WebPerf WG @ TPAC 2020

bit.ly/webperf-tpac20

Logistics

When

October 19-22 2020 - 9am-12pm PST

Registering

- Register!
- Free for WG Members and Invited Experts
- If you're not one and want to join, ping the chairs to discuss

Calling in

Video call link

Attendees

- Yoav Weiss (Google)
- Nic Jansma (Akamai)
- Nicolás Peña Moreno (Google)
- Michal Mocny (Google)
- Cliff Crocker (SpeedCurve)

- Ian Clelland (Google)
- Benjamin De Kosnik (Mozilla)
- Sean Feng (Mozilla)
- Noam Rosenthal (Invited expert)
- Steven Bougon(Salesforce)
- Noam Helfman (Microsoft)
- Andrew Comminos (Facebook)
- Nathan Schloss (Facebook)
- Nicolas Dubus (Facebook)
- Patrick Hulce (Invited expert)
- Scott Haseley (Google)
- Nicole Sullivan (Google)
- Matt Falkenhagen (observer) (Google)
- Andrew Galloni (Cloudflare)
- Ulan Degenbaev (Google)
- Kinuko Yasuda (Google)
- Patrick Meenan (Invited expert)
- Carine Bournez (W3C)
- Subrata Ashe (Salesforce)
- Alex Christensen (Apple)
- Thomas McCabe (observer) (Squarespace)
- Nolan Lawson (Salesforce)
- Dinko Bajric (Salesforce)
- Noah Lemen (Facebook)
- Tim Dresser (Google)
- Gilles Dubuc (Wikimedia Foundation)
- Jeremy Roman (observer) (Google)
- Ian Vollick (observer) (Google)
- Utkarsh Goel (Akamai)
- Ziran sun (Igalia)
- Domenic Denicola (Google)
- Boris Schapira (invited expert, Dareboost)
- Arno Renevier (Facebook)
- Alex Russell (Google, observer)
- Ryosuke Niwa (Apple)
- Camille Lamy (Google)

Agenda

Times in PT

Monday - October 19

Timeslot	Subject	POC
9:00-9:30	**Intros, code of conduct, agenda review, meeting goals**	Yoav, Nic
09:30-10:00	**Frame Timing / smoothness reporting**	Michal
10:00-10:30	**SpeedCurve hot topics**	Cliff
10:30~10:45	Break	
10:45-11:30	**Scheduling APIs**	Scott
11:30-12:00	**Interactions and Event Timing**	Nicolás

Recording: part 1, part 2

Tuesday - October 20

Timeslot	Subject	POC
9:00-10:00	**Prerendering [Page Visibility] [pre* with author support]**	David, Jeremy
10:00~10:05	Break	
10:00-10:55	**SPA reporting**	Michal
10:55~11:00	Break	
11:00-11:30	**BFCache and performance entries**	Yoav
11:30-12:00	**isInputPending update**	Andrew

Recording: part 1, part 2, part 3

Wednesday - October 21

Timeslot	Subject	POC
9:00-9:30	**Network Diagnostics API Proposal**	NoamH
09:30-10:00	**performance.measureMemory API update**	Ulan

10:00-10:30	**Rechartering, Call for Editors**	Yoav, Nic
10:30~10:45	Break	
10:45-11:30	**Long Tasks attribution**	Patrick H
11:30-12:00	**JS self-profiling update**	Andrew

Recording: part 1, part 2

Thursday - October 22

Timeslot	Subject	POC
9:00-10:00	**TAO and CORS/CORP opt-ins**	Yoav
10:00-10:30	**Reporting API updates**	Ian
10:30~10:45	Break	
10:45-11:30	**Reporting API and performance metrics**	Yoav
11:30-12:00	Overflow time	

Recording: part 1, part 2

Secondary Scribes

- Michal Mocny

Session summary

Intros, code of conduct, agenda review, meeting goals

Video, minutes

Smoothness Reporting for Animations and Scrolling - Michal Mocny

Video, minutes

- In this session we outlined the goals for a potential future Smoothness / Frame Timing API for Animations (including scrolling) on the web.

- We introduced the idea of focusing on “Missed opportunities to show expected animation updates” (aka “dropped frames”) which is a more user focused measure of impact, vs just focusing on overall FPS broadly. We also discussed briefly what exactly would constitute an animation, “dropped frame”, and how smoothness relates to responsiveness.
- We also outlined several different options for surfacing the data via API (i.e.: per animation, per frame, or as a single page-level summary)
- Some topics discussed:
    - “variable refresh rate” monitors, where the decision to increase screen refresh rate could be driven by the performance of animations on the page in the first place. How should we balance optimizing for “expected frames”?
    - Should we have smoothness targets? Perhaps some applications want to reach the maximum supported device refresh rate, while others this is less critical.
    - Learning from the video gaming industry.
    - Existing metrics RUM providers track an: #rAF’s over time (however, not super useful right now)
- Next steps:
    - Address questions raised (especially around variable refresh rates)
    - Move proposal to github for more feedback, and open a WICG issue

SpeedCurve hot topics - Cliff Crocker

Video, minutes

- In this session we discussed the challenges and opportunities for Core Web Vitals, including how we can work to normalize metrics across synthetic and RUM that are skewed due to the measurement of the page lifecycle.
- We also discussed issues with reporting the performance of third-parties, specifically ads and the urgency of several proposed issues related to measurement of same origin and cross-origin frames.
- The topic of server timing was also briefly raised, specifically how we should be encouraging others to adopt this (CDNs as unewell as developers) in an effort to provide a better understanding of what is impacting TTFB.
- Next steps:
    - Look at how we can effectively leverage Reporting API to more accurately measure load-limited metrics.
    - Work with browser vendors and others to potentially adjust CWV thresholds such as FID to be more meaningful
    - Push for the prioritization of several open issues related to measuring resources w/in iframes
    - More discussion needed around third-party script scheduling/reporting API

Main Thread Scheduling APIs - Scott Haseley

Video, minutes

- We presented the various scheduling problems and APIs our team is focused on

    - scheduler.postTask: Queue tasks with browser scheduler, with ability to cancel and reprioritize groups of tasks (*in OT through mid-January 2021*).
    - scheduler.currentTaskSignal: provide a way to provide the current task context (priority, etc.) (*also in OT through mid-January 2021)*.
    - **Yield:** Provide an ergonomic and efficient method of breaking up long tasks (revised, narrowly-scoped proposal in progress)
    - 3P Script Scheduling: provide developers with some control over scheduling 3P execution on main thread (*currently exploring data and API shapes*).
- We discussed several other problems/APIs on our radar
    - Priorities on other async work, task ordering guarantees, more context propagation, interaction of postTask and rendering, scheduling microtasks, after-task callbacks, etc.
- Some discussion on 3P scheduling and the idea of creating “scheduling domains” to isolate parts of the page
- AIs/Next steps
    - Share more concrete thoughts about yield and 3P scheduling with group as proposals take shape

Interactions and Event Timing - Nicolás Peña Moreno

Video, minutes

- We talked about the main problems with First Input Delay (FID):
    - It does not measure end-to-end-latency
    - It only considers the first user interaction.
- For the former problem, we need to consider asynchronous work, which is related to Patrick Hulce's talk on longtasks.
- For the second problem, we proposed exposing an interactionID which would enable knowing which events correspond to the same user interaction and would allow correctly aggregating the entries received to compute a per-page metric.
- We discussed the various open questions, such as how to handle pointerdowns (not all end up in clicks) and more continuous events like drags.

Prerendering [pre* with author support] - Jeremy Roman

Video, minutes

- Discussed how current browser privacy models interact with prerendering, and what measures are needed to be consistent with anti-tracking measures.
- Discussed prerendering an uncredentialed version of the page and have developers “upgrade” it when the user navigates.
- Some attendees had questions on the necessity of an “upgrade path” compared to loading credentialed pages in their own partition.
- Some attendees expressed concerns about whether prerendering is likely to succeed given mixed results in previous attempts.

- We briefly discussed the implications of prerendering on performance measurement APIs and Core Web Vitals, saying that we likely want to report both the prerendered load and the user's experience navigating to a prerendered page.
- One attendee expressed concern that prerendering may be vulnerable to timing attacks of some kind, which could be investigated later.

Prerender PageVisibility - David Bokan

Video, minutes

- Discussed options for how page state should be reported in prerendering/portals modes. Should we bring back visibilityState == 'prerender'?
- Related issues were brought up around tab/app switchers, where the page is visibilityState == 'hidden' but content is still shown. Related to #59
- Decided more investigation was needed, Domenic suggested enumerating all the use cases and states and trying to find some reasonable set of exposable states. David Bokan looking into that.

SPA reporting - Michal Mocny

Video, minutes

- In this session we review SPA navigation and problems they provide with RUM metrics gathering, as well as synthetic testing.
    - Review, then discuss what it would mean to identify a soft navigation and how we would ideally change what we measure.
- **Discussion:**
    - Patterns typical with performance measurement when transitioning from MPA to SPA today, as well as issues with Attribution.
    - Issues with blending performance metrics between initial Page Load and soft navigation.
        - Is it important to measure "MPA vs SPA" or "first in session" vs not ("landing page vs not")?
        - Today, the "default" for MPA is a blended report across all nav types. The "default" for SPA is a segmented report with focus on landing pages (by the nature of perf reports).
    - Effects of Caching (on both page load and soft nav)
    - **Request:** support resetting all metrics when timeline is marked by developer.
    - How to reset Paint timing? Suggestion: browsers should do the naive thing, and developers can use Element Timing if they need something smarter.
    - Security concerns about measuring arbitrary paints.
    - History API URL update is a single moment in time, but soft navigations are a span of time. Sometimes most work comes before the URL update, sometimes URL is the first thing, and often the URL is updated arbitrarily in the middle.
- Next Steps:
    - Need more blog posts :)

    - Create a public repo, with focus on listing out *concrete* use cases we aim to solve (with a focus on real existing apps and problems if possible).
        - Aim to ask large web property owners to submit use cases.

BFCache Reporting - Yoav Weiss

Video, minutes

- We discussed the various options of exposing BFCache navigations, and the backwards compatibility implications of firing new NavigationTiming entries
- Decided that the option of firing a new typed NavigationTiming entry is the way to go here, as well as firing the relevant First* entries that go along with it.

isInputPending update - Andrew Comminos

Video, minutes

- Status update
  - Shipping in Chrome 87
- Discussed interactions with Long Tasks API
  - Decided to report isInputPending usage inside of long tasks as a boolean or "input starved" signal, rather than omit entirely
- Discussed potential interop with yielding APIs
  - Deemed unnecessary to yield only to input
  - Current behaviour with setTimeout works for most UAs (particularly those who dispatch events FIFO)

Network Diagnostics API Proposal - Noam Helfman

Video, minutes

- Discussed different use cases and tools for network information diagnostics.
- Proposal to extend network information API with custom threshold a custom logic handling network condition changes has been discussed.
- Proposal for a new API to ping local gateway has been presented – there was some pushback related to privacy and insufficient clarity and justification for the use case.

performance.measureMemory API update - Ulan Degenbaev

Video, minutes

- Recent changes to the API were presented:
    - The API is now gated behind self.crossOriginIsolated
    - The API provides information to identify iframes in the result

    - The format of the result is generalized to allow other memory types beside JS.
- We discussed whether "bytes" in the result mean physical bytes or virtual bytes.
- We discussed the scope of the API and whether it should report the whole browsing context group or not. Concerns were raised about reporting memory usage of cross-origin iframes and potentially exposing the underlying process model.

Rechartering, Call for Editors - Yoav, Nic

Video, minutes

- We discussed the charter draft and various decisions that we needed to make
- We decided not to try and publish “almost done” specs to REC before the rechartering, or at least not block on that
- There was general agreement on moving most specs to the CR-based living standard model
- For specs transitioning out of the Group, we decided to move them out of the deliverables.

Long Tasks attribution - Patrick Hulce

Video, minutes

- We discussed the motivation for attribution and several previous attempts in lab tooling that did not work very well toward that goal.
- We outlined the approach that is currently working well in lab tooling and discussed its implementation cost which uses intertask initiator information.
- There was some interest in exploring this as an addition to the long tasks spec.
- There was some concern about the implementation cost that will require some further research.
- **AI:** Patrick Hulce to file an issue for further discussion.

JS self-profiling update - Andrew Comminos

Video, minutes

- Presented results from Chrome origin trial
  - Positive developer sentiment, useful to discover pathologically bad cases
- Discussed activation mechanism
  - AI: Add support for disabled-by-default features to Permission Policy
- Talked about potential candidates for renaming
  - Popular candidates included JavaScript Sampling API, Performance Profiler
  - AI: File GitHub issue, request feedback from TAG

TAO and CORS/CORP opt-ins - Yoav Weiss

Video, minutes

- We discussed the different categories of information Timing APIs expose and how we can reason about unifying the opt-ins for them.

- We concluded that while CORS does give you access to resource-level information (timing + size), it doesn't currently provide origin-level or network-level information, so we shouldn't extend its semantics to include those.
- We discussed whether CORP should enable exposure of resource size, which devolved into a discussion of the semantics of CORP, and whether it implies that a resource can be just embedded or embedded and read.
- **AI:** Yoav to sum up the discussion on an issue, so we can continue it there

Reporting API updates - Ian Clelland

Video, minutes

- Changes made over the last year to the Reporting and Network Reporting specs were presented, along with an update on their implementation and usage within Chrome.
- We discussed the need for something like Origin Policy to enable out-of-band configuration of Network Reporting, though Origin Policy has some blockers that have prevented it shipping so far.
- We discussed whether there was interest from any non-Chromium implementers to ship Reporting
- We discussed how best to resolve the remaining privacy issues on the spec, including when it is appropriate to bring these issues to the Privacy IG / CGs
- We discussed whether anything other than spec-level guidance can be done about the problem of capability URLs appearing in cross-origin reports.

