss of information in frequency regions and phonemes
Comfort Noise	Desired to avoid perception that call has dropped ITU-T G.711	Undesirable to add any additional noise that impairs source utterance
Sound Quality	Sensitive to human perception ITU-T P.863	Sensitive to preserving speech formants STQ 63-250 Section 6 Alexa Acoustic Testing V3
Level/Gain Control	Standardized to match cross-network communications architectures ITU-T G.111, G.121, TS 26.131	Dependent on trained model, often less sensitive

PR 40: speechRecognition contentHint (Sam Dallstream)

Proposal

- Add a new contentHint for Audio Tracks to mst-content-hint called “speechRecognition” to allow developers to differentiate.
- Link to explainer

Pros

- Minimal changes, only modifies mst-content-hint draft.
- Low-risk, and allows for prototyping and gathering feedback from developers.

Cons

- contentHint is not used in all browsers.

Issues for Discussion Today

- Media Capture & Streams
  - Issue 688: Clarify only fire devicechange event when devices physically added/removed (Jan-Ivar)
  - Issue 668: What happens when a machine suspends? (Harald)
  - Issue 669: "user-chooses": Does required constraints make any sense now? (Henrik)
  - Issue 672: Deprecate inputDeviceInfo.getCapabilities() for privacy (Jan-Ivar)
- Media Capture Output
  - Issue 87: Setting the audio output for a whole page (Youenn)

Issue 688: Clarify only fire devicechange event when devices physically added/removed (Jan-Ivar)

2014-2018 QUIZ:

Q: The `devicechange` event fires when:

A) The user inserts or removes a device
B) The enumerateDevices() list changes
C) Those are the same thing!

Issue 688: Clarify only fire devicechange event when devices physically added/removed (Jan-Ivar)

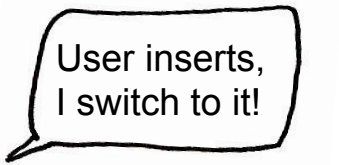
2014-2018 QUIZ:

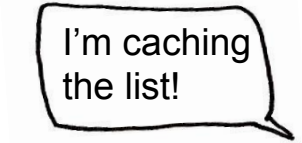
Q: The `devicechange` event fires when:

A) The user inserts or removes a device

B) The enumerateDevices() list changes

C) Those are the same thing! ✔





Issue 688: Clarify only fire devicechange event when devices physically added/removed (Jan-Ivar)

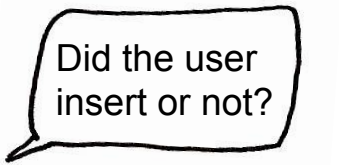
2020 QUIZ:

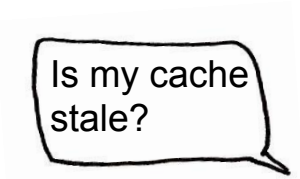
Q: The `devicechange` event fires when:

A) The user inserts or removes a device

B) The enumerateDevices() list changes

C) ~~Those are the same thing!~~





Issue 688: Clarify only fire devicechange event when devices physically added/removed (Jan-Ivar)

The *"new devices detection"* algorithm for most years of this spec has been a simple subtract:

```
let old = await navigator.mediaDevices.enumerateDevices();
navigator.mediaDevices.ondevicechange = async () => {
  const devices = await navigator.mediaDevices.enumerateDevices();
  const newDevices = devices.filter(d => !old.find(o => o.deviceId == d.deviceId));
  old = devices;
};
```

Expecting web devs to do more than that seems ambitious. As I recall, the use-cases are:

1. **Pre-gUM:** User inserts a device for which they had none before, qualifying them for feature.
2. **Post-gUM:** User inserts a device (e.g. headset) preferred over the one they have now.

Now broken (no deviceIds pre-gUM). Apps doing device detection will have to call enumerateDevices() again on gUM success to update their list, or suffer false positives.

Issue 688: Clarify only fire devicechange event when devices physically added/removed (Jan-Ivar)

devicechange says: "The set of media devices, *available to the User Agent*, has changed." (***Not*** *"the application"*) ... “When new media input and/or output devices are made available”

PASSIVE LANGUAGE!

Use case: Inserting a USB device or enabling a BT headset (e.g. putting on AirPods) during a call, is a **strong signal** users want to switch to it, that many apps want, like a key press.

But since October (#632) Safari fires devicechange on gUM success to update list, which is indistinguishable from a USB insert because list grows on systems w/2 cameras or 2 mics.

Problem: as a web developer, I can no longer differentiate between:

1. The user just inserted or enabled a headset, which I want to switch to immediately.
2. The user did not insert or enable anything, and I shouldn't switch to a secondary device.

