W3C WebRTC WG Meeting

May 22, 2018
1 PM Pacific Time

Chairs: Stefan Hakansson
Bernard Aboba
Harald Alvestrand

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:
  - Go over the agenda for the F2F meeting in Stockholm
  - Provide an update on the status of the KITE test framework
  - Make progress on open issues in webrtc-pc
- Will recycle WebRTC 1.0 at CR after the meeting

June f2f (Stockholm, Sweden)

Meeting scheduled for June 19-20, 2018 at the Google Stockholm offices.

TPAC (Lyon, France)

We will meet Monday October 22 and Tuesday the 23rd of TPAC week.

Proposed Agenda for June 19

9 AM - 10:30 AM: WebRTC NV use cases and requirements

10:30 AM - 11 AM: Break

11:00 AM - noon AM: Directional questions

a. How low-level of an API do we want for RTP, ICE and media?
b. Will we support media/data over QUIC, RTP, or both?
c. Will we use ReadableStream/WritableStream?

Noon - 1 PM working lunch: ICE

a. ORTC ICE API (Bernard Aboba)
b. WebRTC-ICE proposal (Peter Thatcher)
c. SLICE transport proposal (Peter Thatcher)

1 PM - 2 PM: Certificates and Identity (Bernard and Harald)

2 PM - 3 PM: Data Transfer (Lennart Grahl and Peter Thatcher)

3 PM - 5 PM: Interoperability and WebRTC 1.0 Testing (Dr. Alex and team)

Proposed Agenda for June 20

9 AM - 10 AM: Access to Raw Media (Harald)

10:00 AM -11:00 AM: Scalable Video Coding (Sergio and Bernard)

11:00 AM - Noon: Sender/Receiver (Peter Thatcher)

Noon - 1 PM Working Lunch: Web Workers (Harald)

1 PM - 2 PM: Open slot

2 PM - 3 PM: Open slot

3 PM - 5 PM: Wrapup and next steps

WebRTC WG Charter (Chairs)

- Current charter has been extended until May 2018.
- Draft Charter: https://w3c.github.io/webrtc-charter/webrtc-charter.html
  - Extends charter to March 31, 2020.
- Charter is out for AC review (until May 25) - make sure your AC rep responds!!

About this Virtual Meeting

Information on the meeting:

- Meeting info:
  - https://www.w3.org/2011/04/webrtc/wiki/May 22_2018
- Link to latest drafts:
  - https://w3c.github.io/mediacapture-main/
  - https://w3c.github.io/webrtc-pc/
  - https://w3c.github.io/mediacapture-screen-share/
  - https://w3c.github.io/webrtc-stats/
- Link to Slides has been published on WG wiki
- Scribe? IRC http://irc.w3.org/ Channel: #webrtc
- The meeting is being recorded.
- WebEx info here

For Discussion Today

- **KITE Update (Dr. Alex)**
- **WebRTC-PC**
  - **Issue 1834: Handling of invalid values in sender.setParameters (AdamBe)**
  - **Issue 1839: What should sender.getParameters().codecs return prior to negotiation? (Taylor)**
  - **Issue 1840: What should receiver.getParameters() return? (Bernard)**
  - **Issue 1852: RTCRtpDecodingParameters (Bernard)**
  - **Issue 1858: What happens when an answerer stops a transceiver that others are “bundled” on? (Taylor)**
  - **Issue 1872: sendEncodings in “create an RtpSender” should reflect platform capabilities (Harald)**

KITE Update (Dr. Alex)

- Slides

For Discussion Today

- **WebRTC-PC**
  - **Issue 1834: Handling of invalid values in sender.setParameters (AdamBe)**
  - **Issue 1839: What should sender.getParameters().codecs return prior to negotiation? (Taylor)**
  - **Issue 1840: What should receiver.getParameters() return? (Bernard)**
  - **Issue 1852: RTCRtpDecodingParameters (Bernard)**
  - **Issue 1858: What happens when an answerer stops a transceiver that others are “bundled” on? (Taylor)**
  - **Issue 1872: sendEncodings in “create an RtpSender” should reflect platform capabilities (Harald)**

Issue 1834: Handling of invalid values in sender.setParameters (AdamBe)

- Reflections
  - We don't use unsigned types consistently
  - We don't need to check the upper boundary of max-type values
- Checked (partly)
  - `scaleResolutionDownBy: RangeError` if < 1; Make unsigned? Check upper boundary?
- Currently unchecked
  - `maxFramerate:` Is double; Make unsigned? Check lower boundary?
  - `maxBitrate:` Is unsigned long; Check lower boundary?
  - `ptime:` Is unsigned long; Specified to respect maxptime; Check boundaries?