Reporting API and performance metrics - Yoav Weiss

Video, minutes

- We discussed a proposal for a reporting API, where developers construct the metrics they want reported in JS, but the browser ensures the reports are sent before the document is dismissed.
    - that will increase report reliability and prevent developers from having to rely on dismissal events to manually send their beacons.
- We concluded that such a proposal would be useful to reduce the need for backend "session stitching", even if this requirement will not go away entirely in cases where we want to report both real-time and continuous results.
- RUM providers expressed interest and said they'd migrate to such a solution, if available

Minutes

Monday Oct 19

Intros, code of conduct, agenda review, meeting goals

- **Yoav:** Welcome!
- **Nic:** Welcome to TPAC!

- ... Mission is to think about, measure performance of web. User agent features and APIs
- ... Highlights: New co-chair, HR Time L2, PaintTiming WebKit implemented
- ... F2F SPA focused interim meeting was cancelled due to COVID
- ... Closed 86 issues in Github
- ... Rechartering discussion on Wednesday
    - **Draft:** https://docs.google.com/document/d/1K6l5JlEaUq9eSBNI9HGoLfeN9hpXLHlHPRW-WTjtBlU/edit
    - Current charter runs through 2020-12-31
- ... Call for Editors - also on Wednesday. Looking for new editors to take specs forward
- ... Lots of incubations we may want to adopt: 13!
- ... Usage is up (mostly)
- ... New member organizations and invited experts
- **Yoav:** W3C has a code of conduct (CEPC)
- ... Promote diversity and seek diverse perspectives
- ... Try to avoid speakers dominating time
- ... We don't operate a queue, prefer natural discussion
- ... We can use chat to indicate they want to speak
- ... CEPC has section on unacceptable behavior and safety vs. comfort, important to to read
- ... We are recording this event and will be publishing this later
- ... Agenda for today
- ... (round of introductions)

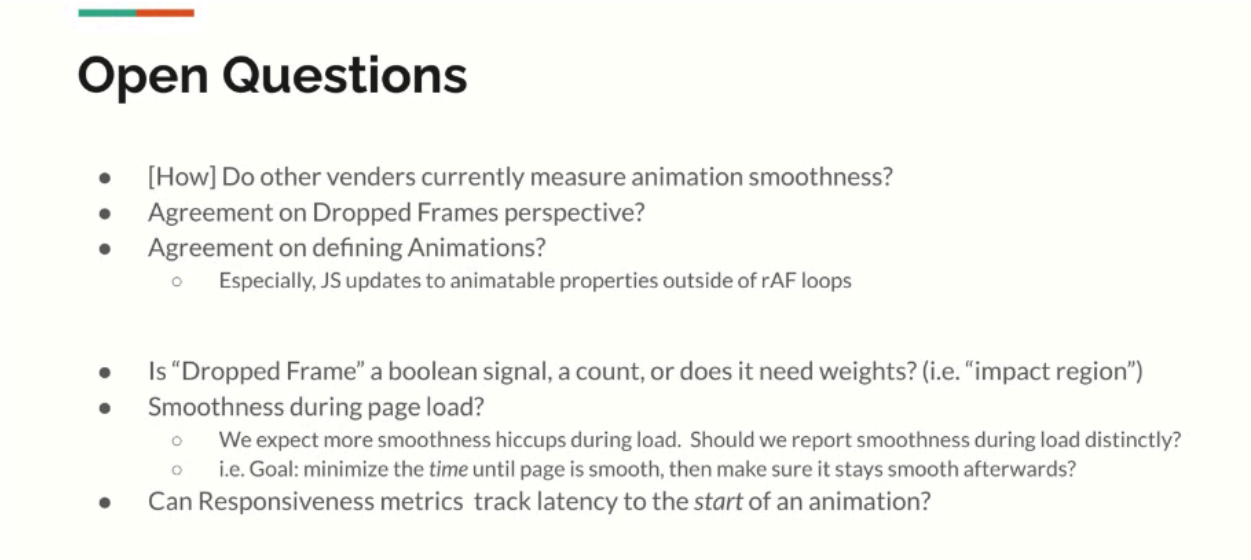
Smoothness Reporting for Animations and Scrolling - Michal Mocny

- **Michal:** Why discuss smoothness?
- ... Animation and Scrolling is big part of UX on the web
- ... Stuttering is very visible
- ... How do we measure smoothness?
- ... Often in terms of FPS, especially coming from gaming
- ... When it comes to the web, it has some flaws. A perfectly static website is perfectly smooth, doesn't generate any new frames
- ... With multi-threading, what does FPS even measure
- ... Why it matters: Missed opportunities to show expected animation updates, aka "dropped frames"
- ... Animations: Scroll, pinch/zoom, rAF loops, CSS, canvas/video/etc
- ... Animations are not: inserting new content, clicking button to produce UX response, form input appearances, background loading
- ... Dropped frames: Anytime an animation is expected to produce an update for vsync, yet the page does not do so
- ... Animations do not always expect to produce an update
- ... CSS animations can have idle periods defined
- ... Event handlers can delay animations based on scrolls in multiple ways (e.g. delays in handler or triggering updates afterwards)

- ... How do we report on missed animations?
- ... Option 1: Report all raw data points, and see which animations are able to complete for each frame ID
- ... Concerns: ergonomics, performance, privacy/security
- ... Option 2: Report per animation (summary) at the end
- ... Simplifies attribution
- ... Report #expected frames, #produced frame, duration
- ... Can calculate %smooth or FPS
- ... straw API:

```
interface PerformanceAnimationTiming : PerformanceEntry {
      entryType: “animation”,
      type: <one of “css-animation”, “scroll”, “js-animation”, ...>,
      startTime: <when it started>,
      duration: <how long did it last>,
      interval: <average vsync interval during this animation>,
      framesProduced: <# of frames produced during this animation>,
      framesExpected: <# of frames that were expected during this animation>,
      element: <the animating element, if/when possible>,
      id: <the animation id, if present>
};
```

- ... Option 3: Report per frame (on timeline), similar to FrameTiming API proposal
- ... Better matches to what user experiences
- ... Number of updates produced / expected
- ... Option 4: Single summary, "final smoothness"
- ... Or maybe at key moments such as visibility changes
- ... Mix and match the options
- ... Could offer page-level summary but also give details per animation or something
- ... Open questions:
-



-

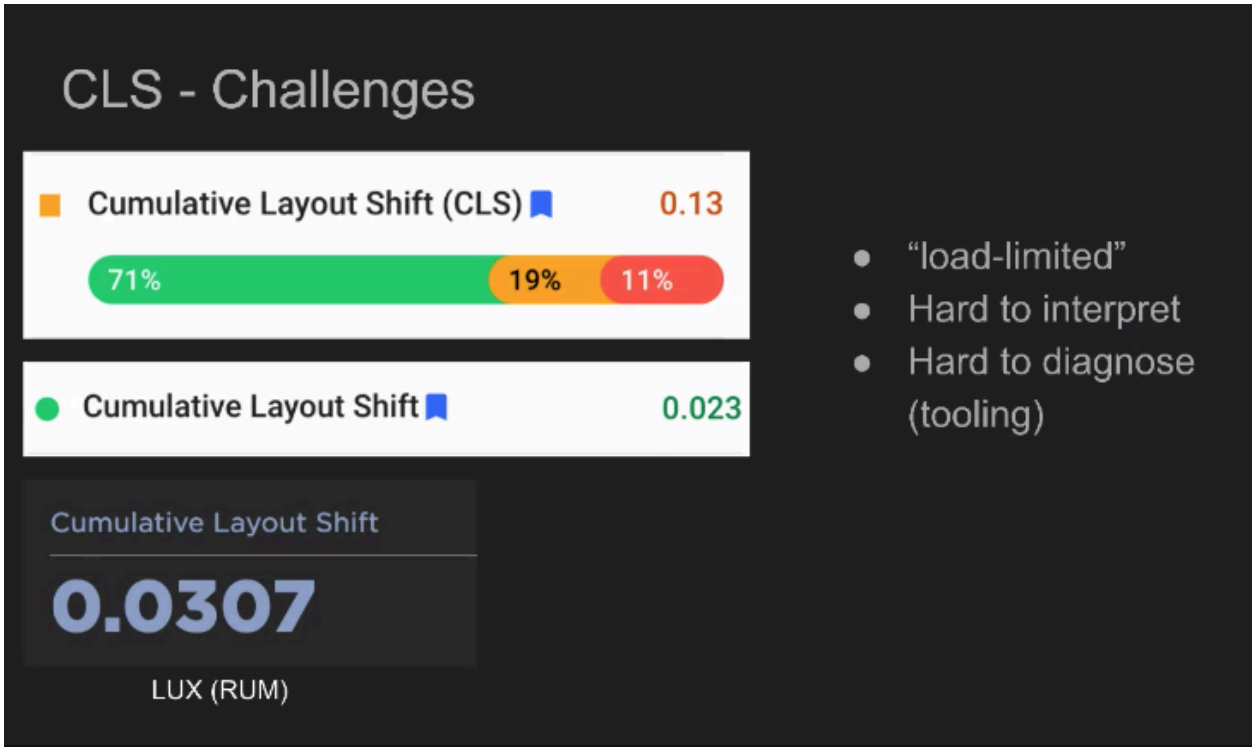
- **Ryosuke:** What do you mean by expected frames in option 2?
- **Michal:** For a CSS animation, for a key frame that was expected to move from the previous frame, and if the browser cannot produce it in time
- ... for animations where there's no transform between keyframes, there's no expected new frame being produced
- **Ryosuke:** In some environments, they adjust frame rates based on what's animated.
- **Michal:** In Option 2, there is an interval (which is avg vsync interval), so if frame rate increased, the interval would change
- **Ryosuke:** Assuming you can get that information, I'm not sure we have that capability
- ... For JS based animations, it can be more complex because we're depending on script to draw or not. Not clear if we missed a frame. Expectation of a frame is complex to me.
- **Tim:** It's unclear in the case where vsync is not constant. In the case where it is constant, is it hard to know if there's no change?
- **Ryosuke:** Hard to define this. On a device that was 120hz, and an animation that started when the page was running at 30hz, and it needs to ramp up. Number of frames being produced is dependent on the animation being done.
- **Michal:** In this case there was an opportunity to present (vsync), but it could not be
- **Sadrul:** "The next rendering opportunity". Have to depend on the system to give the presentation frequency. If a device ramps up from 30 to 60 to 120fps, for each framerate, we know up to how many frames we can present.
- **Ryosuke:** On fast displays (120hz), the page may not be able to even animate that fast
- **Tim:** You can switch refresh-rate and not affect the UX
- ... Look at dropped frames per unit time, instead of dropped frames per expected frames
- **Ryosuke:** If everywhere else, pages are rendering at 120hz, and another page is only at 30hz, the user would notice the difference
- **Michal:** Does the refresh rate ramp up for developer-provided frames? CSS animations are defined semantically, but what about rAF loops where the developer code just tries to keep up.
- **Ryosuke:** The system is deciding how fast to present things. It sees how fast you're going and then tries to goes faster. I don't know the exact algorithm here.
- **Tim:** Seems like you have better experience with variable refresh rate cases, do you have alternate ideas on how to present UX in those cases
- **Ryosuke:** These are really hard cases
- ... In the case where refresh rate isn't changing at all, there's no problem
- ... I don't really have a great suggestion here
- **Robert:** If we know the system has the opportunity to ramp up, we know any other rate is dropped frames.
- **Noam:** Would it make sense for API to allow developers to specify what they're interested in for smoothness criteria. e.g. in some applications, 20FPS is OK, in some it's intolerable below 50-60FPS.
- **Ryosuke:** If we can say here's the theoretical max rate this device is capable of versus what it presents at
- **Michal:** The worst thing to happen is there was an opportunity to present, and the animation had a useful thing to give, but for whatever reason it wasn't updated

- ... There are other things to optimize for, like having the interval as low as possible. But you want to make sure you want to present when you have the opportunity to
- Patrick Meenan: Would be useful to look at the video game industry for guidance, e.g. report device supported and minimum framerate, percentiles
- **Tim:** That kind of research seems valuable to me. We could also take a look at the time it takes to present a frame. Downside of being less representative of user experience.
- **Sadrul:** How do other vendors measure smoothness right now
- **Nic:** We do capture very simple metrics: frame rate over time, but not very valuable, due to the issues presented here. Would love more useful UX metrics. Currently just measuring rAF.
- **Benjamin:** Is the intent to grow towards media capabilities WG and have this be applicable to video?
- **Michal:** Haven't considered that
- **Tim:** A good job for the video use-case would require a lot of additional data, bitrate being streamed, etc. My guess is that it'd need first-class support
- **Benjamin:** So different APIs
- **Tim:** My intuition is that would be the way to go. This would handle video in a sane way, but won't give a full picture of video smoothness
- **Michal:** Other thoughts?
- **Nic:** In many ways it's nice to report on the "bad" things. Easier to present or track user pain. Even just counts of dropped frames is useful to track
- **Carine:** Video/media use cases are in-scope for other WGs. We can talk to them
- **Michal:** Next steps - expect more on this
- **Ryosuke:** Any repo where this is being discussed
- **Michal:** Coming soon
- **Ryosuke:** Let us know!

