W3C WebRTC WG Meeting

November 26, 2019
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 hope to decide on issues blocking the final candidate recommendation webrtc-pc as well as to make progress on webrtc-stats and privacy issues.
  - We expect to re-issue CRs of WebRTC-PC and WebRTC-Stats after this meeting.

About this Virtual Meeting

Information on the meeting:

- Meeting info:
  - https://www.w3.org/2011/04/webrtc/wiki/November_26_2019
- 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

- WebRTC-Extensions: Adopt WebRTC Extensions spec?
- WebRTC-PC
  - Issue 2313: [[SendCodecs]] is updated based on answers, but [[SendEncodings]] is updated based on remote description (Harald)
  - Issue 2369: [[ReceiveCodecs]] is only set on answer. How to achieve early media? (Harald)
  - Issue 2315: Racy setParameters/getParameters behavior (Jan-Ivar)
  - Issue 2314: Can simulcast offers renegotiate RIDs? (Jan-Ivar)
  - Issue 2316: Can IceTransport go checking -> completed? (Harald)
  - Issue 2363/PR 2372: Transports on offer need to be rolled back (Henrik)
  - Issue 2368/PR 2373: Clarify what happens to codecs when rolling back (Henrik)
  - Issue 2367: Clarify what happens when rolling back an ICE restart (Henrik)
  - Issue 2330: Backwards compat concerns with RTCError (Henrik)
- WebRTC-Stats
  - Issue 304: Stats need to markup which members are required (Harald)
  - Issue 374: Exposing RTCIceCandidateStats.networkType might trigger fingerprinting (Youennf)
- Media Capture and Streams
  - Issue 612: Move enumerateDevices behind permission (Youennf)
  - Issue 639: Enforcing user gesture for getUserMedia (Youennf)
  - Issue 640: Only reveal labels of devices user has given permission to (Jan-Ivar)
  - Issue 642: Only Firefox turns off device on disabled track. Stronger language needed?

Adopt WebRTC Extensions spec?

A number of features have been added to webrtc-extensions. Including…

“At risk” features moved from webrtc-pc:

- `RTCPeerConnection.getDefaultIceServers()`
- `RTCRtpEncodingParameters.maxFramerate`
- OAuth (`RTCIceCredentialType::"oauth"`, `RTCOAuthCredential`, `RTCIceServer.credential`)

New features:

- `RTCRtpReceiver.playoutDelayHint` (to be renamed `playoutDelay`?)
- `RTCRtpEncodingParameters.adaptivePtime`
- *(Being discussed)* `hardwareAccelerated codec capabilities`

Extensions need an official home. See also webrtc-pc#2323.

Is this Working Group interested in adopting this document?

Issues for Discussion Today

- WebRTC-PC
  - Issue 2313: [[SendCodecs]] is updated based on answers, but [[SendEncodings]] is updated based on remote description (Harald)
  - Issue 2369: [[ReceiveCodecs]] is only set on answer. How to achieve early media? (Harald)
  - Issue 2315: Racy setParameters/getParameters behavior (Jan-Ivar)
  - Issue 2314: Can simulcast offers renegotiate RIDs? (Jan-Ivar)
  - Issue 2316: Can IceTransport go checking -> completed? (Harald)
  - Issue 2363/PR 2372: Transports on offer need to be rolled back (Henrik)
  - Issue 2368/PR 2373: Clarify what happens to codecs when rolling back (Henrik)
  - Issue 2367: Clarify what happens when rolling back an ICE restart (Henrik)
  - Issue 2330: Backwards compat concerns with RTCError (Henrik)

Issue 2313: [[SendCodecs]] is updated based on answers, but [[SendEncodings]] is updated based on remote description (Harald)

- Original issue focused on the PT (payload type) value being updated inconsistently.
- Later updates removed the payload type from RTCRtpCodingParameters; it remains (read-only) in RTCRtpCodecParameters.
- We could argue that SendCodecs should be set on offer, because PTs get allocated then, but the referential issue is gone.
- Proposal: Close issue. Open a new one if there is a problem.

Issue 2369: [[ReceiveCodecs]] and early media (Harald)

- Per spec, [[ReceiveCodecs]] is only set when type is “answer” or “pranswer” (2 places: local and remote)
- This means it’s null when we are the offerer. This is a bug.
- Early media means that when we offer a new PT, media can arrive with that new PT before we see a (delayed) answer.
- Can’t happen at initial O/A because keys are unknown - can’t decrypt.
- Can happen at subsequent O/A.
- Down-negotiation affects sending; it shouldn’t logically affect receiving (we can still handle what we can handle).
- PTs can’t change their type, ever.

Proposal: Move [[Receiver]].[[ReceiveCodecs]] setting out of the “if description is answer” block. Always set it to all assigned PTs, with what we can receive.

Issue 2315: Racy setParameters/getParameters behavior (Jan-Ivar)

Problem B: *getParameters()* has side-effects / action at a distance (it bumps `transactionId`):

```
const params = pc.getParameters();
someUnrelatedFunction();
await pc.setParameters(params); // InvalidStateError if bar == true!

function someUnrelatedFunction(pc) {
  if (bar) console.log(JSON.stringify(pc.getParameters());
}
```

Problem A: The following may race with a remote simulcast offer:

```
const params = pc.getParameters();
await wait(0);
await pc.setParameters(params); // Intermittent InvalidStateError!
```

...because SRD clears [[LastReturnedParameters]] if it gets a simulcast offer.

