rtcweb
Internet-Draft
Intended status: Standards Track
Expires: July 9, 2015

K. Guduru
Samsung Electronics

January 5, 2015

# WebRTC Codec Preferences
# draft-guduru-rtcweb-codec-preferences-02

Abstract

WebRTC specifies to implement a minimum set of required codecs to ensure guaranteed interoperability between WebRTC peers. WebRTC allows browser implementers to support vendor specific codecs apart from mandatory codecs. This document specifies the way for JavaScript applications to give preferences for media codecs in WebRTC context out of the available codecs in browser for creating the offer / answer.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 9, 2015.

Table of Contents



## 1. Introduction

WebRTC specifies to implement mandatory codecs inside the browser to ensure guaranteed interoperability between browsers and to increase the session establishment success rate [I-D.ietf-rtcweb-audio]. WebRTC specifications allows browser implementors to support codecs apart from mandatory codecs. This specification describes a mechanism for JavaScript application to check the available codecs, to set preferences among the available codecs and provision to remove codecs from the SDP, if the application don't want to use them. This specification extends RTCPeerConnection [I-D.ietf-rtcweb-jsep] by adding two new methods, getSupportedAudioCodecs and getSupportedVideoCodecs for retrieving codecs supported by browser, RTCRTPSender and RTCRTPReceiver for setting the codec preferences and for retrieving the previously configured codec preferences of accepted tracks. It also extends offer / answer options with sequences of audio and video codec list to set codec preferences for accepted tracks without codec preferences set and for future tracks. These preferences are used for preparing the SDP when subsequent createOffer or createAnswer is called. This specification fulfils the requirement to provide API for web application to control media format, API requirement A5 specified in

[I-D.ietf-rtcweb-use-cases-and-requirements], and avoids the SDP mangling for removing and re-ordering of codecs.

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

## 3. Retrieving Supported Codecs

This specification allows JS application to retrieve audio and video codecs supported by the browser.

### 3.1. getSupportedAudioCodecs

The getSupportedAudioCodes method of RTCPeerConnection returns a sequence of audio codecs supported by browser. This sequence SHOULD solely contain all the audio codecs that are supported by browser. This function can be called from the JavaScript application at any time after creating the peerConnection and before closing the peerConnection.

### 3.2. getSupportedVideoCodecs

The getSupportedVideoCodes method of RTCPeerConnection returns a sequence of video codecs supported by browser. This sequence SHOULD solely contain all the video codecs that are supported by browser. This function can be called from the JavaScript application at any time after creating the peerConnection and before closing the peerConnection.

## 4. Retrieving Codec Preferences

This specification provides an API for Java Script application to retrieve the list of codec preferences that were previously set.

### 4.1. getCodecPreferences

The getCodecPreferences method of RTCRTPSender / RTCRTPReceiver returns a sequence of codecs, if set earlier. The order of codecs in this sequence SHOULD be same as that previously set by the application. The sequence contains a list of audio codecs, if the MediaStreamTrack of RTCRTPSender / RTCRTPReceiver is of type audio. The sequence contains a list of video codecs, if the MediaStreamTrack of RTCRTPSender / RTCRTPReceiver is of type video.

## 5. Setting Codec Preferences

This specification provides API for Java Script application to set the sequence of codec preferences. The order of codecs in this sequence SHOULD be in the order preferred by the JavaScript application. Codec preferences can be set through RTCRTPSender / RTCRTPReceiver or through RTCOfferAnswerOptions.

### 5.1. setCodecPreferences

The setCodecPreferences method of RTCRTPSender / RTCRTPReceiver sets the codec preferences for accepted tracks. The sequence of codecs SHOULD be in the order specified by Java script application. The sequence of codecs MUST be of kind audio, if the MediaStreamTrack of RTCRTPSender / RTCRTPReceiver is of type audio. The sequence of codecs MUST be of kind video, if the MediaStreamTrack of RTCRTPSender / RTCRTPReceiver is of type video.

### 5.2. RTCOfferAnswerOptions

createOffer and createAnswer methods supports optional "RTCOfferAnswerOptions" attribute. This specification extends RTCOfferAnswerOptions with optional preferredAudioCodecs and preferredVideoCodecs attributes, to set the audio and video codec preferences respectively. preferredAudioCodecs and preferredVideoCodecs, if present in RTCOfferAnswerOptions, contains a sequence of audio and video codecs in applications preferred order. If the application does not want to specify any codec preference, then these attributes MUST not be added to RTCOfferAnswerOptions.

#### 5.2.1. preferredAudioCodecs

preferredAudioCodecs, is an optional constraint which SHOULD contain a sequence of audio codecs in applications preferred order.