SpeedCurve hot topics - Cliff Crocker

- **Ciff:** Working on webperf for ~18 years. Want to give some feedback and report what customers are seeing in the wild
- ... Core Web Vitals, ads measurement, Server Timing
- ... Seen a huge wave of interest as a result of Core Web Vitals in terms of people wanting to make their sites faster, from outside our echo chambers: SEO, marketing, CEOs, etc
- ... Problems: Browser support for the metrics. Want to see adoption beyond just Google (Facebook, Adobe analytics and other marketing tools)
- ... LCP RUM data: p50 1.2s, p75 2.2s, p95 5.2s
- ... LCP been easy to understand and people tend to be doing well here
- ... Chrome 86 fixed opacity issues, resulting in more accurate (higher) LCPs
    - But getting support calls as a result, may result in erosion of confidence in the metric
- **Nic:** Chrome has a public change log, that we can point mPulse customers to better understand those changes. Helpful for browser vendors to share that.
- **Cliff:** Yeah, we recently started doing something similar

- … CLS - people are doing relatively well, but depends on industry
- … Took some time to get people's heads around that. We got over that hump - lower score is good, but they run into challenges when comparing field and lab data

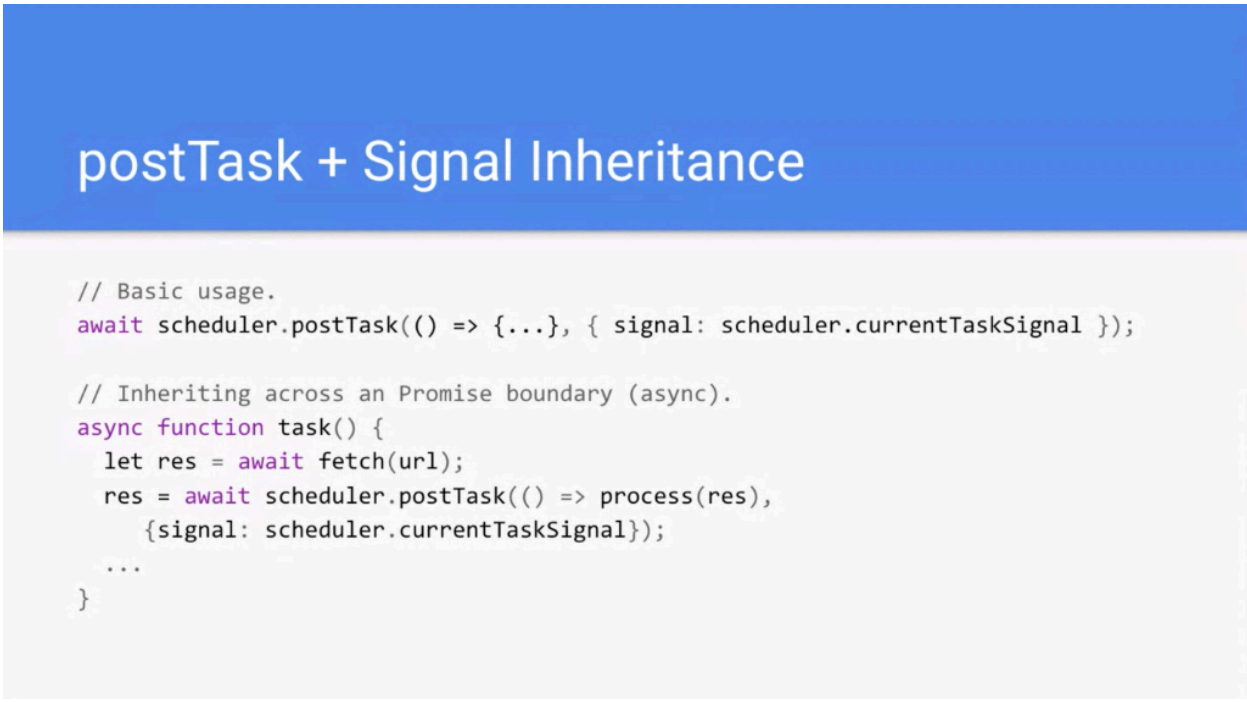


- ... Load-limited affects RUM tooling
- ... Hard to compare RUM to CrUX. Tooling doesn't support what's collected in CrUX
- **Benjamin:** So how are you explaining this?
- **Cliff:** CLS is a cumulative score, show when our beacon is fired (at "load" event), and see that other shifts happen after that
- … Don't know if this is something where we can improve how we collect metrics, or if there's a better way to normalize CrUX and RUM
- **Michal:** CLS is full page lifecycle feature, there are also problems related to other metrics (e.g. smoothness, responsiveness) where user action is not represented in the lab.
- **Yoav:** I think there are two problems here. For all User Interaction dependent metrics, lab and RUM will inherently differ. That's fine and we'll accept that.
- ... Many RUM providers today don't capture metrics for the full lifetime of the page (e.g. at onload or after). So we will have differences between CrUX and RUM vendors. We need to try to close that gap. I have one potential solution to that later days.
- Nicolás: Is the main complain lab settings vs rum data, or lab settings vs. CrUX
- **Cliff:** I think the later, it starts there. But then when looking at Lighthouse, you also see a difference.
- ... Challenge comes back to how tools are quick to measure this
- ... This is how analytics has always done this, there are some metrics we can capture later but we generally get everything at load-ish event
- ... Reporting API is potentially a way to close that gap

- **Patrick:** For a full page lifecycle session, a 0.1 CLS for a 12 hour gmail session vs. a 0.1 CLS for a load page are very different experiences
- **Michal:** Active accumulating is not always the best measure
- **Cliff:** We've had to improve our tooling, and normalize data between synthetic, CrUX, RUM
- **Patrick:** For Lighthouse specifically, we are working towards getting post-load data. Trying to work on naming the different CLS measurements here.
- **Benjamin:** the "I don't know how to fix this" is concerning
- **Cliff:** Agree that it's a problem, and tools need to fix it. Maybe we can focus on the largest shifts first
- ... FID - most customers pass it fine. Seems like the bar is too low, 75th %ile is far below 100ms
- ... customer example: 19ms 75%ile FID, but "JS longest task" at 300ms, which is not great
- ... So not shining a big enough light
- ... LTs may be after load, wait for user interaction, so that may explain some discrepancies
- ... Want LT attribution, today it's hard to find the source of the LT
- ... FID may need to be a higher bar
- **NPM:** Patrick will be discussing how to do LT attribution in Lighthouse and maybe we can use it here too
- **Annie:** Folks on the team also think that the bar may be low
- **Pat:** Maybe it's just not a problem in 75% of cases?
- **NPM:** Problem may be elsewhere
- **Cliff:** Display ad performance - how can we measure 3P performance
- ... Discussion with a customer a few weeks ago - how haven't we fixed it yet?
- ... Anything we could be doing about this today?
- ... Otherwise, visibility into resources in iframes would be huge
- **Nic:** As a RUM vendor, +1 to all that
- **Scott:** For Scheduling, we see a desire from folks to control what 3Ps are doing, deprioritizing them, etc. Great to collaborate on getting metrics on this
- **Cliff:** Server Timing, still under used by pages. Guessing that a lot of this is coming from Akamai. Fill that this is a missed opportunity
- ... Call to action - are we really doing everything we can with Server Timing.
- ... SpeedCurve doesn't yet collect it
- ... WPT surfaces it, but can we do more?
- ... Would love to see other CDNs surface it
- **Nic:** RE CWV, have your customers reported good/bad things about the metrics, business metrics, etc?
- **Cliff:** Yeah, got a mix. Strong correlation with FID, of bounce with LCP, but less with CLS
- ... They're happy with CLS that they knew there was a problem, and now we can measure it
- ... Want to collect more data on long term conversion rates, but see strong correlations with user frustration

Main Thread Scheduling APIs - Scott Haseley

- **Scott:** Keeping high level, not diving into things as much
- ... Available APIs:
  - scheduler.postTask - Prioritized task scheduling, enables coordination through priorities, controllable (cancel or change priority)
    - Post a task and get a Promise that resolves when the task is done
  - TaskController has a signal that enables aborting a full class of tasks or change their priorities
  - Currently available in Origin Trial, and behind a flag since M82. React and AirBnB plan to start experimenting with it ASAP.\
  - React replaced their user-land scheduler. AirBnB use it to break up long tasks
  - Next steps: getting feedback from OT and see where we go
- ... Followup - signal inheritance
  - Folks want to inherit the signal to create subtasks that listen to priority changes
  - postTask + signal inheritance



  - There's some risk with propagation. Inheriting the context of some tasks, but that soon degrades to "everything is high priority"
  - It's in the OT, but unclear if this will be part of the shipped API
- **Subrata:** How is starvation handled in this case?
- **Scott:** Tradeoff between giving developers strict guarantees and giving the browser control. Starvation is isolated to postTask tasks.
- … First pass is strict priority order. Want to get feedback on that to see if it's good enough, and have metrics to measure starvation. Keeping an eye out for it

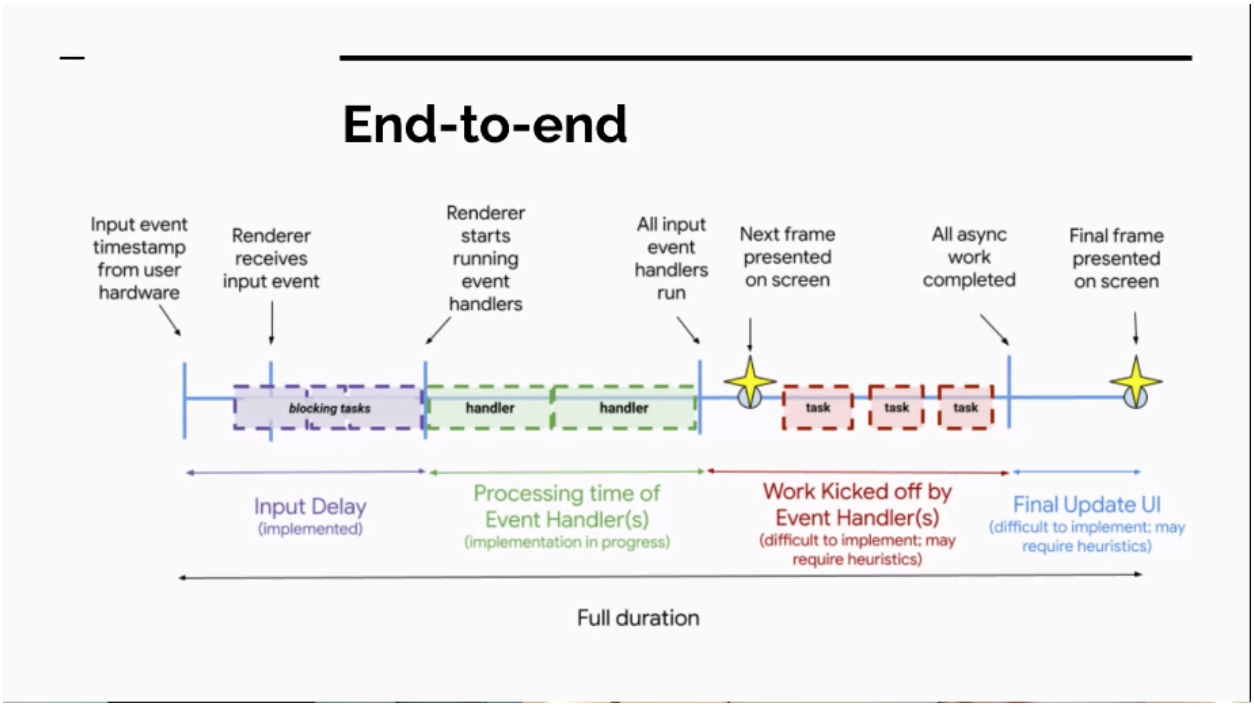
- ... Other problems we're exploring
    - Yielding in Long Tasks
    - Has come up in the context of isInputPending
    - Currently folks schedule setTimeout/postMessage, but you may "miss your place in line" and it's not great ergonomically
    - Also want to reduce the penalty for yielding, enabling faster control
    - Otherwise, folks want "Yield to rendering/input/network/etc"
    - Initially proposed scheduler.yield(), returning a Promise which returns when the thing you're yielding for is done
    - Want to minimize scope, just for postTask, which may open up options
        - E.g. integrating generator functions, so we could make postTasks understand generator, and turn functions there to yieldy functions
- **Michal:** Reminds me of the spawn pattern
- **Scott:** You could imagine this changed into "I want to yield to specific things". It's exciting for us how easy it is to insert yield points that would work with both async and sync code
- … Will be exploring that soon
- … postTask - app specific priorities
    - A lot of partners that develop their own schedulers. Continuing to do that would require building their own queues on top of postTask
    - One proposal is to add an option for a "rank" for the order of the task in the priority queue
    - If you have multiple parties on the page that have different ranking systems, we'd want to be able to schedule both
- **Yoav:** It's unclear how you would reconcile multiple ranking systems from different parties? Do they all go into a single queue based on the same rank? Or normalize?
- **Scott:** If you provide a task signal, that indicates the priority can change. Not clear if that will work for developers. Need to understand what guarantees they want.
- ... From browser's perspective we want freedom to schedule as needed
- ... Could provide a larger domain, where everything in the domain is ranked
- ... With v1, what guarantees do we provide and will that hurt us down the line
- **Yoav:** If you don't provide a domain, you're just relying on the signal, or some other queues?
- **Scott:** A signal, or there might be a default domain, simple apps would just use that.
- ... 3P script scheduling:
    - Give developers some control over when 3P scripts run on main thread
    - Explore what is being done on main thread by 3P libraries
    - Control might be deprioritizing, delaying, etc
    - How do we prevent or mitigate a 3P task from using highest priority all of the time
    - IFRAMEs might be a good place to annotate what is important
- ... Further out: postTask priorities on async things that developers don't have control over, I/O such as IndexDB or network resources
    - Tried prioritizing all DB tasks, showed some improvements, but unclear if this impacts on all sites in general
    - Allowing developers to specify priority might make sense

- ... Can postTask be extended to microtasks? Demand from frameworks
- ... After-task callbacks: being able to measure all microtasks to determine full duration
- ... Async task graph tracking: May spawn multiple async tasks, useful to know for bookkeeping, measurements, task harness
- ... postTask and rendering: how important is rendering compared to other work (tasks)? DOM R/W coordination
- **Noam:** Instead of just how many tasks, provide a time budget
- ... Would avoid response input delay
- ... Multiple very short tasks to avoid a LongTask can still starve rendering
- **Scott:** We've made some changes since then, are open to feedback and questions
- **Cliff:** On 3P prioritization stuff, is there a proposal ready?
- **Scott:** Not a formal proposal yet. We want to take a look at the space more wholistically first. I can update these slides with that proposal.
- **Cliff:** I think it would be valuable, though it could be hard for a developer to know what 3P domains are important
- **Nicole:** We realized we needed more data about what these 3P plugins are doing anyways. Took 10 and put them into a site to see what it's doing when.
- **Yoav:** Even things like URLs need some sort of opt-in from 3P frames to expose what they're loading. Similar to Nic's proposal for 3P frames to opt-in to parent frames
- ... Controlling 3P frames is easier than to get details about what they are
- **Nicole:** Surfacing what third parties are doing upward will just put application authors in just the same bad position they are now, managing behavior of something out of their control
- **Yoav:** Theoretically, unless RUM providers can make that surfacing actionable
- **Nic:** As a RUM provider, big request from our customers to get insights into what is causing the pain, and we could jump through those hoops
- **Scott:** Useful for then if there is some sort of control, we could make it actionable
- **Nicole:** Maybe we could reach out to security folks to see what is possible.
- **Nicole:** Maybe we should see if we can start a convo about security folks to see if it's possible
- Nicolás: Also a problem of when a 3P script injects into the main thread, might make it more incentivized. If 3P are punished for being in their own context, they will try to inject into the main context
- **Yoav:** We shouldn't make that only possible only in the context of the frames and not main frame, then 3P will shift their work to main frame
- **Scott:** We're focused on the main frame/thread as well that's competing with 1P code etc
- ... example GA is doing setTimeouts for all of its things and those happening at inopportune times