Issue 2315: Racy setParameters/getParameters behavior (Jan-Ivar)

Solution: Queue a task in *getParameters* to clear [[LastReturnedParameters]].
Only increment *transactionId* when [[LastReturnedParameters]] is null.

Solves B: It's now a getter with no discernible side-effects:

```
const params = pc.getParameters();
someUnrelatedFunction();
await pc.setParameters(params); // Always works (same transactionId)

function someUnrelatedFunction(pc) {
  if (bar) console.log(JSON.stringify(pc.getParameters());
}
```

Solves A: Async behavior is now deterministic (always fails).

```
const params = pc.getParameters();
await wait(0);
await pc.setParameters(params); // Always InvalidStateError
```

Issue 2314: Can simulcast offers renegotiate RIDs? (Jan-Ivar)

JSEP says: *"Offers and answers inside an existing session follow the rules for initial session negotiation. Such an offer **MAY** propose a change in the number of RIDs in use. To avoid race conditions with media, any RIDs with proposed changes **SHOULD** use a new ID, rather than re-using one from the previous offer/answer exchange. RIDs without proposed changes **SHOULD** re-use the ID from the previous exchange."*

Differs from local side: *addTransceiver* says *"Once the envelope is determined, layers cannot be removed."*

For interop, should webrtc-pc dictate how implementations MUST respond? **If yes, Options:**

Once [SendEncodings].length > 1 already then incoming simulcast offer does what?

Options: When interpreting changes to existing encodings:

1. Ignore all changes to existing encodings
2. Update encodings (and order) based on matching rids (**Q:** is order significant?)
3. Update encodings based on order, (moving rids around but) no renaming otherwise
4. Erase all encodings, replace with new ones

Non-match options: 5a. Create new encodings, or 5b. Ignore new encodings
Missing options: 6a. Remove missing encodings. b. Leave them alone. c. Set `active=false`?

Issue 2316: Can IceTransport go checking -> completed? (Harald)

- RTCIceTransport.state reflects the [[IceTransportState]] slot
- A single iceTransport can:
  - Be set to “new” on creation
  - Be updated by the ICE Agent
  - Go to “closed” on close()
- The “completed” state says:
  - The RTCIceTransport has finished gathering, received an indication that there are no more remote candidates, finished checking all candidate pairs and found a connection. - 5 conditions.
- Conditions 1, 2 and 3 can happen while in “checking”.
- Condition 4 (“finished checking”) and condition 5 (“found a connection”) can be the same event - but only if the last checked candidate is the only successful one.
- Proposed answer: YES. This should be mentioned in a “Note” paragraph.

Issue 2363/PR 2372: Transports on offer need to be rolled back (Henrik)

Transport life-cycle (see codepen on Chrome) is asymmetrical:

- For the offerer, transports are created at SLD(offer).
  - This includes “in case the remote endpoint does not support bundling”-transports.
  - The additional transports are discarded at SRD(answer) if bundling is accepted.
- For the answerer, transports are created at SRD(answer).

This makes sense for “early media” use cases. Thus, transports in “have-local-offer” need to be considered “transports on offer”.

Currently, transports are not nulled on “rollback”, even though they are “on offer” like everything else that we roll back.

Proposal: Revert the RTCRtp[Sender/Receiver].transport internal slot to the previous stable state value on rollback (i.e. null or a previously established transport).

Issue 2368/PR 2373: Clarify what happens to codecs when rolling back (Henrik)

Per earlier slide **Issue 2369: [[ReceiveCodecs]] and early media**, [[ReceiveCodecs]] can be set at setLocalDescription(offer).

- Therefore, we need to be able to roll back [[ReceiveCodecs]].
- [[SendCodecs]] is a non-issue for rollback because it is only set at “answer”.

Proposal: Revert [[ReceiveCodecs]] to the previous stable state value.

Issue 2368/PR 2373: Clarify what happens to codecs when rolling back (Henrik)

Per earlier slide **Issue 2369: [[ReceiveCodecs]] and early media**, [[ReceiveCodecs]] can be set at setLocalDescription(offer).

- Therefore, we need to be able to roll back [[ReceiveCodecs]].
- [[SendCodecs]] is a non-issue for rollback because it is only set at “answer”.

Proposal: Revert [[ReceiveCodecs]] to the previous stable state value.

Related concern: Is there a race between getParameters(), rollback, and setParameters() that we need to be aware of?

Does this require special handling in the Perfect Negotiation example (#2370)?

Issue 2367: Clarify what happens when rolling back an ICE restart (Henrik)

```
pc1.restartIce();
pc1.onicecandidate = () = { <send newCandidate to pc2> }
const pc1Offer = await pc1.setLocalDescription();
<send pc1Offer to pc2>
async function onReceiveRemoteOffer(pc2Offer) {
  // The implicit rollback invalidates newCandidate.
  // If pc2, as the unpolite peer, does not setRemoteDescription(pc1Offer) then
  // addIceCandidate() would throw.
  await pc1.setRemoteDescription(pc2Offer);
}
```

pc2 will presumably not set pc1Offer because it is in has-local-offer.
pc2.addIceCandidate(newCandidate) would throw!
Upon pc1 restarting ICE again due to renegotiation, new candidates would be generated and sent.

Proposal: This is indented behavior; same problem as a regular incoming offer collision. Close #2367, but cover this in Perfect Negotiation example (#2370).