#### 5.2.2. preferredVideoCodecs

preferredVideoCodecs, is an optional constraint which SHOULD contain a sequence of video codecs in applications preferred order.

## 6. Generating SDP

JavaScript application can set the codec preferences in browser through setCodecPreferences method or RTCOfferAnswerOptions. The setCodecPreferences will not immediately change the existing codec in encoder or decoder of accepted tracks. It preserves the codec preferences until createOffer / createAnswer method is invoked for

the next time, and considers updated codec preferences for generating the new SDP.

The preferred codec sequence MAY contain the same number of audio codecs or video codecs returned by getSupportedAudioCodecs / getSupportedVideoCodecs respectively. Sequence of codecs, preferred by the application, MUST contain at least one codec returned by getSupportedAudioCodecs / getSupportedVideoCodecs. Codecs other than those supported by browser SHOULD be ignored, if present in sequence of codecs that were set by the application. The offer / answer SHOULD NOT contain audio codecs other than those specified by JavaScript application and the order of preference SHOULD be with high priority for the codecs first in the sequence.

For generating the SDP, codecs SHOULD be prioritized as per the preferences set through setCodecPreferences. If preferences were not set through setCodecPreferences, codecs SHOULD be prioritized as per the preferences set through RTCOfferAnswerOptions. If the preferences were not set either through setCodecPreferences or RTCOfferAnswerOptions, then SDP will be generated with implementation specific codecs preferences.

## 7. Usecases

Many codecs are available for encoding and decoding audio, video data with their own advantages and disadvantages. Applications MAY use any of the available codecs in browser based on its requirements like providing high quality, reduced bandwidth usage, reduced power usage. The requirements for prioritizing a codec MAY change after establishing the session.

Bandwidth availability may not be same for a complete session. User may increase or decrease the bandwidth provided, by communicating with network provider, while the session is ongoing. The bandwidth availability may change because of network congestions. Some codecs performs pretty good at higher bandwidths while some other codecs perform well at lower bandwidths. The performance of single codec at all the bandwidth ranges is not same, which demands to use different codecs at different bandwidth availability.

Battery usage is a big concern in mobile devices. Some codecs may use less power with the tradeoff of bandwidth / quality. At times of less available battery charging, users may like to continue the conversation for more time with slightly reduced quality, leading to a requirement to change the priority of codecs during an active session.

## 8. Example

Consider an example where a browser implementation supports G.722 and AMR-WB apart from OPUS and G.711 as audio codecs. Let the browser default order of priority for audio codecs be G.722, AMR-WB, OPUS and G.711. Then the partial SDP representing audio codecs, that is generated as a result of createOffer or createAnswer will be as follows.

```
m=audio 1 RTP/SAVPF 103 109 111 0 8 126
a=mid:audio
a=rtpmap:103 g722/8000
a=rtpmap:109 AMR/8000/1
a=rtpmap:111 opus/48000/2
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:126 telephone-event/8000
```

If the application wants to modify the audio codec preference to AMR-WB, OPUS, G.722 and G.711, then JavaScript application will retrieve the supported codecs and set the codec preferences accordingly. Then the partial SDP representing audio codecs, generated through createOffer / createAnswer will generate the SDP with the codecs in application's preferred order.

```
m=audio 1 RTP/SAVPF 109 111 103 0 8 126
a=mid:audio
a=rtpmap:109 AMR/8000/1
a=rtpmap:111 opus/48000/2
a=rtpmap:103 g722/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:126 telephone-event/8000
```

## 9. Security Considerations

TBD.

## 10. IANA Considerations

This document requires no actions from IANA.

## 11. Acknowledgements

I would like to thank Justin Uberti and Harald Alvestrand for their valuable comments and suggestions.

## 12. Normative References

[I-D.ietf-rtcweb-audio]
Valin, J. and C. Bran, "WebRTC Audio Codec and Processing Requirements", draft-ietf-rtcweb-audio-05 (work in progress), February 2014.

[I-D.ietf-rtcweb-jsep]
Uberti, J. and C. Jennings, "Javascript Session Establishment Protocol", draft-ietf-rtcweb-jsep-06 (work in progress), February 2014.

[I-D.ietf-rtcweb-use-cases-and-requirements]
Holmberg, C., Hakansson, S., and G. Eriksson, "Web Real-Time Communication Use-cases and Requirements", draft-ietf-rtcweb-use-cases-and-requirements-14 (work in progress), February 2014.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

Author's Address

Kiran Kumar Guduru
Samsung Electronics
Samsung RnD Institute India - Bangalore
Bangalore, Doddanakundi 560037
India

Phone: +91-888-4995166
Email: kiran.guduru@samsung.com