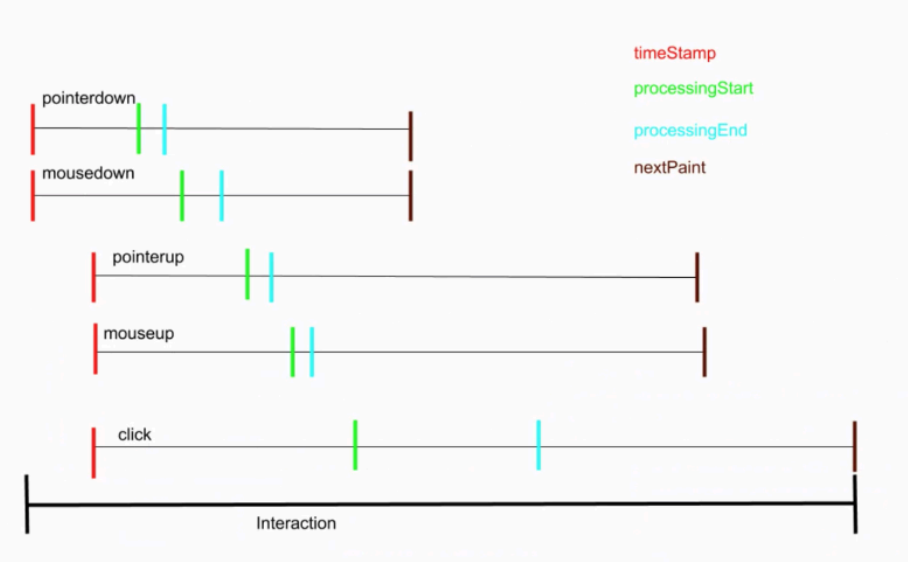
Interactions and Event Timing - Nicolás Peña Moreno

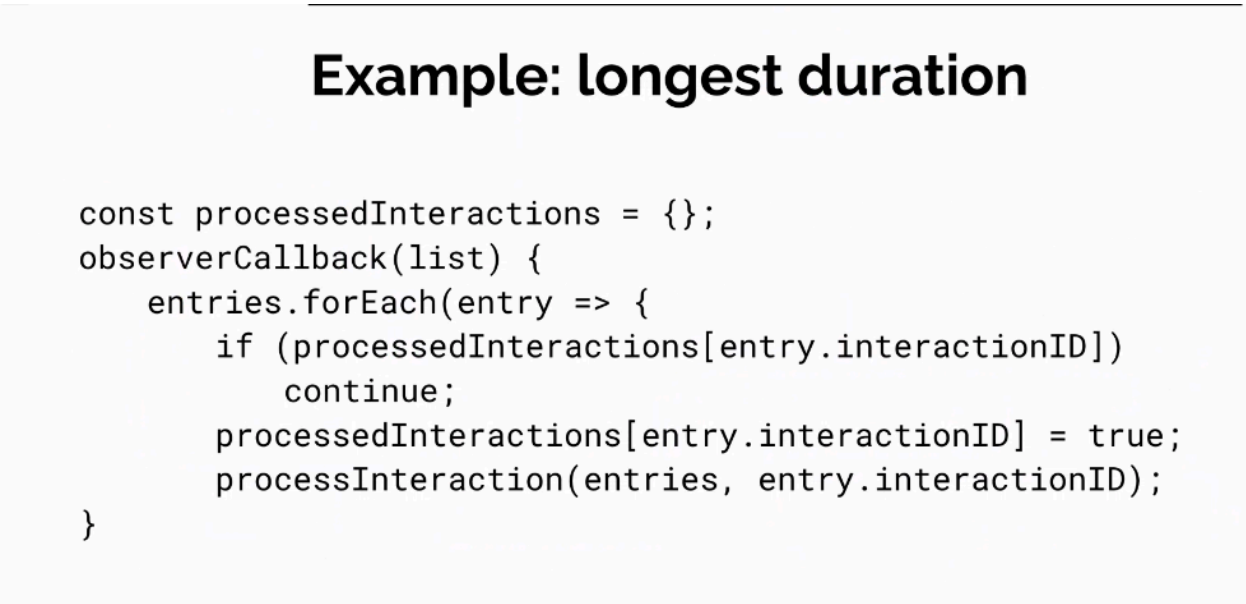
- Nicolás: Responsiveness: Current metric in CWV: FID
- ... Does not capture "end to end" latency of user interaction - when user clicked till the event starting to be processed

- ... Only measures the "first" - not great for SPAs
- ... Diagram shows different points in time for a user interaction



- ... After a user interaction, there is the first frame, the first star above
- ... Then there's other async work
- ... How do we measure the "Final Frame", aka the second star above?
- ... Determining the "end" for async work requires tracking some sort of causality, would need to be a heuristic. Related to LT attribution, which needs something similar
- ... Problem of aggregating, e.g. a single user interaction may correspond to a large sequence of events. E.g. tap => mouse up, mouse down, pointer up/down, and click
    - Can result in over counting of taps
- … Want to enable developers to track per-event metrics, but also per user-interaction
- … Proposal - adding an interactionID to PerformanceEventTiming that would be the same for events triggered by a single user interaction
- Example
- From this, via a PerformanceObserver, you would group interactions by InteractionID and get the max of those handlers

- 

- 

- **Alternatives:** New Interactions API (would require yet another API)
- ... Polyfill on top of EventTiming: events with "close" timestamps would be in same interaction
- **Noam:** Very important space to solve the end-to-end measurements, as well as the interaction ID. Tried to tackle this with existing APIs.
- … used EventTiming API without addressing the multiple event issue, and then combined Element Timing to measure the added image's paint timing.
- Nicolás: Wonder how different that is from the duration value that's already available in EventTiming. That captures a timestamp to the next paint after the event.
- **Noam:** The problem is that it's not guaranteed that the relevant paint would be in the next frame
- **Yoav:** In the scenario you're describing, with a specific element you want to measure, this makes sense. We'll talk about more various task tracking in context of LongTask

attribution later. Can bring that back to coalescing of multiple events to a single user interaction.

- **Noam:** Important for us to address as well.
- **Nic:** For mPulse interaction tracking, tracking through FID. See a lot of value in tracking async tasks triggered by events.
- **Cliff:** Tracking FID as well. With attribution, this is very helpful. Started tracking other input things, like when user input happens, but not sure it's super helpful, because applications vary on that front. Keen on getting more timing around event handling.
- **Pat:** Aggregation of duration and filtering by event id - when we stagger events, do we want to take the start of the first event and the end of the last one?
- Nicolás: Open to ideas about what we should be measuring. Feedback welcome on https://github.com/WICG/event-timing
- **Pat:** Start of pointerdown till end of click in the example seems to be what we'd want
- Nicolás: We care about the user's interaction until the thing is completed.
- **Pat:** It's possible with your proposal, just not a simple oneliner
- Nicolás: Currently only surface slow events
- **Tim:** You'd still have a problem you can't necessarily map from a pointerdown to a click event
    - Do we want to count the duration the user help their finger to the screen? Maybe
- Nicolás: Yeah, unclear what we want to measure with long taps. With interactionID could let you know it was a long tap. Not sure if the event surfaces that.
- **Noam:** How would that work with a combination of events, e.g. pointerdown to a mouse-move, dragging
- Nicolás: That's a good question, the discrete events in EventTiming, but for a drag how would you classify the interaction
- **Noam:** Sometimes you care about the pointer up event, not the pointer down. But the API with the interaction ID enables you to do that
- Nicolás: Maybe until the finger is lifted, that's how long the interaction is tracked

Tuesday October 20

Prerendering [pre* with author support] - Jeremy Roman

- **Jeremy:** Looking at bringing back pre-rendering
- ... Requires three parts: Trigger, Opt-in, Behavior Changes
- ... Behavior Changes: Limit nuisance behavior and limit cross-site tracking
- ... via uncredentialed pre-navigation fetch
- ... Sites cannot access their unpartitioned storage before navigation
- ... Opt-In: Site may need to "upgrade on navigation" if a user is already logged in after un-credentialed pre-navigation fetch
- ... Trigger: Referring page needs to indicate that pre-navigation fetching is safe from unwanted side-effects, resource is compatible, and user is likely to navigate. Link rel prerender is the previous iteration of this

- **Domenic:** this is all pretty early. Portals came up with the concepts, but then realized it's more general
- **Yoav:** Something I find interesting about upgrade path is that it can encourage cacheable static upgrade HTML that can upgrade itself to a customized experience
- **Domenic:** most sites already have a logged out view, and they can add a bit of JS to modify that state
- **Michal:** What does the upgrade flow look like?
- **Jeremy:** exact shape of the API depends on how the API evolve and how storage access and other adjacent APIs evolve
    - current API shape
- … Ultimately what authors want to do here is get notified when storage is available, and then load the data that's dependent on personalized state.
- … If the entire application is personalized there's little you can load ahead of time, but static sites may be different. Visiting e.g. Wikipedia, most of the content would be the same for all users, and only a small number of components would change
- **Michal:** thinking there would be some event that indicates that state change: prerender to render or storage availability, etc.
- **Domenic:** trying to figure this out. There are APIs for the different changes: permissions API that will tell you if you'd be auto-denied when prerendered, storage access in many browsers. Unclear if we should tell people to use these signals individually or bundle them. Early days still.
- **Alex:** If a site doesn't declare "I can upgrade" and we fetched it, will we throw it out?
- **Jeremy:** If the user has local credentials, we cannot use the response
- **Alex:** and if the user doesn't?
- **Jeremy:** Maybe. I don't see why we couldn't
- **Domenic:** we would still be unable to use the rendering, and it was rendered with shut down APIs
- **Jeremy:** yeah, but we could potentially use the response bodies
- **Alex:** If a site is static and credential-less, they'd still have to opt-in and their upgrade would do nothing
- **Jeremy:** yeah, as we can't determine that from the client side
- **Pat:** You're talking about reusing the renderer. The original prerender dies because of excessive resource usage. How are things different today?
- **Kinuko:** We still worry about resources, but we have throttling infrastructure that can enable us to prerender without consuming too many resources
- **Jeremy:** UAs that run on devices that have no resources for that, they can always do nothing with prerender hints. Hope that pages would identify that they are prerendered and use less resources while prerendered
- **Pat:** but that was the case before. The work on the site's side is the same. Was there adoption last time
- **Domenic:** I think we cannot depend on sites. The shift in thinking came when we implemented BFCache. We also had background tabs. This is extending the same principle - keep things around that will improve UX in the future.

- **Jeremy:** We also put more effort on sites caring about performance - e.g. CoreWebVitals. But agree that we can't rely on sites
- **Pat:** the performance impact when it works can be awesome. But worried that it may mostly work for lightweight sites and SPA/SSR would send down the credentialed experience and not want to do the credential transitions on the client side.
- **Michal:** implications towards performance measurement?
- **Jeremy:** started thinking about it, but no conclusions yet
- **Domenic:** pages still want to know when they prerender, but also need to know if that made the UX super fast. So we need 2 different metrics: the traditional load timings that may have happened in the background and what the user actually experiences, which could be 0 or close to 0. Still don't know how it will be manifested
- **Michal:** Couldn't agree more and will be talking about related issues.
- **Yoav:** One question to other browser vendors around non-credentialed prerender coupled with lack of access to localstorage. In the past we talked about privacy preserving prefetch and this is in a way an extension to that. Would that be something that is implementable?
- **Alex:** Implementable yes, but a little strange. Initial thought would be do a load with credentials and storage in its own partition.
- **Domenic:** Reason for no-storage is that you do need to eventually transition to first party partition. E.g. You'd need to migrate/merge two IndexDBs
- **Alex:** In our implementation, if you're on a.com and you prerender b.com, in b.com's partition, there would be no problem if it becomes the main frame's content
- **Jeremy:** If you load b.com in partition, it's not the partition that stores the user's current cookies and credentials, the user doesn't see themselves as logged in when they transition
- **Alex:** May be a difference between cookie partitioning in Webkit and Chromium. In WebKit I see no problem and no need for this transition
- **Domenic:** This is new technology. There's no way today to take an iframe and make it the top-level page. Without a transition, there could be a situation where the user has 2 tabs to the site, but will only be logged in in one of them
- **Alex:** If you're prerendering something, the current page won't have access to its contents, right? There's a difference in our thinking here.
- **Domenic:** wrote up our threat model
- **Yoav:** Maybe the difference comes from a model that works both from prerendering and portals
- **Jeremy:** I think this just exists with prerendering
- ... If you do fetch with credentials from a.com to b.com, it allows you to essentially do a subresource request with credentials
- **Domenic:** We're designing this to align more to Safari's model that Chromium's current model
- **Yoav:** We can discuss in upcoming WG call or offline on Github
- **Ryosuke:** Whenever you do a prerender you do a cross-origin request in a separate state each time, correct? Are there timing attacks?

- **Domenic:** There are timing attacks possible which is why we're trying to partition here. There are probably other attack here.
- **Ryosuke:** Something we can look into.
- **Yoav:** Can we talk about visibility aspects?