Issue 2330: Backwards compat concerns with RTCError (Henrik)

Today operations throw OperationError, e.g:

```
{
  code: 0,
  name: 'OperationError',
  message: 'Failed to execute…',
  stack: 'Failed to execute…',
}
```

Versus the spec's RTCError extending DOMException:

```
{
  code: 0,
  name: 'RTCError',  // Backwards-compatibility concern!
  message: 'Failed to execute…',
  stack: 'Failed to execute…',
  // Additional fields, nice to have, unlikely to cause backwards-compatibility issues.
  errorDetail: 'sdp-syntax-error',
  sdpLineNumber: 42,
  httpRequestStatusCode: null, sctpCauseCode: null, receivedAlert: null, sentAlert: null,
}
```

Proposals:

- Construct RTCError with compatible name ('OperationError', are there other names?).
- Mark additional fields *At risk*.

Issues for Discussion Today

- WebRTC-Stats
  - Issue 304: Stats need to markup which members are required (Harald)
  - Issue 374: Exposing RTCIceCandidateStats.networkType might trigger fingerprinting (Youenn)

Issue 304: Stats need to markup which members are required (Harald)

- Not a blocker.
- Some stats would be meaningless if some fields were missing
  - Example: RTCDataChannelStats without “dataChannelIdentifier”
- Other stats are meaningful even if fields are missing
  - Example: RTCCodecStats.channels, which is usually only set for audio in stereo mode
- Can use “required” keyword to point this out
- Largely editorial task - no code changes
- Proposed resolution: Do, but after CR is published

Issue 374: Exposing RTCIceCandidateStats.networkType might trigger fingerprinting (Youenn)

networkType reveals whether a candidate is “wifi”, “ethernet”, “vpn”, etc.

- Main use case is to do bad connection analytics and identify root causes of network issues

Problem

- This increases the fingerprinting surface
- This could be misused to try optimizing the service based on this information
- The user does not need to provide this information for the website to provide the service

Proposal

- Move this stat to an extension spec
  - This stat is not as mature and implemented as other stats
  - This does not require any change to existing implementations
    - Implementations shipping this stat can expose this information and be compliant
- Refine the proposal in the extension spec
  - Identify precise use cases, find more focused solutions
  - Remove 'vpn', reduce allowed values to 'wireless' and 'wired'
  - Only expose this stat for the selected candidate pair

Issues for Discussion Today

- Media Capture and Streams
  - Issue 612: Move enumerateDevices behind permission (Youenn)
  - Issue 639: Enforcing user gesture for getUserMedia (Youenn)
  - Issue 640: Only reveal labels of devices user has given permission to (Jan-Ivar)
  - Issue 642: Only Firefox turns off device on disabled track. Stronger language needed? (Jan-Ivar)

Issue 612: Move enumerateDevices behind permission (Youenn)

- Problem: enumerateDevices is leaking fingerprinting information
  - The number of devices and persistent device ID values
- Solution: provide limited information by default
  - Spec mandates to expose at most one device of each type, no device ID
  - If getUserMedia is called successfully, provide all available information
- Can we strengthen the model?
  - Remove 'device-info' permission
  - Expose no device information at all by default

Issue 612: Move enumerateDevices behind permission (Youenn)
Remove 'device-info' permission

- Issue
  - Web site requests camera to take a profile photo
  - On user next visits, web site can freely monitor device setup
- 'device-info' permission is difficult to understand
  - Not exposed to user/not controllable by user
- 'device-info' permission is no longer needed
  - Expected flow is to call getUserMedia and then enumerateDevices
- Proposal PR641: Replace 'device-info' by an internal slot stored on the page
  - Slot is 'denied' by default, 'granted' after one successful getUserMedia call
  - Slot goes away on page navigation
  - Slot can be set to 'denied' by User Agent during lifetime of the page
    - User explicitly revokes access, page is not capturing for a long time...

Issue 612: Move enumerateDevices behind permission (Youenn)
Expose no device information at all by default

- Problem: Leaks whether 1 camera and/or 1 microphone is available
- Information is (maybe?) useful
  - Page can have a dedicated UI if no capture device is available
  - Page can update its UI if it sees once a capture device becomes available
- Potential solutions
  - Make enumerateDevices return a list with 1 camera and 1 microphone
    - Empty the list if getUserMedia returns NotFoundError
    - Refill the list if getUserMedia does not return NotFoundError
  - Make enumerateDevices return an empty list
    - Provide ways to render different UIs (CSS property?)
- Proposal
  - For REC: status quo, spec allows User Agents to not leak any information
  - Post REC: continue working on tightening improvements

Issue 639 (1/2): Enforcing user gesture for getUserMedia (Youenn)

- Problem
  - User enters a video call
  - User denies camera/microphone access as he/she only expects to listen
  - Later on, user wants to use the microphone to ask a question
- Current solutions are suboptimal
  - Website reloads the page/frame (Safari)
  - Website provides hints to the user to reset permission using browser UI
    - User will then click on a website UI to trigger a new getUserMedia call
- Proposal: If getUserMedia is triggered by a user gesture and permission is 'denied', set permission to 'prompt' and proceed
  - Only for 'temporary' permissions, i.e. permissions that will last for the page
    - Only applicable to Firefox and Safari

Issue 639 (2/2) : Enforcing user gesture for getUserMedia (Jan-Ivar)

Problem A: Prompt-spamming on page-load. **Q:** Is this a real problem for cam+mic? Sites have disincentive to spam due to Chrome's aggressive permanent "Block" option.