Issue 1839: What should sender.getParameters().codecs return prior to negotiation? (Taylor)

- The current text says:
    - “The codecs sequence is populated based on the codecs that have been negotiated for sending, and which the user agent is currently capable of sending.”
- This seems to indicate that prior to negotiation completing, the set should be empty. Is this intentional, and reasonable?
- Proposed answer: yes and yes.
    - If you want to know the codecs the browser is capable of (and thus would offer) you can use `RTCRtpSender.getCapabilities`.
    - `getParameters` is for what is available to send, so the application can pick one and put its payload type in `RTCRtpEncodingParameters`.
    - Until a remote description is set, you don't know what payload types they'll use, so it wouldn't make sense to return anything.

Issue 1840: What should receiver.getParameters() return? (Bernard)

- Section 5.3 states:
  - The `getParameters` method returns the `RTCRtpReceiver` object's current parameters for how track is decoded. When `getParameters` is called, the `RTCRtpReceiveParameters` dictionary is constructed as follows: `encodings` is populated based on RIDs present in the current remote description, *and whether media is actively being decoded or not.* The `headerExtensions` sequence is populated based on the header extensions that the receiver is currently prepared to receive. The `codecs` sequence is populated based on the codecs that the receiver is currently prepared to receive.
- Note: the `active` attribute is not present in `RtpReceiveParameters.encodings[]`. Should we delete the redlined text?
- Does the text imply that `headerExtensions` and `codecs` can be populated prior to completion of negotiation (after SLD, but before SRD)?

Issue 1852: RTCRtpDecodingParameters (Bernard)

- Section 5.1 states:
  - This specification does not define how to configure `createOffer` to receive multiple RTP encodings. However when `setRemoteDescription` is called with a corresponding remote description that is able to send multiple RTP encodings as defined in [JSEP], the `RTCRtpReceiver` may receive multiple RTP encodings and the parameters retrieved via the transceiver's `receiver.getParameters()` will reflect the encodings negotiated.
- Do we expect browsers to implement this (optional) feature?
- Note: only the `rid` attribute is included in the `RtpDecodingParameters` dictionary:
-
```
dictionary RTCRtpReceiveParameters : RTCRtpParameters {
    required sequence<RTCRtpDecodingParameters> encodings;
};
```
-
```
dictionary RTCRtpCodingParameters {
    DOMString rid;
};
```
-
```
dictionary RTCRtpDecodingParameters : RTCRtpCodingParameters {
};
```

Issue 1852: RTCRtpDecodingParameters (cont'd)

- Assuming we expect any implementations:
  - Why is it useful for `RtpDecodingParameters` to include the `rid` attribute and nothing else?
    - Since there are no other attributes (e.g. no payload type, SSRCs, etc.) if more information is needed, the application will need to parse the SDP anyway.
  - Since the `rid` attribute is not ***required***, if the browser does not support simulcast reception, the `rid` attribute is presumably unset?

Issue 1858: What happens when an answerer stops a transceiver that others are "bundled" on? (Taylor)

- If there's an established bundle group:
  - `a=group:BUNDLE audio video`
- The answerer can't reject the "audio" m= section without rejecting "video" as well, due to limitations of the BUNDLE spec.
- So, what happens if the answerer calls `audioTransceiver.stop()`? Either:
  - Both m= sections are rejected (both transceivers end up stopped).
  - Answerer temporarily keeps the audio m= section alive, as "`a=inactive`", and rejects it in a subsequent offer/answer.
- Either way, may require JSEP changes?

Issue 1872: sendEncodings in “create an RtpSender” should reflect platform capabilities (Harald)

- Known issue: How many encodings can we send?
  - Ever, or in the current negotiation state?
- Known issue: What to do when user asks for # higher than can be sent?
- Proposal: Do something that makes sense
  - Set # of encodings to max # that can be sent
    - If max # changes over time, reduce (assumption: never increase)
  - Enable only the first one by default, more if user asks
  - Ignore requests for more layers than available
- Probably simple to spec, implement and test. Probably addresses basic use cases.

For extra credit

Name that bird!

Thank you

Special thanks to:

W3C/MIT for WebEx

WG Participants, Editors & Chairs

The bird