Prerender PageVisibility - David Bokan

- **David:** Need some way to signal to a page it's been pre-rendered
- ... Is it hidden enough? Some content shouldn't load/run in prerender, e.g. ads, analytics
- ... Content could be visible, needs to be up to date, e.g. portal
- ... Open Questions
- ... Compatibility: Check for visibilityState=="prerender", some pages check for if-visible-else, which could break
- ... Will require opt-in so compat is nice-to-have but not critical
- ... What does document.hidden return
- ... Portals: Allow showing a preview of a prerendered page, differs from prerender as it might be visible to a user
- ... Pages could be put into portal. Can go from visible to in a portal
- **Benjamin:** Portal idea might get us out of the jam of some visibilityState and App Switcher
- **Domenic:** We have an explainer for Portals, but it's being rebased on top of prerender explainers
- **David:** A portal kind of looks and feels like an IFRAME, where it starts out embedded on the page, but it's a link where you can see a preview of the navigation. Curious about app launcher discussions?
- **Benjamin:** We've had some discussions whether a page is visible or hidden if you're in App Switcher, that problem may have relevance to Portals
- **Michal:** Our discussion around App Switcher is if we're going to move off a two-state from visible/hidden, what else happens there
- **Yoav:** Content is "viewable" but it's not the top-level document. If we had a "preview" state that might help with both.
- **David:** In a portal you're not able to interact with content, more of a preview
- **Michal:** "Non-interactive" was the proposed state
- **Benjamin:** Having a new visibility state for Portal might be a smoother path forward
- **NPM:** Isn't visibilityState determined on the page level right now? Say if you have an iframe not in the viewport, it will still be considered visible.
- **Domenic:** People's intuition often think of portals as visible, but it's more aligned to a popup window that's overlaid on the page. Visibility state separate from the document.
- **Yoav:** Seems like we have multiple states that have overlap. For pre-render we could have hidden content, but we also need to know if it's prerendered to not do various things. It's both hidden and non-interactive, where one is a subset of the other. We need some way to mix-and-match those. Is the best path 3 or 4 states with overlap between them, or orthogonal signals we can mix together?
- **Michal:** That came to mind as well because so many sites check for "not visible" as a hint for hidden, etc. Maybe one way to get out of that bind is to think if visible/hidden is appropriate for all of those cases, and we have a separate visible type, such as for App Switcher
- **Domenic:** I could imagine a bunch of different boolean switches: app switcher, portal, background tab, etc. Could make a matrix of all of those and see if there's overlap and if

we want a bunch of booleans or a big enum
- **Jeremy:** the other aspect RE the ideal design is how can we reasonably get from the current to that ideal design with relation to existing content
- **David:** The other interesting thing is what would you consider the visibility state for a portal on a page. It's drawing content and you want it to be ready. But if you have a portal in a BG tab you don't want it to do a lot of work. Don't want it tied to top-pages visibility, because it's a timing communication channel, if you switch tabs and both get the visibility state at the same time.
- Nicolás: can't you do that with iframes?
- **Domenic:** IFRAMEs get their own storage partitions, while portals do not
- Nicolás: Makes it hard to reason about the visibility state of a portal
- **Yoav:** For a visible portal, you'd want it to animate, where for an invisible portal you wouldn't want that.
- **NPM:** There's two general use cases, (1) is know whether you want to execute some amount of work and (2) tracking performance of paint metrics
- ... Want portal to count as having paint metrics when it's still a portal, but hard if we can't tell if it's backgrounded
- **Domenic:** Might be better to always think of it as backgrounded. It's ok if your preview doesn't have animations. Extra preview on top of prerendering, and a prerendered page is not visible
- **Benjamin:** Does the pre-rendered page never change?
- **Jeremy:** We don't envision taking a single screenshot, we allow paints to happen and frames to be generated. UA could do that at a lower rate if desired.
- **Michal:** but that conflicts with the goal of treating it as hidden, when some pages avoid e.g. background play
- **Jeremy:** you want to avoid BG play when users don't see it. But you don't want the page to no show up at all. I'd imagine most people not do delay attaching DOM events on page visibility.
- **Domenic:** and to be clear, you do have to opt in. So the site should see they are rendering reasonably when portaled.
- **Yoav:** One conflicting requirement in preview state is for tab switcher and app switcher case, we do want animations and video to continue to play.
- **Benjamin:** But we don't know what the implications are on HTML
- **Domenic:** We've been thinking of prerendering things as browsing contexts, maybe app switchers are browsing contexts in some level. But largely for privacy reasons we don't want portals to know they're visible and doing animations and stuff.
- **Yoav:** Your concept of creating a matrix with all states sounds like the right next step.
- **Michal:** On the previous topic of prerender, I don't know if Benjamin had an opinion.
- **Benjamin:** Don't have anything to say, initial impression is better to start with something simple, the localStorage thing is complex
- **Alex:** I see the wisdom in not allowing localStorage and credentials, that could be problematic
- **Benjamin:** Do we know what types of sites got the most benefit from prerender?
- **Kinuko:** Good question, there were a few pages that had issues. I don't have data if there was a significant difference between sites.
- **Yoav:** There are also fundamental issues between credentialed content pre-rendering and non-credentialed prerendering. Credentialed prerendering would change site state (e.g. log off user), Non-credentialed problems would be with users missing their credentials because the site didn't do the transition work
- ... Are other browsers doing any prerendering? URL-bar based one?
- **Ryosuke:** When the user is typing the URL, there's no tracking, so can prerender the

site with credentials
- **Domenic:** What if a user types a user like example.com/logout and you prerender that with credentials
- **Alex:** Probably issues with edge cases like that
- **Domenic:** Some things like to shut down during prerendering like speech synthesis APIs. We want to produce an exhaustive list of APIs that we should shut down, which will be useful for browser-initiated prerendering as well. Not mandatory, but hopefully useful
- **Ryosuke:** if the user types the URL, there's no need for non-credentialed prerendering
- **Domenic:** Two variants, same-origin and cross-origin, where cross-origin does all privacy protections, but annoyance prevention is relevant for both
- **Kinuko:** Visibility discussion could be useful to that case as well
- **Ryosuke:** Things like auto-play audio probably don't exist in UAs anymore, but there are also probably other edge cases for things to disable
- **Jeremy:** Things that are behind user-gestures are fine, but there are some APIs which aren't protected by that.

SPA reporting - Michal Mocny

- **Michal:** SPA dynamically rewrite the page content instead of navigating, commonly use frameworks, use hash fragments or history API to update URL
- … RUM metrics and tools typically target traditional loads, and synthetic tools are also not great
- … Either in the dark on these sites, or get the wrong picture
- … types of SPA navs: full page recreation, content swapping, component update, infinite scroll
- … Loading: dynamically load what you need, large common template, preload everything (common with scripts),
- … initial load: client side rendering vs. SSR + hydration
- … subsequent loads: interaction vs. automatic
- … frameworks have signals for route transition starts and visual updates
    - Coverage is partial
    - "Soft done" when all initial work has been started, but not yet done
    - Async work may not be captured
    - zone.js tried to link async work, but may not work with modern async JS
- … some RUM frameworks mark route starts, listen to network usage & DOM updates
- … We also have UserTiming. Conventions there could help
- … Automatic marking via heuristics?
    - If we had conventions and marks, that could help when testing heuristics.
- ...Work on revamp of the history API - maybe we can provide an API to provide a well-lit path to give us something to hook onto when measuring
- … Measurement for SPA
- … Responsiveness - Event Timing can give us data on long event durations. Could provide onload?
- … We'd also want to track the next input, similar to FID
- … There's also the question of attribution
- … next paint we get from event timing, but interested in FCP or LCP

- … gets problematic with partial updates: the first paint is always contentful, or maybe we only want to count paints for new content
- … visual stability - layout instability is fine, but CLS is cumulative, so ignores softnavs
- … One difference between page load and softnavs is that layout shifts that happen post load would get a signal of hadRecentInput, but new content that gets added as part of a softnav and then shifted is similar to a page load, even if it happened within the 500ms timeout after user input.
- … Transitions from MPA to SPA -
    - SPA navs measure more soft navs, but less page loads
    - Faster transitions, but slower page loads
    - Transition causes many things to change at once, so hard to find causes
    - Fewer page loads result in worse caching
    - “More pageviews” - may be a result of transitions
- … Attribution - ideally, we could report on each route change and reset metrics
- … But you can’t necessarily report the same metrics
- … Should we blend the metrics with their page load equivalents?
    - Similarity to BFCache and visibility changes
    - Many fast post load navs don’t compensate for one terrible first load
    - Bounce rates are impacted more by the initial page than by transitions
    - Distribution may change
- **Yoav:** first page is important, but isn’t that the same for MPA?
- **Cliff:** yes! performance is always a distribution. SPA laid is not the same of full page load, but it’s important to segment. Question for Pat Meenan about SPA and synthetic, because when you script something in WPT, LCP is still being reported for SPA navigations. Was just on a call with a customer that is moving to a SPA, but interested when the LCP/product image paints after the softnav. So they rely on synthetic metrics.
- **Pat:** In WPT, LCP may trigger if the largest in the new loads is larger than the previous loads. Element Timing is the only real way to instrument today
- … render metrics would be relative to the state of the SPA
- **Michal:** LCP candidates don’t get reported after interaction, no?
- **Npm:** that would work if the routes are triggered by script, not input
- **Pat:** yeah, that’s how it works
- **Cliff:** For CLS, is this something that can be cleared?
- **Michal:** CLS is not reported, so libraries can calculate it the same way Chrome does. But there’s work underway to normalize it over time and across routes
- … If we start attributing to URL, we’d want to cut it per route
- **Nic:** that’s what we do with Boomerang
- **Michal:** Depends if you want to match CrUX or improve attribution
- … Back to comparing initial load for MPA vs. SPA, there’s definitely a difference between first load and reloads, but suspect that SPAs would have bigger differences. Cliff mentioned that they do separate that out and dig deeper. Do you find it important to separate the metrics or blend them.
- **Cliff:** transitioning to SPA replaces traditional page views, so need to measure holistically

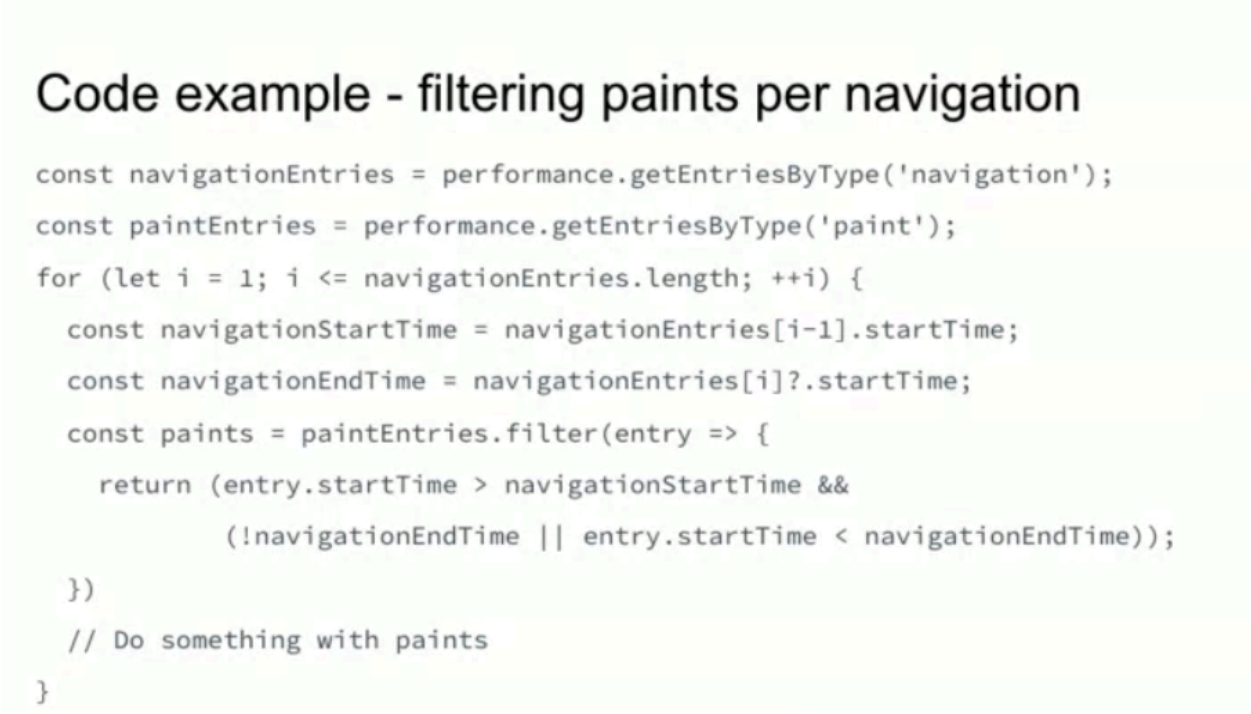
- .. the answer is it depends, but having the ability to separate that matters
- **Michal:** biggest concern is that the entr point for a website is paying the cost upfront and all future routes are near instantaneous. Blending is weird here
- Noam R: you need to picture both. You want the first load for business impact and subsequent loads to better understand the performance of the full user flow.
- … interesting to look at loads in the same session in MPA vs. SPA
- … SPA would give you better results in those cases
- **Yoav:** For SPAs, having the site fetch all possible content on first page so other pages are faster is something you can also do with MPAs with prefetch/etc
- **Cliff:** thinking of cold vs. warm cache, we don't really do much about that today. But the same is true for soft navigation, some of their resources may be cached. Important to distinguish page load types
- **Michal:** the best page load is no page load - so is that a perf regression of an improvement?
- **Npm:** We've always discussed SPAs, super clear that this is something that's needed. Main question - will we require developer annotation, new history API, etc? Also, what metrics do we want to surface - FP, FCP, LCP, FID?
- **Steven:** working with SPAs for 5 years now, and need an API to say "we're doing a softnav" to reset all the performance metrics. For navigation browser know things that they don't know for SPAs. Would love all the metrics to track specific SPA pages that need improvement
- **Benjamin:** How do you signal?
- **Steven:** today we do that in our library to reset the library instrumentation, but not the browser metrics. We change the URL, but heuristics may get it wrong, because the timing can change. Would love an API.
- **Michal:** Event Timing reports on all long events, FID can be polyfilled, but for first inputs after a transition, you want it even if it's fast, so it makes sense to reset in those cases and report the first event after softnavs.
- **Npm:** annotating the SPA navigation requires something different from Event Timing. API that changes the URL, e.g. discussions about a new Navigation api that can fix history API as well enable monitoring. Requires URL changes that are not suited for Event Timing
- **Michal:** issues around UX. Also, the history API is not great. Separate from that, if you want to support opening new tabs + client side routing requires a lot of work. Proposals to fix that. It wouldn't measure on its own, we still have to answer all these questions.
- … hardest for me to think through paint - can we just do the same things we do for the initial paint? Or should we just care about the new content?
- Noam R: If we go with the application clearing the metrics, the application can use Element Timing for something finer. Cannot rely on the history API, but can allow the application to explicitly reset the metrics.
- **Cliff:** Couldn't we tell during the navigation state, when we started the event and when the last layout shift happened and ignore it if the timestamp was before the start of the SPA navigation. Wouldn't requiring clearing anything, you could still mark a "next contentful paint" or trigger a new LCP. Doesn't seem that complex to do today.