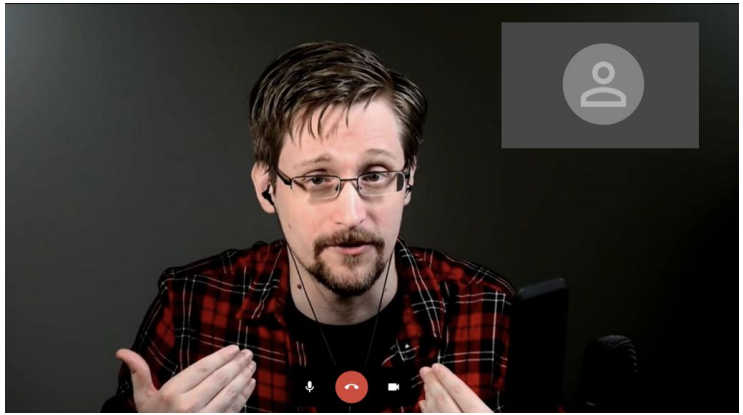
Problem B: Live-cam room-dive. While away, malicious site may navigate user into attacker's room on well-known WebRTC site with persistent permission. Hot cam/mic.

Don't think **A** is a problem. **B** is limited to persistent permissions. Room-dive with prompt only is actually a decent use-flow for invite links (2 clicks to enter room).

User gesture likely not web compatible at this point.

Option 1: Do nothing.
2: Prompt if permission is granted but no user gesture
3: Require user gesture. Let browsers handle migration (e.g using **2** as strategy).
4: Something else? Device-picker ala getDisplayMedia?

Issue 640: Only reveal labels of devices user has given permission to (jib)



Why now?

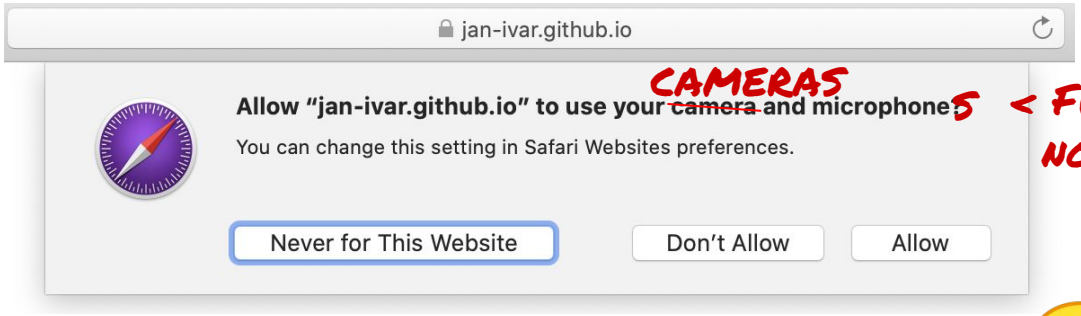
- **PING 2019 review raised this as a problem**
- Privacy climate has changed…

- ...for legitimate reasons:
  - enumerateDevices() 2017 usage 0.07% of all pageloads
  - enumerateDevices() 2019 usage > 4.5% of all pageloads
  - getUserMedia() remains at 0.06%, the rest are trackers

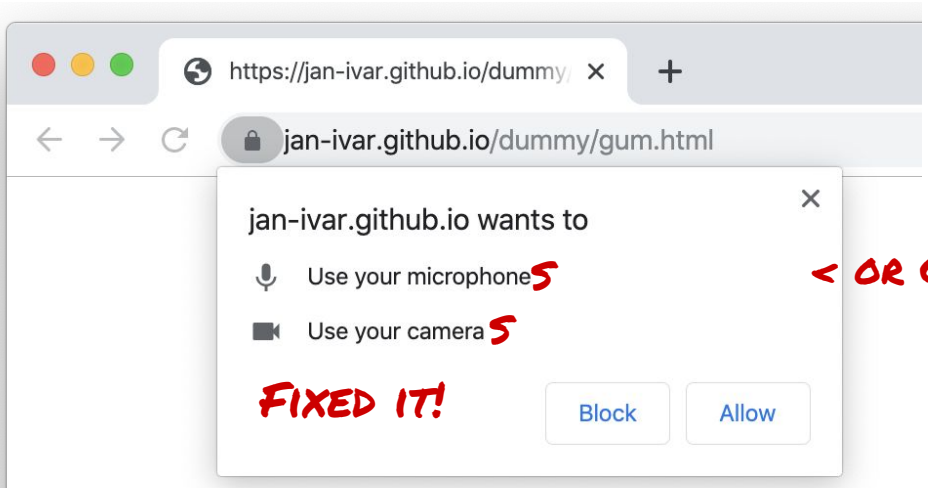
- Exposing info on all of a user's devices beyond the granted one(s) goes beyond fingerprinting to reveal actual private information the user didn't intend to share about what they own and have plugged in.