Issue 688: Clarify only fire devicechange event when devices physically added/removed (Jan-Ivar)

Proposal A:

- Allow the JS list to change without an event when getUserMedia succeeds.
- Only fire `devicechange` when devices added/removed by user action.

Proposal B:

- Only changes to JS list cause `devicechange` to fire (e.g. to/from 0 post-gUM firing)
- Fire a new `deviceinserted` event if one or more devices added by user action.

Issue 668: What happens when a machine suspends? (Harald)

Background

- Implementation property: Chrome used to fire a “closed” event and close the PC when machine suspended.
- This was used by some users. Caused confusion when we removed the event.
- Never in spec, no corresponding event in DOM that would let us specify it.
- Recommendation from editors: Do nothing. Continue after suspend, and let the timeouts fall where they may.

Led to discussion on what happens with devices.

- Locked device doing recording is “creepy”.
- Suggestion: Mute cameras and microphones on suspend (fire muted), resume (and fire unmuted) when user is able to interact with device (unlocked) - not before.
- Searching for DOM events that correspond to those times in order to specify.

Issue 669: "user-chooses": Do required constraints make any sense now? (Henrik)

In-chrome pickers competes with in-content pickers. *Where are we headed?*

Today, **“required”** constraints remove devices from the selection.

- Does this make any sense with **“user-chooses”?**
  Example: SD camera faces me, HD camera faces my room. Application prefers HD, but that’s not what the user wants! Why not let the user pick?

To what extent should filtering out devices be allowed in **“user-chooses”**?
More importantly, to what extent do we want to expose deviceIds and labels?

Can of worms?

- deviceIds are used for in-content selection.
- deviceIds are used to avoid prompt when re-visiting website.
- deviceIds are used to toggle cameras (e.g. “front” and “back”).

Issue 669: "user-chooses": Do required constraints make any sense now? (Henrik)

Proposal: When using “user-chooses”...

- deviceId can still be **required** and filter out devices.
- Any other **required** constraints are *ignored if they would reduce the set of devices*.
  - E.g. you may force HD on a HD/LD device, but you may not exclude LD-only devices.

Flavor A: Full in-content picker.

- deviceId and labels of *ALL* devices are exposed when permission is granted to *ANY* device.

Flavor B: Partial in-content picker.

- Expose current and *minimally sensitive deviceIds* like for “front” and “back” camera.
- For other devices, a special deviceId for “other devices” to cause re-prompt.

Flavor C: In-content picking is not supported.

- Deprecate deviceIds. Only user selects.
- Avoid re-prompt on revisit with **{previousDevice:true}** which prefers “whatever I used last time on this domain”, may avoid re-prompt.

Issue 672: Deprecate inputDeviceInfo.getCapabilities() for privacy (jib)

Chrome/Edge & Safari have `info.getCapabilities()` w/info on all devices after gUM.

Reason: Lets site enforce its constraints while building picker, or choosing other device outright. Most sites enforce some constraints. **But:** It's a trove of fingerprinting info!

`"user-chooses"` provides feature-parity, without the information leak:

```
await navigator.mediaDevices.getUserMedia({video: constraints, semantics: "user-chooses")
```

So once #667 merges, can we deprecate `info.getCapabilities()`?

Issue 87: Setting the audio output for a whole page (youenn, 1/2)

Use cases for HTMLMediaElement.setSinkId

- Application provides UI for user to select speaker
  - Need to apply setSinkId to every current or future element of the page
    - No support for third-party iframes (require tight coordination)
  - No WebAudio support without a new API
- Application uses the default audio output for all sources except one
  - HTMLMediaElement.setSinkId well suited here
  - Main use-case seems to be notification and there is the notification API

Additional notes

- User often expects one and only one audio output
- The OS usually provides UI to allow user to know where audio is flowing
  - Might be confusing for user if they are not in sync

Issue 87: Setting the audio output for a whole page (youenn, 2/2)

Proposal

- Provide a way to set the audio output for the whole page
  - navigator.mediaDevices.sinkId/setSinkId
- Global value can be selectively overridden using HTMLMediaElement.setSinkId

Advantages

- A global property is closer to what user expects and what OS UI is showing
- Very easy to implement the first scenario (probably most important scenario?)
- Works with third-party iframes
  - Feature policy to allow third party iframes to mutate this global value

Additional question

- With notifications API and a page-wide audio output setter, do we still need HTMLMediaElement.setSinkId?

For extra credit

Name that bird!

Thank you

Special thanks to:

WG Participants, Editors & Chairs

The bird