- **Michal:** You're right that you could just use the nav timestamps to slice a session. For paint it's different. I like Noam's suggestion, but now you're measuring different things
- Noam R: we can call it "contentful paint"
- **Npm:** It's the first contentful paint after the SPA navigation
- **Yoav:** even without changing the name we can easily distinguish between one and the other, based on timestamps. Open question with paint metrics is whether we can report them for arbitrary things, from a security perspective. Killing :visited would go a long way to make that reasonable, but not sure that covers everything
- **Tim:** same process cross-origin iframes can also leak things
- **Npm:** if you have a cross-origin image you can get information on based on their painting times
- **Cliff:** is the idea to do something so developers are doing as little work as possible and get that out of the box? Or do we want a common way for developers to annotate so that we can report it everywhere?
- **Michal:** One idea is to come up with a User Timing convention, get the frameworks to speak the same language and hook into that. Relies on developer hints, so we'd have to see
- …. The next step is for that to inform better heuristics
- … and finally with the navigation API, it could restrict abuse: no reason to use it too frequently - only be used in places where top-level navigations are being used today
- … So other than just identifying - what do we measure and what do we do with it
- … Good to hear you want both blended view and separate first load view
- **Cliff:** This has come up and we will hear more from it. Web Vitals is still new, but it will come up
- **Nic:** For our customers, they can segment hard navs from soft navs, obvious that for soft navs the metrics are not there, and we have to educate them. Would be great to have a consistent view and metrics
- **Michal:** one concrete takeaway - maybe have the default be blended for MPAs and segmented for SPA. Would that minimizes surprises?
- **Cliff:** Sounds like a good blog post
- **Ryosuke:** Would be useful to better understand use-cases here with concrete examples: React app, twitter, etc and study what they're doing.
- **Michal:** I looked into that. Maybe we can have a session on a call to go over such an example.
- **Ryosuke:** Similarly, we can hear from e.g. SalesForce how they are measuring now. E.g. some apps change the URL after the navigation. There are multiple ways to write routers, etc. SPAs can also do partial navigations. Would be interesting to see ways that apps consider navigations. Not sure if there's agreement there.
- **Michal:** Tried to show this. Also history updating is a single moment in time, but some apps load pages through async processes.
- **Ryosuke:** There's also an in-app prefetch case, where they fetch content ahead of time. Specifically to SPAs we need to be able to measure the things that happened before the navigation.

- **Michal:** in the conversation on prerendering, Domenic mentioned that they want to measure both prerendering and the user experience. I haven't previously thought about attributing the prerender work to the cost of the future route. Right now, the thing that's doing the prerendering is the one to which the cost is attributed. If we're using browser features for prerendering, that may make it easier to attribute the cost.
- **Yoav:** AI to kick-off repo to cover the problem first, collect use-cases, properties can contribute from their experience what they're trying to measure
- Noam (in chat): I would give +1 to Steven's idea of an API which allows the app to notify that a "soft navigation" occurred. Relying on URL/history change is probably not sufficient for some apps.

BFCache Reporting - Yoav Weiss

- **Yoav:** We want to align user-experience metrics including BFCache
- ... Fire new entries for BFCache rather than overwrite old entries/timestamps
- ... Single time origin for all navigations
- ... Option 1: Another NavigationTiming entry, nicely uses NT array, no new entry types
- ... Compat questions, does anyone look at non-[0]th entry? 0.125% of Alexa 100K touch a second entry, most loop over all entries. None relied on being exactly 1 item.
- ... Downsides: Have a NT entry with a lot of 0-value properties (related to load time of resource)
- ... [code example]



- ... Option 2: New type of PerformanceEntry, doesn't include all of the zero'd values but contains the startTime

- ... Option 3: New entry, also has pointers to pains/lcp/longtasks/etc. But at the time this entry is created, we don't have all these times necessarily
- ... Option 4: PerformanceObserverEntryList on the entries to get associated ones
- ... "It's weird"
- ... How does it fit in with other observers
- ... Favorite as Option #1: compatible, fits with current API shape
- ... If so, should we also use it for other navigation such as SPA navs
- **Michal:** In your testing, when did sites check the entire array
- **Yoav:** During page load process, I looked at the code samples but didn't debug through them
- ... Those sites may not even see those BFCache navigation because it happens after the point in time they're collecting those metrics
- **Nic:** Is there anything else to track here as part of the duration, I'm assuming there's non-zero work here? Is there anything we need to measure?
- **Yoav:** We'd want to kick off FCP, LCP, FID, etc as well in that scenario, filtered to the timestamp
- **Benjamin:** Firefox notes when BFCache is used to restore, but we don't note a duration.
- **Yoav:** Would it be reasonable to re-fire paint entries and FID in case of restore?
- **Benjamin:** Possible yes
- ... We're more interested in when cache is used
- **Sean:** One thing I noticed is that all proposed options look different in what they're reporting. What problems do we want to solve?
- **Yoav:** I don't think they're different in terms of what they're reporting, main thing is that a navigation happened, and at the point it happened. Secondarily, we want to re-fire entries and filter them based on the start time of that BFCache nav.

isInputPending update - Andrew

- **Andrew:** Everyone's favorite boolean
- ... Recap: Was called shouldYield, hasPendingUserInput, now isInputPending
- ... Chrome 87+ Origin trial in 2019
- ... Overview: Way to remain responsive for work that doesn't want to yield
- ... Permits yielding less frequently if unnecessary
- ... Why? Input only because most display blocking work is due to input
- ... It does block painting/network and that's OK, not a substitute for yielding
- ... FB measurements and partners show it can save time and throughput by yielding only to input, throughput up 25% from one partner
- ... Interaction with LongTask API: Ideas to adding annotations to LT entries
- ...
- **Noam:** Can't say a session is bad because it had a LT, it might've been bad when there's user input
- **Tim:** Tricky cases are when others are using isInputPending for yielding, but some may be using isInputPending for other cases or not yielding
- ... Let people register for a different LongTask entry for those with isInputPending calls

- **Andrew:** Introduces the idea of a "sanctioned" LongTask
- **Patrick:** Problem coming in with Lighthouse, deadline with rIC. They may be checking the deadline but doing it so frequently. Maybe they check the rate at which they check isInputPending.
- **Andrew:** Interesting, though you could be checking it as frequently as you want but if you may not ever do anything based on that signal
- **NPM:** Maybe in LongTasks, an array of timestamps for isInputPending calls
- ...
- **Nic:** It would be interesting to mash wasInputPending with isInputPending calls to see if they reacted to input
- **Michal:** I know there are some RUM tools using TBT, and ...
- **Tim:** When you have moderate data there are cases where LongTasks give you data that you might not see otherwise
- **Michal:** The fact that the page called isInputPending gives confidence perhaps
- **NPM:** If it was a 2 second LT and they called it at the beginning or ending, that's not a good UX
- **Andrew:** Comes back to wondering if any annotation is useful?
- ...
- **Andrew:** InputStarved signal, where isInputPending was checked but it was more than X milliseconds before they yielded
- ... Will take some of these suggestions to Github issue
- ... Yielding to input: If yielding, how? setTimeout()/postMessage(), versus scheduler.yield(), or something new?
- **Scott:** ... (may not cover this case exactly)
- **Michal:** If the scheduler.yield() proposal is extended to handle this, yield and return to me with high priority, does it just replace isInputPending() entirely?
- **Scott:** Maybe. There may be problems with throughput where you don't want to yield that frequently
- **Tim:** My guess is we don't want to specify user will handle all user input before handling all micro tasks. Giving browsers flexibility to whatever they choose to do feels preferable.
- **Ryosuke:** Webkit does not have any mechanism to prioritize input
- **Andrew:** Is input frame-aligned?
- **Ryosuke:** Input is FIFO
- **Andrew:** We may not need a yield-only-to-input function
- ... I think Tim's suggestion of expecting reasonable behavior and if not default to a timeout in the worst case or potential throttling
- **Michal:** Presumably isInputPending() signal may give you confidence to introduce a longtask where you would yield. But there could be other scripts that want to run. What about yielding to other script and making sure a bad actor can't take over the thread.
- **Andrew:** IIP is for script that already wants to do a longtask and has incentive for throughput wins, improve responsiveness and UX. Cooperative multi-tasking is out of scope for this...

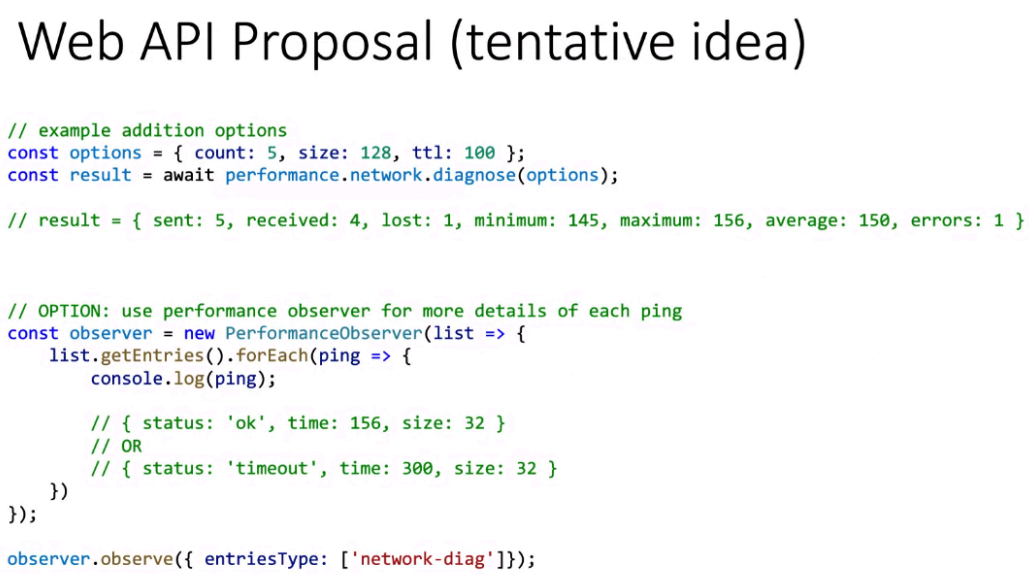
- **Scott:** Our team has discussed some options here, ..., we could lie in isInputPending if you're blocking the thread for too long and there are other things. Not sure if this is something we want to spec.
- **Michal:** isOkToBlock(), for future cases where it's not strictly related to input. In this case it's nice to scope it directly to the problem.
- **Yoav:** Regarding incubation status question, we had discussed it a few calls ago. One previous call it was favorable to consider for adoption, but a more recent call went the other way. Didn't want to make Rechartering dependent on incubations.

Wednesday October 21

Network Diagnostics API Proposal - Noam Helfman

- **Noam:** Not a spec or proposal, understanding a need for browser capability
- ... Browser provides very limited network diagnostic information
- ... May want to optimize UX based on network conditions
- ... Key use case is local and last mile network conditions
- ... Web app is required to determine state of local connection, enables app to notify user of local connectivity issues
- ... App could suggest user move to a better reception area
- ... App could adapt (payload size, call frequency)
- ... Can collect this as RUM data
- ... Existing solutions: NEL (no local network info, no client-side info)
- ... Network Info API (too much or not enough sensitivity, not configurable)
- ... Custom solution (via XHR calls, no local conn info)
- ... Possible approaches. Approach 1: Network Info API. Extend current API to configure sensitivity thresholds (RTT to consider timeout, consecutive timeouts trigger notification)
- ... Limitation is in flexibility, since different apps may have different needs (e.g. games requiring lower latency)
- ... Approach 2: ping API. Attempt to diagnose local last mile connection -- pings default gateway
- **Yoav:** For Network Info API, the idea of custom thresholds is interesting. Current API exposes to many bits of entropy and doesn't give enough info in other cases.
- ... Would love to have your active involvement on the repo -- current WICG only
- ... Repo was not adopted as part of Devices and Sensors group
- ... I think there's a path forward to modify the API to address concerns from other vendors
- ... Could allow people to gather a small number of bits, but the bits they care about
- ... For Ping API, knowledge of how far away I am from my router and providing that to random web pages seems risky
- ... Would like to split out use cases where there's a need for local network information from general how is the network behaving
- **Noam:** Let's cover privacy aspects in later slide

- ... [example]