Issue 640: Only reveal labels of devices user has given permission to (jib)

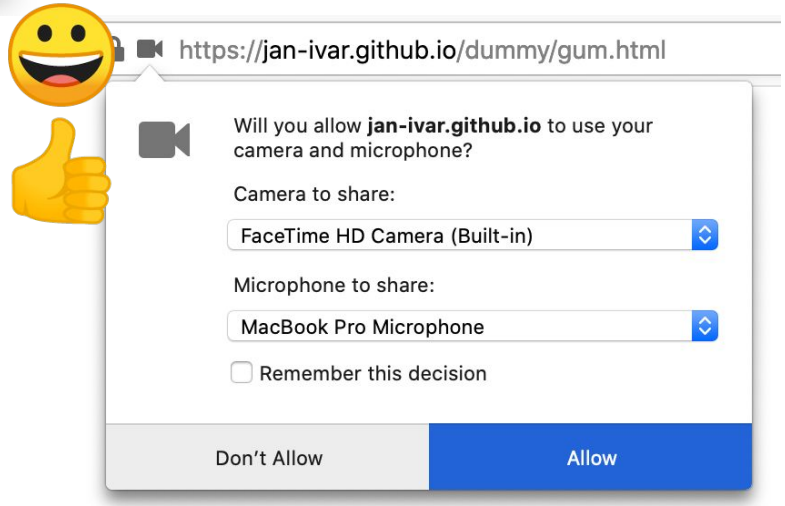
But hold on… ***which*** *"devices has user given permission to"* ? (try it)



< FIXED IT! YOU'RE SHARING ALL DEVICES!
NO MEANINGFUL DIFFERENCE IN SAFARI



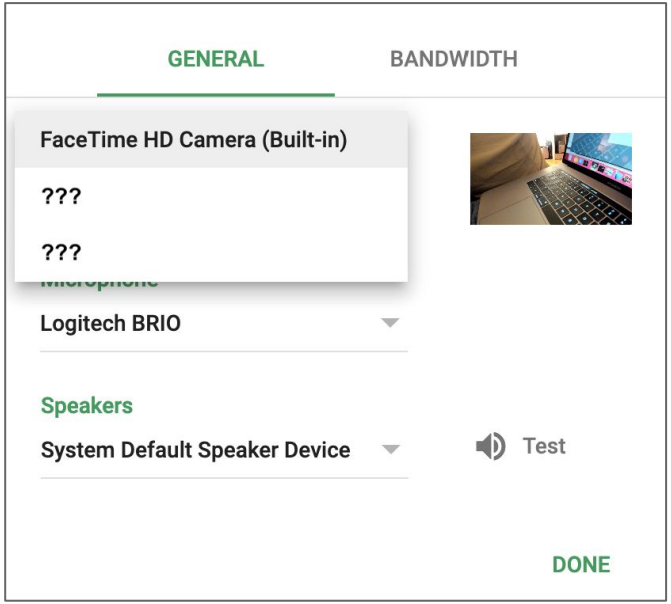
< OR CHROME



Issue 640: Only reveal labels of devices user has given permission to (jib)

Firefox shares all labels *instead of* all devices

Literally, PING request would make only Firefox non-compliant (most private option) forcing it to weaken privacy to comply (grant all devices), or



PING: People can see in the window!
Reality: 3 out of 4 houses are missing a wall.

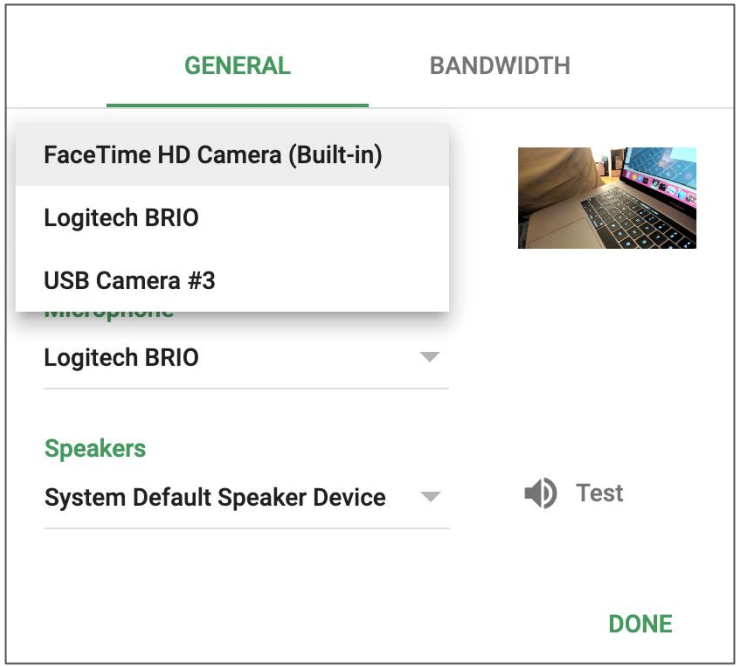
PING wants privacy-by-default flow:

