Google

Web API Developments: MSE, EME and MediaCapabilities

Mile High Video 2020

Matt Wolenetz, MSE spec editor
Joey Parrish, EME spec editor
Chris Cunningham, MediaCapabilities spec editor

Media Source Extensions

Media Source Extensions

- W3C Recommendation since November 2016
- Incubations coming soon to editor's draft:
  - SourceBuffer.changeType() (shipped in Chrome, FF, Safari)
  - MSE-in-Workers (experimental in Chrome 88)
- Tentatively coming soon:
  - Integration with WebCodecs
  - Eviction policy API
  - Gap handling API
- ~19 open issues tentatively in scope for MSE v2

Media Source Extensions - SourceBuffer.changeType()

```
const typeA = 'video/webm; codecs="vp9"';
const typeB = 'video/mp4; codecs="avc1.4d001e"'
const mediaSource = new MediaSource();

// Elided: isTypeSupported(), attachment, objectUrl revocation, media fetching,
// awaiting async appendBuffer() completions, etc. steps.

function onSourceOpenedFirstTime() {
  sourceBuffer = mediaSource.addSourceBuffer(typeA);
}

// Later:
sourceBuffer.appendBuffer(initAndMediaForTypeA);

// Later:
sourceBuffer.changeType(typeB);
sourceBuffer.appendBuffer(initAndMediaForTypeB);
```

Media Source Extensions in DedicatedWorkers

```
if (MediaSource.canConstructInDedicatedWorker) {
  const worker = new Worker('worker.js');
  worker.onmessage = (url) => {
    video.src = url;
  };
}

// worker.js
const mediaSource = new MediaSource();
const objUrl = URL.createObjectUrl(mediaSource);
mediaSource.addEventListener('sourceopen', () => {
  URL.revokeObjectUrl(objUrl);

  // addSourceBuffer(...), appendBuffer(...), etc.
}, { once: true });

postMessage(objUrl);
```

Media Source Extensions - links

- Codec Switching Explainer: **https://tinyurl.com/mse-codec-switching**
- MSE-in-Workers explainer: **https://tinyurl.com/mse-in-workers**
- MSE-in-Workers demo: **https://tinyurl.com/mse-in-workers-demo**
- MSE + WebCodecs issue: **https://tinyurl.com/mse-plus-webcodecs**
- Eviction Policies explainer: **https://tinyurl.com/mse-eviction-policies**
- Gap handling issue: **https://tinyurl.com/mse-gap-handling**

- Issue Tracker: **https://github.com/w3c/media-source/issues**

Encrypted Media Extensions

Encrypted Media Extensions

- W3C Recommendation since September 2017
- Editor's draft adds:
  - Encryption scheme queries
  - Session type: Persistent Usage Record
  - Key status: Usable In Future
- Proposed, not landed:
  - HDCP status detection
  - Handle hardware context resets, surprise session closures
- Proposals needed:
  - 5 different(-ish) issues about improving key rotation & interop of it

Encrypted Media Extensions - Persistent Usage Records

- New session type (actually, it's old, but it's back)
- Persists a record of key usage (first and last decryption time)
- Sends this record back to the license server when remove() is called
- Does not persist keys

Encrypted Media Extensions - Usable-In-Future Key Status

- New key status
- Keys that will only be valid at some future time
- The polar opposite of expired
- Maps to Android's STATUS_USABLE_IN_FUTURE in Android Q

Encrypted Media Extensions - HDCP Status

- Query the key status that would result from HDCP policy requirements
- Start fetching the right content sooner
- Shipped in Chrome 73

Encrypted Media Extensions - HDCP Status

```
const keyStatus = await mediaKeys.getStatusForPolicy({
  minHdcpVersion: 1.0,
});

if (keyStatus == 'usable') {
  // Start fetching streams that require HDCP 1.0 to decrypt.
} else {
  // Start fetching lower-quality streams that don't require HDCP.
}
```

Encrypted Media Extensions - Encryption Scheme Queries

- Find out what encryption schemes are supported
- *No more guessing!*
- Shipped in Chrome 81

- "**cenc**", "**cbcs**" (any skip pattern), and "**cbcs-1-9**" (just 1:9 pattern)

Encrypted Media Extensions - Encryption Scheme Queries

- Find out what encryption schemes are supported
- *No more guessing!*
- Shipped in Chrome 81

- "**cenc**", "**cbcs**" (any skip pattern), and "**cbcs-1-9**" (just 1:9 pattern)

- Polyfill available:
  - **eme-encryption-scheme-polyfill** on NPM
- *Does the guessing for you!*
- *Pretend customer devices get platform updates!*
- *1.5kB gzipped!*

Encrypted Media Extensions - Encryption Scheme Queries

```
const mksa = await navigator.requestMediaKeySystemAccess(keySystem, [{
  videoCapabilities: [{
    contentType: 'video/mp4; codecs="avc1.640028"',
    encryptionScheme: 'cenc',
  },{
    contentType: 'video/mp4; codecs="avc1.640028"',
    encryptionScheme: 'cbcs-1-9',
  }]
}]);

const caps = mksa.getConfiguration().videoCapabilities;
if (caps.some(cap => cap.encryptionScheme == 'cenc')) {
  // Choose "cenc" content URL
} else {
  // Choose "cbcs-1-9" content URL
}
```

MediaCapabilities

MediaCapabilities (and a little CSS)

- W3C Public Working Draft since January 2020
- New since MHV 2019
  - EME decoding capability queries (launched in Chrome)
  - HDR decoding capability queries (Chromium impl in progress)
  - CSS Media Queries 5 spec added 'dynamic-range'
- Tentatively landed:
  - video-* prefixed CSS Media Queries (for TVs with distinct video and graphics planes)

MediaCapabilities + EME (Launched in Chromium 80)

```
const encryptedConfig = {
  type: 'media-source',
  audio: { ... },
  video: { ... },
  keySystemConfiguration: {
    keySystem: 'com.widevine.alpha',
    videoRobustness: 'SW_SECURE_DECODE', // Widevine L3
  },
};

navigator.mediaCapabilities.decodingInfo(encryptedConfig).then(result => {
  console.log('This media configuration is:\n' +
      (result.supported ? '' : ' NOT') + ' supported and' +
      (result.smooth ? '' : ' NOT') + ' smooth and' +
      (result.powerEfficient ? '' : ' NOT') + ' power efficient.');

  // Now use `result.keySystemAccess.createMediaKeys()` to setup EME
});
```

HDR Capabilities (Spec done, Chromium launch 2021)

```
// Is the screen HDR?
if (!window.matchMedia('(dynamic-range: high)').matches)
    return;

// Does the UA understand the transfer function, color gamut, and metadata?
navigator.mediaCapabilities.decodingInfo({
  type: 'media-source',
  video: {
    contentType: 'video/webm; codecs="vp09.00.10.08"',
    width: 1920,
    height: 1080,
    bitrate: 2646242,
    framerate: '25',
    hdrMetadataType: 'smpteSt2086',
    colorGamut: 'p3',
    transferFunction: 'pq',
  }
});
```

Questions?