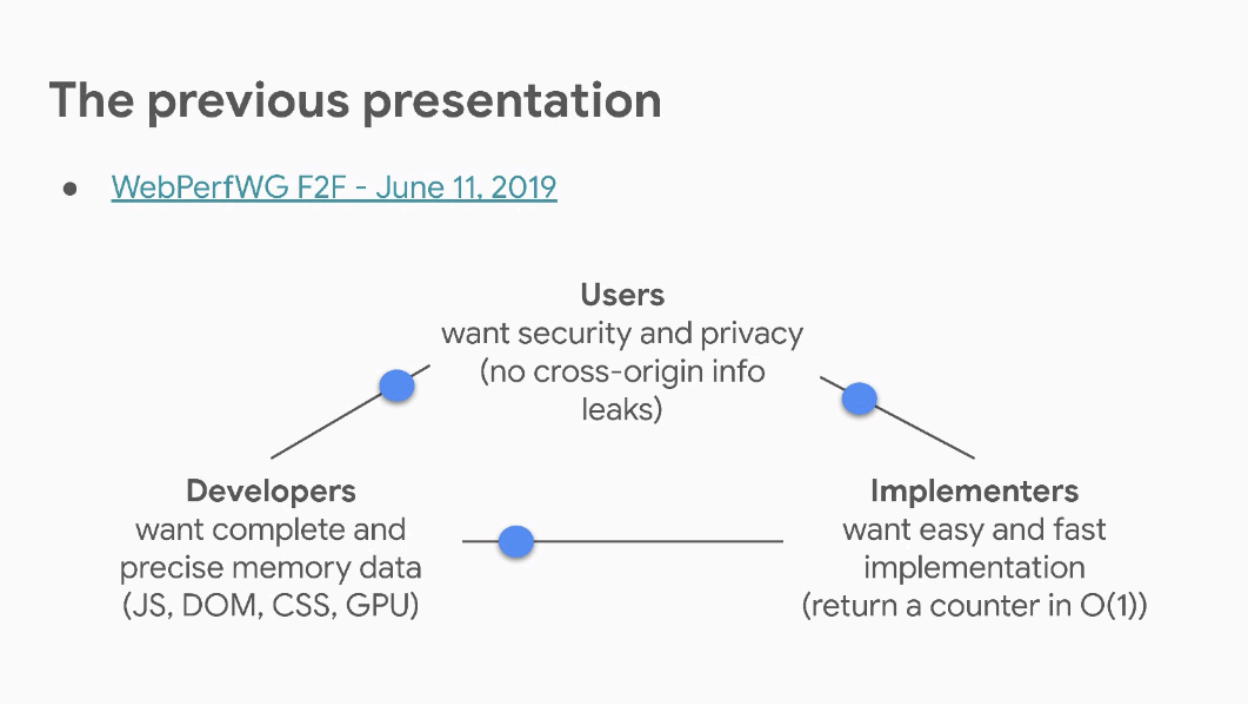
-
- **Yoav:** I think the API shape will be overshadowed by privacy issues
- **Noam:** Privacy concerns. Should user receive prompt to acknowledge and accept diagnosis. Apps already diagnose using hacks via XHR, image load timing, etc.
- ... What is valid to report without violating privacy concerns
- **Yoav:** Main difference I see here is you're pinging a specific local IP address that is not exposed
- ... Today web content does not know what the local gateway is
- ... Adding an explicit way to ping the default gateway could be a concern
- **Noam:** Another issue is exposing larger fingerprinting surface
- ... UA should not report the default gateway IP address
- ... Obfuscate ping results to mitigate fingerprinting
- ... NetInfo API rounds to 25ms, could be more or less granular than that even
- **Subrata:** How is it going to help in a VPN-connected network?
- ... Second question is -- we already have a RTT, how does ping response time differ?
- **Noam:** Regarding VPN, that definitely changes routing table
- ... More than 1 gateway can exist (default gateway would go to VPN IP)
- ... Was going to consider diagnosing even more destinations, ping FQDN from site origin only
- ... TCP ping to detect network congestion -- harder to define and do securely
- ... Main concerns I'm hearing are around privacy, is that correct?
- **Yoav:** Concern around privacy with new information being exposed. Maybe that could be put behind prompt?
- ... I would like to understand the use-cases for this to be flushed out, versus NetInfo for local diagnostics
- **Noam:** Does anyone else here have a need for diagnostics of local network?

- **Patrick:** Are the diagnostics here the responsibility of the web app, or more browser responsibility? More consistent, but gives the same user feedback.
- **Noam:** If the app detect bad connectivity, it could adapt, change the payload or send data less frequently
- **Yoav:** But that could be covered by NetInfo API too, not necessarily local NetInfo
- ... One note from Alex Russel in chat is that an async API that provides a prompt could be a good idea if this moves forward
- ... Going back to NetInfo, wondering if other vendors have feedback on what a revamped API might be something they want to ship
- **Benjamin:** We're all on-hands for QUIC, so this would be on backburner
- **Yoav:** Would this be something you'd eventually be willing to expose
- **Benjamin:** Maybe, but let's cut speculation short, we can't commit to this
- **Sudeep:** I'm Co-Chair of Web and Networks Interest Group
- ... For NetInfo API, there are some proposals in 5G space with a lot of variations seen. Whether hints can be shared with apps, for doing buffering in advance, etc. Has dependencies on information from operator network, etc.
- ... Second comment is orthogonal, there is an idea in our interest group where it's the other way around, to extend developer tools to extend time-variant conditions. There could be a network trace format where it's taken from the real-world and test how a web app behaves in network conditions, so the developer can adapt to network conditions.
- **Yoav:** Network trace idea sounds interesting in context of synthetic testing, but possibly orthogonal to this. If you could share links that'd be great.

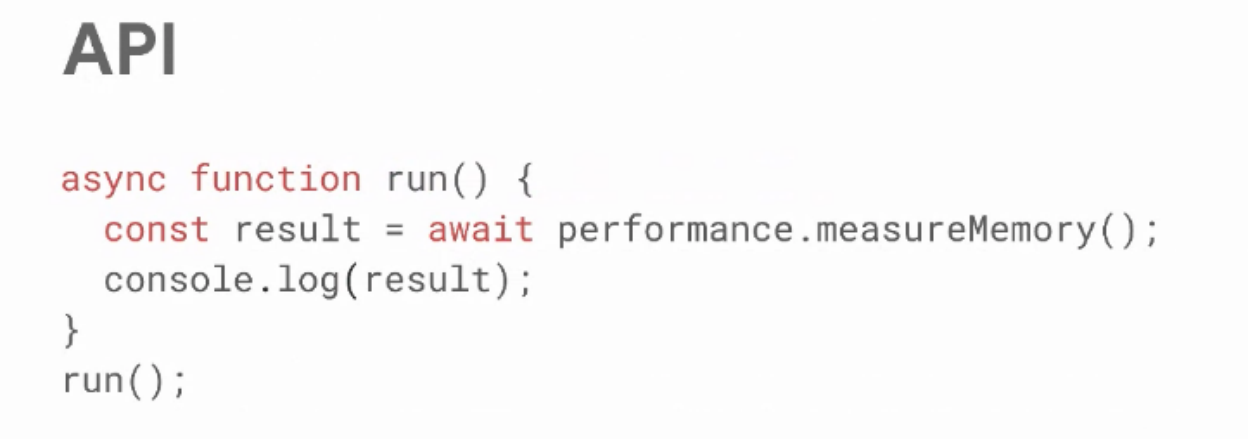
performance.measureMemory API update - Ulan Degenbaev

- **Ulan:** Why care about memory? Use more memory to improve performance
- ... Use more memory due to memory leaks (slow growth over time, slowdown due to paging, garbage collection)
- … Hard to detect in local testing, which run for short times or as a result of specific actions
- … Main use case - how to measure memory usage in production
- ... Useful for long running complex applications

- … Tradeoffs

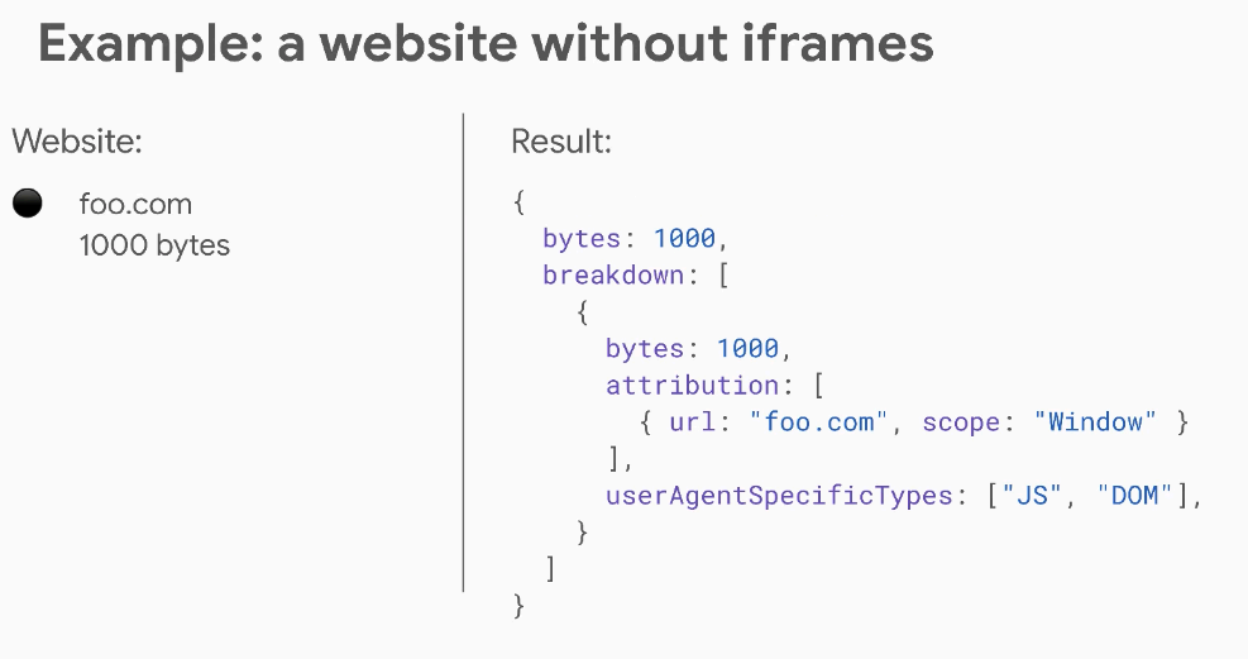


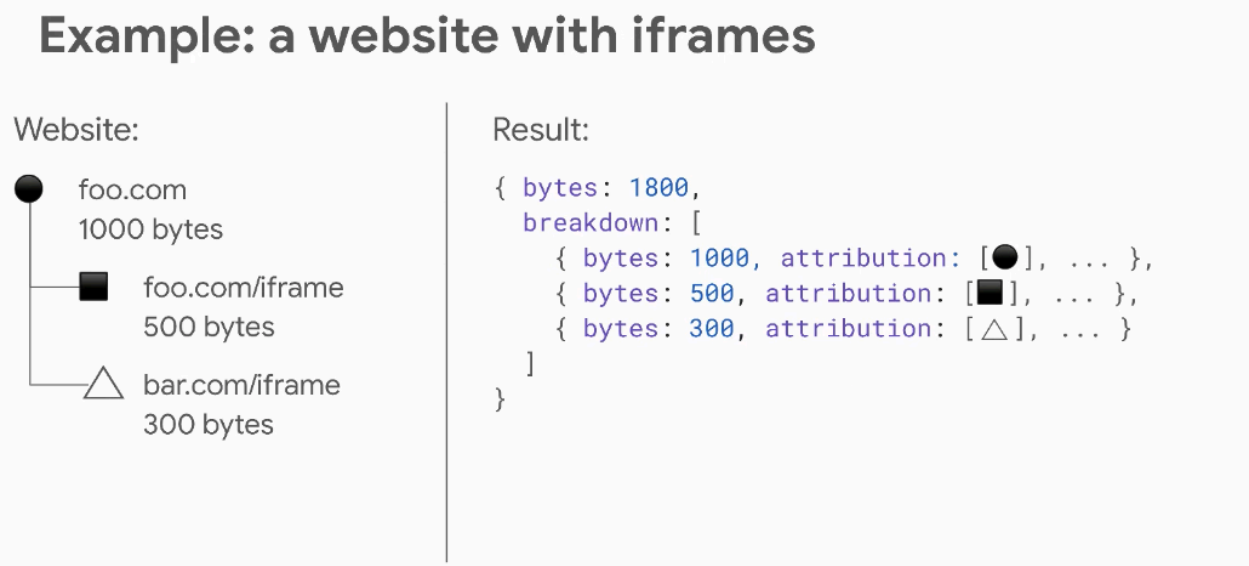
- API shape



- … Comparison to the non-standard API: current API is well defined, where the previous API may have leaked heap sizes from unrelated pages, based on implementation
- … Current API enables to break down the usage by owner (e.g. iframes) and type
- … Better security through cross origin isolation
- ... An attacker can use the API to infer sizes of cross-origin resources, gate API behind self.crossOriginIsolated via COOP/COEP

- … Breakdown example





- … The browser could also group together attributions in cases that it cannot distinguish between them
- … We’re ok to expose sizes of iframes but not ok with exposing URLs or strings
- … For same origin we can provide the most recent URL (post redirection), but for cross-origin we can only provide the pre-redirect URL
- … Origin trial is running in Chrome 85-87. Currently gated behind site-isolation, using simpler attribution format, can measure worker memory
- … Revamping to new attribution format, hoping to ship in Chrome 88
- … Got useful feedback from Mozilla, some concerns about interop that results in renaming to “userAgentSpecific*” to prevent confusion
- … Suggestion to add dummy entries to prevent users from relying on specific structures

- … Another open question - what should be the scope of API? Whole website or the parts that are in the current process?
- ... Feedback from users - Promises take a long time to resolve - because the measurement happened as part of GC. Local testing can force measurement using a flag
- … Other feedback resulted in adding worker memory
- **Rniwa:** What bytes are you reporting?
- **Ulan:** Sizes of objects allocated by that process
- **Rniwa:** What's the reported size? In many OS you have a memory compressor, so the bytes in the physical sense may differ from the object size? Also dirty bytes from non-dirty bytes can differ. You can have a completely empty page that's dirty.
- … Also, how much of the memory is active?
- **Ulan:** In Chrome we don't try to approximate the physical memory usage, but report the sizes as we allocate them. If the OS does something fancy, that wouldn't be captured.
- … Useful to avoid exposing system info due to fingerprinting concerns
- … To avoid fingerprinting, we should not surface memory that the webpage doesn't allocate
- **Rniwa:** For example if you have a blob and map the contents of a blob
- … So the definition is implementation dependent, we could report anything
- **Ulan:** Allocate a random buffer, should not be completely random
- **Rniwa:** Need to agree what kind of bytes we're reporting
- **Benjamin:** Great work on this! CORS check is great progress. Can we make progress on the open issues?
- **Ulan:** 3 open issues about interop.
- … Scope somehow related to security, smaller scope would provide better security, but results may vary more per device
- **Noam:** In terms of attribution, does the API distinguish between JS, images, GPU, etc?
- **Ulan:** Currently only JS memory, but plans to add DOM and GPU memory
- **Noam:** Would the GPU memory expose the total available GPU memory? In contrast to RAM, where the OS uses paging, full GPU memory can cause bad things. Would be good to get full GPU memory.
- **Ulan:** We could maybe approximate it. Limits of GPU memory is out of scope, different privacy and security review
- **Noam:** In practice it's already exposed by allocating canvases
- **Alex:** Have there been any success stories of real regressions found using this?
- **Ulan:** Origin trial has been running for a while but with the old scope, latest changes not available yet, maybe in a month or so
- **Alex:** Such a story could make this more compelling
- **Ulan:** Questions about scope?
- **Domenic:** Found arguments compelling
- **Rniwa:** Concerned about exposing memory from cross origin iframes
- **Yoav:** To emphasise, this is only for pages that are cross-origin isolated
- **Rniwa:** Concern still stands about exposing memory information for cross-origin iframes
- **Ulan:** For information leak?