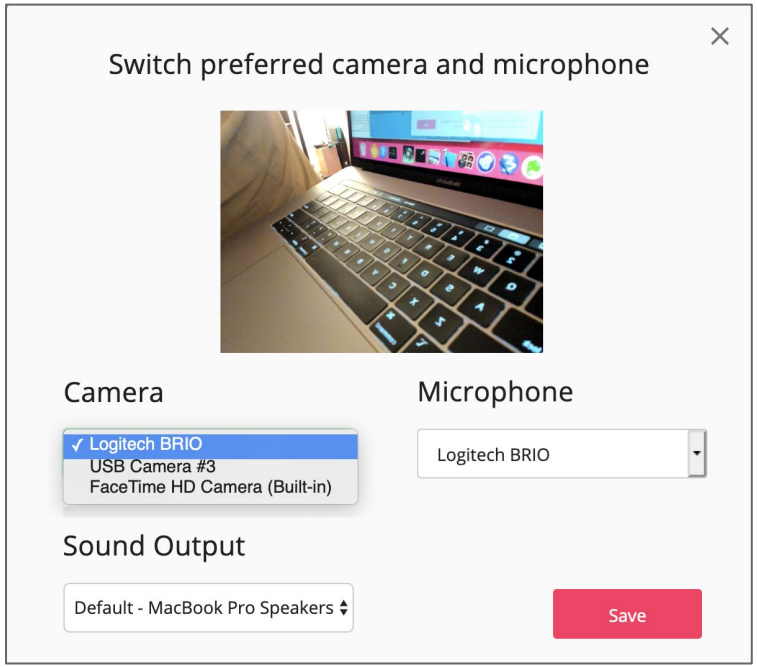
1. site asks for category (or categories) of device
2. *browser prompts user* for one, many or all devices
3. site gains access to only the device + label, of hardware *user selects*.

Firefox mostly does this already by default. Others do not. How'd we get here?

Issue 640: Only reveal labels of devices user has ***chosen*** (jib)

Q: Why do we expose all labels? **A:** To make in-content device selection possible

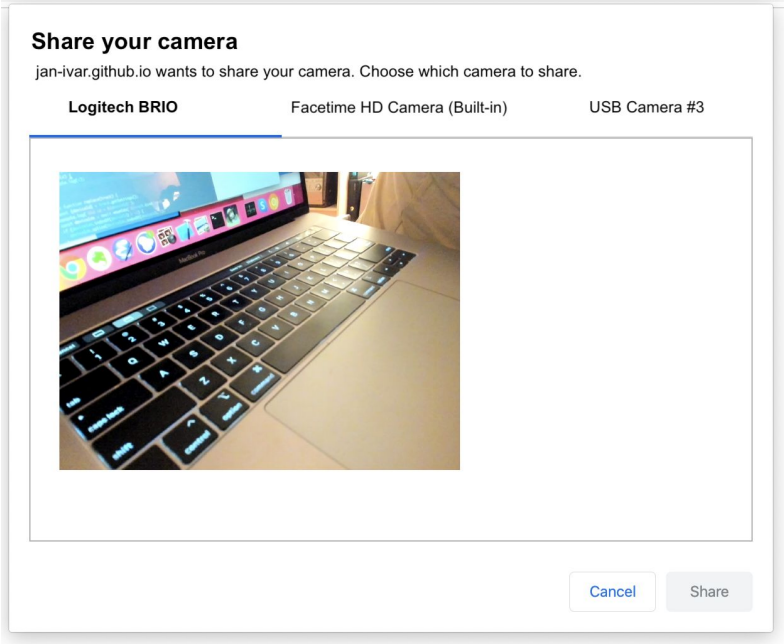


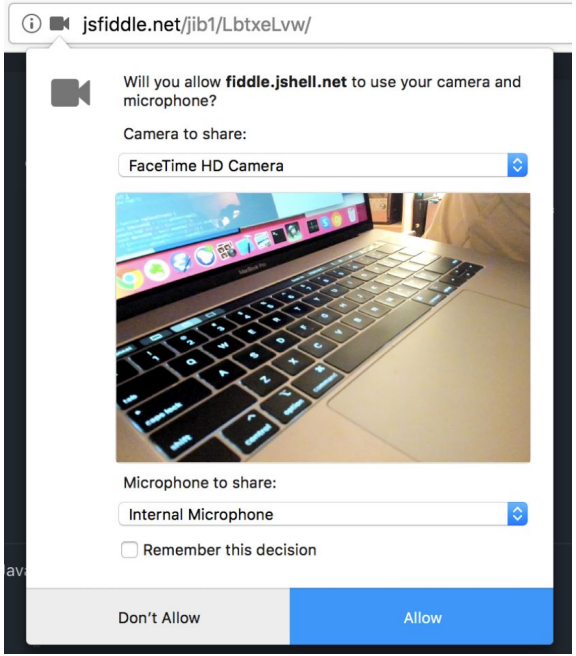


No other reason. Not terribly exciting after 7 years of revealing lots of device info...

Issue 640: Only reveal labels of devices user has ***chosen*** (jib)

In hindsight, this could have been done in-chrome (modeled on success of getDisplayMedia)

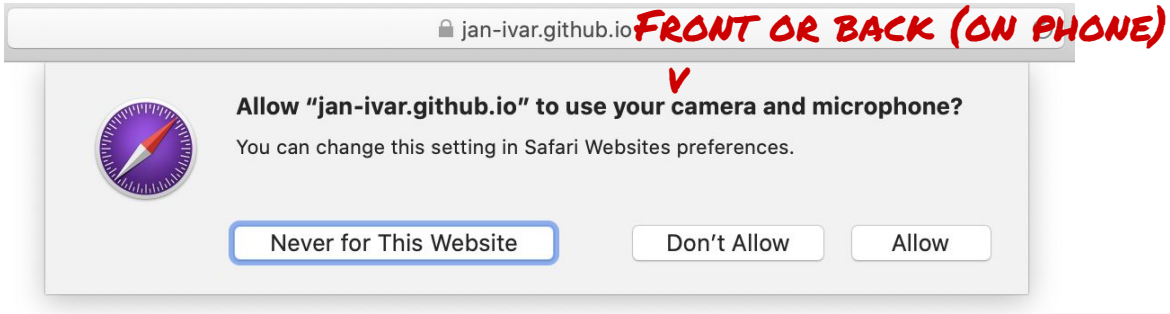


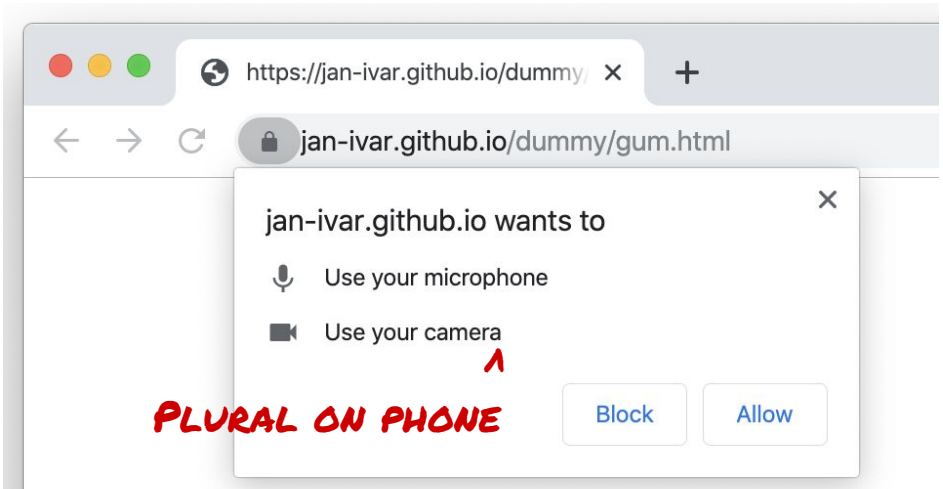


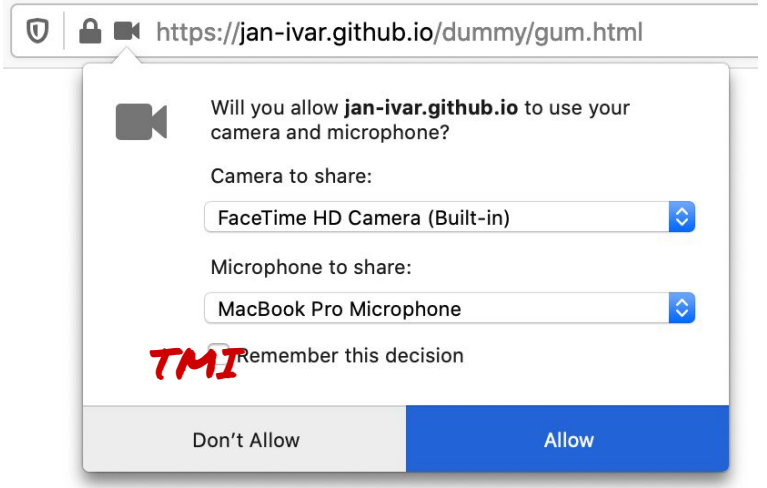
- In-chrome selector removes the need to grant all devices ❤️
- Desktop: UAs can implement previews for all (even non-chosen) cameras (in grid)
- Mobile: UAs can handle tricky platforms that can't open multiple devices (temporarily mute)

[Issue 640](): Only reveal labels of devices user has ***chosen*** (jib)

But most users have **1** camera **or** distinguishable `"user"`/`"environment"` pair

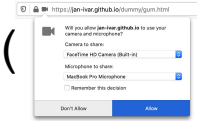






Issue 640 / PR 644: Only reveal labels of devices user has ***chosen*** (jib)

Solution: Use dedicated in-chrome device selector *when* user has multiple devices

Proposal: Mandate selector in gUM() *unless* constraints reduce to 1. *Changes*:

User with previously ***persisted*** permission for camera:	Single camera	Front/back camera	Multiple cameras

`getUserMedia({video: true})`
`getUserMedia({video: {facingMode: "user"}})`
`getUserMedia({video: {deviceId: cameraId}})`
	Granted!	*Selector!*	*Selector!*

`getUserMedia({video: {facingMode: {exact: "user"}}})`	Granted!	Granted!	*Selector!*
`getUserMedia({video: {deviceId: {exact: cameraId}}})`	Granted!	Granted!	Granted!

User *without* persisted permission: replace “Granted!” with “Selector!” or “Prompt!” ()

Issue 640 / PR 644: Only reveal labels of devices user has ***chosen*** (jib)

Proposal (continued): First, remove "live" short-circuit in getUserMedia algorithm:

1. Request permission to use a PermissionDescriptor with its name member set to the permission name associated with *kind* (e.g. "camera" for "video", "microphone" for "audio"), and, optionally, consider deviceId member set to any appropriate device's deviceId,
- while considering all devices attached to a live and same-permission MediaStreamTrack
- in the current browsing context to have permission status "granted",

resulting in a set of provided media

- **Same-permission** in this context means a MediaStreamTrack that required the
- same level of permission to obtain as what is being requested (e.g. not isolated).

Needed, otherwise getUserMedia just returns what you already have, without selector prompt.

Old behavior workaround:

```
const stream1 = await getUserMedia({video: true});
const stream2 = await getUserMedia({video: true}); → const stream2 = stream1.clone();
```

Issue 640 / PR 644: Only reveal labels of devices user has ***chosen*** (jib)

Proposal (continued): Then, enshrine new behavior in the spec:

- 1. Request permission to use a
+ 1. Let *descriptor* be a PermissionDescriptor with its name member set to the permission name associated with *kind* (e.g. "camera" for "video", "microphone" for "audio"), and, optionally, consider its deviceId member set to any appropriate device's deviceId,
+ 2. If the number of unique devices sourcing tracks of media type *kind* in *candidateSet* is 1,
+ then request permission to use the sole device with *descriptor*, resulting in provided media.
+ Otherwise, prompt the user to choose a device with *descriptor*, resulting in provided media.

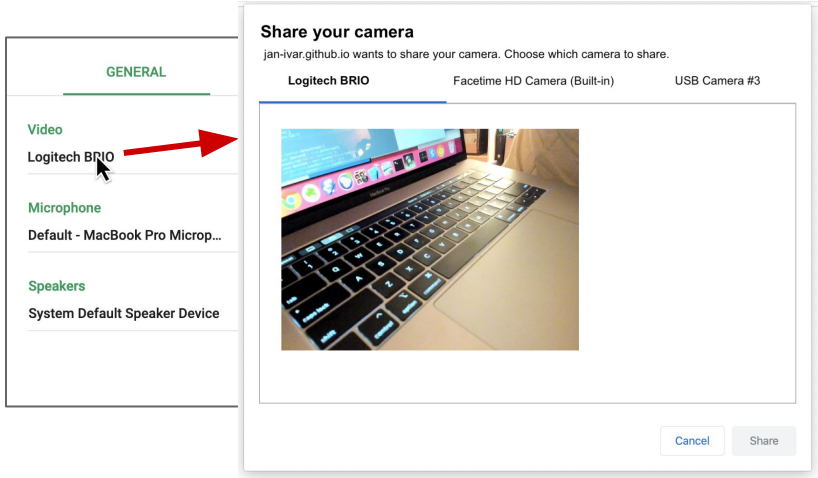
In-content selection can now be replaced with in-chrome selection as follows:

```
camera.innerText = cameraTrack.label;
camera.onclick = async () => cameraTrack = (await getUserMedia({video:
  {deviceId: cameraTrack.getSettings().deviceId}
})).getVideoTracks()[0];
```

Issue 640 / PR 644: Only reveal labels of devices user has ***chosen*** (jib)

Benefits: In-chrome selection lets us:

- Retire labels from `enumerateDevices` (keep `deviceId` & number of devices/kind)
- `track.label` still gives current device label



Feature-detection:

- Labels missing in `enumerateDevices()`

User Agents choices:

- Minor semantic difference between initial prompt, and changing device:
  User agents can detect this from context: s/Share/Change/ + Preview ÷ blocking
- Default choice can be set by passed-in `{deviceId}` in both cases, or excluded from prompt with `{deviceId: {exact: [...allDeviceIdsButOne]}}`

Issue 640: Only reveal labels of devices user has given permis.. (Youenn)

If difficult to enforce per-device exposure rule, enforce per-device-type exposure

- Expose all microphones if one microphone is granted
- Expose all cameras if one camera is granted
- Do not expose speakers once output speaker picker API is available

Still possible to use `groupId` to get microphone corresponding to camera

Issue 642: Only Firefox turns off device on disabled track. Stronger language needed? (Jan-Ivar)

Live cam/mic tracks can be turned on/off with `track.enabled`. Sole use case:

```
muted.onclick = () => { track.enabled = !muted.checked; }
```

Semantically, this is the agreed on abstraction for web sites, regardless of UA permission model. But users expect the privacy of having the camera (or microphone) hardware light off when off.

Spec agrees[1]: ”when a track becomes either muted or disabled, and this brings all tracks connected to the device to be either muted, disabled, or stopped, then the UA *MAY*, using the device's deviceId, deviceId, set [[devicesLiveMap]][deviceId] to false”

Google Hangouts agrees, but instead adds (cam only) hack to make it work in Chrome (stop + re-gUM). This backfires with needless re-prompt on unmute in Firefox.

Firefox agrees, and implements the spec advice. See blog Better privacy on camera mute

1) The spec doesn’t actually mention hardware, only “privacy indicators”. However, this omission seems consistent with the overall design of this spec (e.g. constraints, mute/unmute events etc.) where user agents are left to manage hardware however they best see fit within the constraints outlined by the API. Having physical and logical indicators align seems a reasonable end-user expectation; the spec notably does not say this intuitive interpretation is forbidden.

Issue 642: Only Firefox turns off device on disabled track. Stronger language needed? (Jan-Ivar)

How Firefox works (fiddle):

1. Relinquishes device when all its tracks are disabled.
2. Reacquires device when any of its tracks are re-enabled (takes ~1 second to reacquire device)
3. Failure to reacquire fires ended event on track(s).
4. *Live* indicator always on for 3 seconds minimum (*"MUST remain observable for a sufficient time"*)
5. *Accessible* mandated Privacy Indicator remains in URL bar while muted.
6. Privacy mitigation plan is to have Firefox fire muted event if re-enabled without focus (Bug 1598374)

Option A: Stronger language to enforce web compat around this behavior in all browsers
Option B: Drop enabled
Option C: Do nothing (leaves web sites having to feature detect to avoid both light and prompt)
Option D: We think existing language is strong enough. File bugs on vendors.

For extra credit

Name that bird!

Thank you

Special thanks to:

WG Participants, Editors & Chairs

The bird