- **Ulan:** It won't be observable if they are in the same process or not
- **Rniwa:** Have to read the proposal

Rechartering, Call for Editors - Yoav, Nic

- **Yoav:** We were scheduled to be rechartered back in June, extended old charter to accommodate Process 2020 to Dec 31 2020
- ... Desire to move to Living Standards, and CR draft model
- ... Get to CR and update the CR once in a while as needed
- ... Draft charter
- ... Doesn't change much around deliverables
- ... Changes to scope around making it more explicit, previous was vague around somehow improving the user experience
- ... Splitting categories of improvement into measurement, scheduling and adaptations
- ... This covers all deliverables we have today as well as what we had in the past
- ... Can we publish pre-rechartering? ResourceTiming L1, PageVisibility L2 to REC
- ... Post-rechartering, Beacon to PR and requestIdleCallback to REC
- ... Should we remove preload and resource-hints as deliverables
- ... Moving resource hints directly into HTML
- ... Unclear if we remove them entirely, or state that they're deliverables in transition
- ... For everything else, get to CR, make changes to CR-draft, iterate
- ... We have some specs that could use adoption for editors: Server Timing, Long Tasks, requestIdleCallback, ResourceTiming+NavTiming rewrite
- ... Level of effort varies based on the spec. Even if you have a few hours a week that you could dedicate to the subject it would be appreciated and could use your help.
- ... Make sure issues don't lag
- ... We can help you get started and be successful
- **Carine:** First question was about going to REC before the beginning of the year, for RTL1 and PVL2
- **Yoav:** RTL1 is complete and PVL2 has one open issue that we could resolve rather quickly
- ... Otherwise we can leave it as a deliverable and aim to get it next quarter
- **Carine:** We can't remove them from the charter because the working group needs to maintain them
- **Yoav:** We plan to keep maintaining them. RTL1 would be done, but RTL2 will not be. We have both in the charter. Committed to keeping RTL2 as a living standard.
- ... PageVisibility we think would be similar, L3 ongoing as a living standard
- ... Clean up the fact that we have two different levels as a deliverable
- **Carine:** We have 2.5 months, it will be tricky but possible
- ... For RTL1, are there modifications, do we need a new CR?
- **Yoav:** I don't think so, we just forgot to move it to REC
- **Carine:** Yes, we should be able to do that by EOY
- ... For PageVis, do you want it as a Living Standard?

- **Yoav:** Ideally we could have L2 go to REC, then L3 be a living standard for anything we add in the future
- **Carine:** Do you anticipate L3 would be highly different than L2
- **Yoav:** Potentially we would be interested in adding more mods or signals (RE: prerender)
- ... There will be an API change
- ... Do you prefer we just take L2 to be living standard?
- **Carine:** That's a possibility, not necessarily the right thing to do, depends on the approach for versioning
- ... We use levels for the moment, but we didn't have living standard before
- **Yoav:** Either way we want it to be more backwards compatible
- **Benjamin:** Thanks for this discussion, I wanted more clarity about the living standards part.
- ... More clarification about how this evolves. Does that mean we can't remove interfaces?
- ... Can we just call it Page Visibility 2020?
- **Carine:** Difference between going to REC the old way and now as Living Standard
- ... When you enter PR, you have to say it's going to become a Living Standard
- ... When you're in PR as that, you go through a different process. Which combines patent protection done at CR and the AC vote at PR.
- ... The other change is the CR draft and snapshot are distinguished
- ... Where before some CRs were editorial and some were substantial
- ... The only thing about living standard is about amending and changing the recommendation in place
- **Benjamin:** And where is the point of comments to indicate a change in living standard
- **Carine:** There's something in the process called "last call for review"
- ... And that will be a CR/PR mix of review
- ... Normally that should lead to recommendation that will result in a new spec with additional content
- **Yoav:** In previous discussion we concluded, the Living Standard variant of amended REC would be something that has relatively high overhead, so we'd prefer the CR-draft version. Draft snapshots will be the tip-of-tree.
- **Carine:** Yes, previously if you wanted to add a feature to a REC, adding an API to something that already exists, you had to go to PWD, an entire new track. Now if you're marked as Living Standard, you can incorporate that and directly go to CR. CR are snapshots with patent policy protection.
- ... You don't really need to go to REC again, unless your spec is relatively mature
- ... You don't have levels, but when you think your L2 is done you publish as REC then work to L3 as CR directly without going to PWD
- **Yoav:** I think that piece we'll have to think about closer to an existing example
- ... Maybe we can try that out with PageVis, move it to REC as living standard, then move it back to CR to add prerendering bits
- **Carine:** That does not prevent publishing L3 as WD

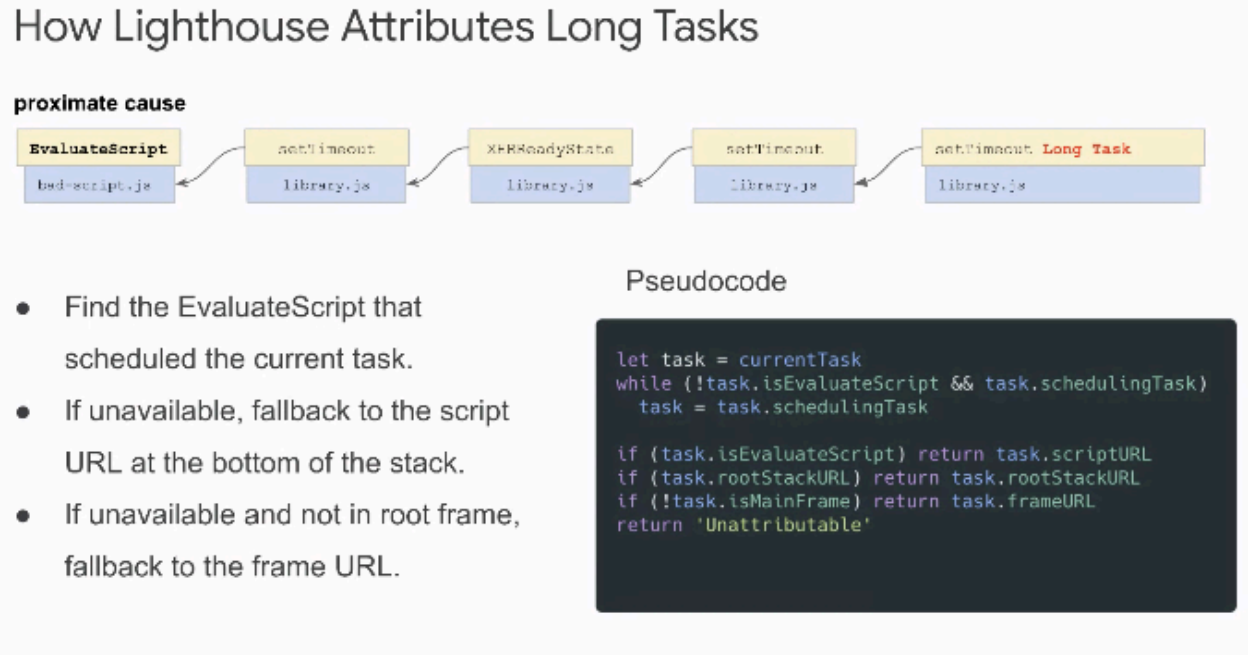
- Michael Smith: One thing I wanted to talk about was something Benjamin asked specifically
- ... Worked for W3C on a lot of transitions
- ... Benjamin asked is it considered a breaking change if you remove an API
- ... The general answer we have is that if any change will invalidate someone's previous review, then you should give your reviewers an opportunity for review
- ... Think about if you reviewed someone else's spec, if you find out after the fact they made a change afterward and went ahead that invalidated the previous review they did, that wouldn't be considerate
- ... Just think about if they'd want an opportunity to re-review
- **Benjamin:** How does disposition of comments work then
- ... From TAG review, that seemed ad-hoc
- ... I want to make sure there's a place where these re-reviews can happen
- ... I just want to make sure we track these reviews and comments
- ... So people in other WGs can see what we're up to
- **Yoav:** Do you have an example of other specs that follow that?
- **Benjamin:** I do have an example, I'll follow up and provide one
- **Yoav:** That sounds generally useful
- **Benjamin:** That would track things in a changelog in the doc
- **Yoav:** The other question was about removing preload/resource hints as deliverables, or can we add them as a separate section about deliverables-in-transition, for things that are in-flight between W3C and WICG
- **Carine:** If we don't plan on delivering them from this group, it should be removed from deliverables in this section
- ... For AC review of this charter, it would be nice for them to know what is happening for these specs
- **Benjamin:** Is there a section non-normative text
- **Carine:** We don't have a section, but more as a freestyle section of the charter
- ... Either under "other deliverables" or "liaison to other groups"
- **Michael:** To clarify, is the content of those specs being integrated into HTML Spec?
- **Yoav:** Yes
- **Michael:** I would say when that does happen, along with whatever else you do, please make sure we know what's going on, especially relationship with WHATWG where we coordinate
- **Yoav:** Yep, Philippe is aware
- ... Review the draft, and add comments there
- ... We'll kick off rechartering soon

Long Tasks attribution - Patrick Hulce

- **Patrick:** Working on Lighthouse. Not a formal proposal, but want to describe how Lighthouse tackled this problem and if we can apply it here
- … Lighthouse is a lab tool that captures performance metrics. Identifying the cause of main thread work is really critical for us.

- … Knowing that a lot of work happened is not useful on its own. People want to know why!
- … Developers need to know who's responsible for every single long task
- … We want to find an "proximate cause" for work. Which is different from why is the long task actually long. (which is the current focus of "attribution" in the LT spec)
- … Want to walk through motivating examples
- … Terms
    - EvaludateScript - initial script execution
    - Self-time - time spent executing while source is at the top of the stack
- Examples
    - Badscript.js causing browser layout thrashing
        - Self time is not good here
        - It caused a lot of layout work
    - EvaluateScript with a library
        - Most time spent in jQuery, but badscript.js is calling jquery and is the cause
        - Maybe we want to top of the stack
    - setTimeout with a library
        - No stack!!
        - Need to find the script that scheduled the setTimeout
    - Chained Script
        - What if another script adds badscript.js?
        - In lighthouse we stop at a script with a url, so in this case badscript.js is the cause, not the script that inserted it
- … Observation - attribution should take causality to the extreme, the "document" could be the cause of all the long tasks
- ... Taken not far enough, and the "browser engine" is the cause of all.
- … In Lighthouse, cross-origin leaks are not really a concern, and have access to a profiler

- … They attribute long tasks by walking back to tasks that eventually triggered tasks



- … Works fine for setTimeout and XHR, less so for fetch() and event listeners, due to implementation deficiencies, but cannot provide attribution for DOM mutations or non-JS async work
- … 99.6% of LTs get attribution, 96% have all LTs attributed
- … Been using it for 2 years, and it seems to meet developer expectations
- … That method of attribution doesn't over-attribute work to polyfills and monkey-patches
- … But, scheduler scripts see inflated attribution if work is pushed through a queue, which is not tracked
- … Also, don't know how it would work for post-interaction work, as LightHouse doesn't do that
- … Open questions: Is the model of proximate cause for attribution widely applicable to other things?
- ... Could this be implemented cross-browser?
- ... Would this model yield developer value for other APIs like Long Tasks or measureMemory after cross-origin concerns are taken into consideration?
- **Yoav:** Wondering whether the pieces where we cannot attribute are fundamental, or just implementation driven. In particular, scheduler scripts and work being pushed from a queue. Can that work be attributed to avoid over-blaming scheduler scripts.
- **Patrick:** I think the scheduler problem needs to be solved at least partially via some sort of developer annotation. It needs to say it's work from one context or another.
- ... Automatic attribution, even if it were heuristic based, might be lower quality
- **Tim:** Talked to V8 team about tracking each variable and what's mutating it, and their feedback was it was expensive
- **Michal:** How much do you need stack sampling here?
- **Patrick:** Standard auditing isn't enabled by default for traces, so it's just a bonus to narrow down what's happening. All annotation here is micro-tasks relationships.

- **Yoav:** Regarding last question and cross-origin concerns, if we limit information which is just the pre-redirect URL, which is already information the page has or can obtain, a page could stagger the loading of different scripts to figure out which script is triggering a long task. Seems like something people can do today, regardless of the LT API, just setTimeout timing attacks.
- **Michal:** We already talked about your talk for interactions/responsiveness. It's possible that EvaluateScript is happening in cases like click handlers, etc, rather than attributing longtasks back
- … Any future async work that's in the chain could be attributed to the initial work
- … So we could use this for responsiveness
- **Patrick:** Almost like taking the extreme example that it came from the document
- … The correct information can be useful for other scenarios, excited to see how you'd use it.
- … Also, opinions from RUM providers?
- **Nic:** For Akamai mPulse, we're capturing LTs in the world, but have no attribution. Customers always ask what they can do with it. More things we can track from RUM to point fingers can save time and help customers find culprits in the wild. Any more information on LTs would make the data way more useful. Right now can't say more than "this happened". So attributions is a major request for us.
- … Also, what 3P are doing on the page can help folks from stopping to point fingers on the mPulse script as the culprit as we wrap top-level APIs
- **Patrick:** Another useful need for that attribution could be in the case of exception and stack traces. Would be useful that have traces go beyond async handler, what script added the handler
- **Carine:** Mentioned cross-browser implementation feasibility. Did you investigate?
- **Patrick:** I suspect it could be, but would love to hear from browser vendors. Task coalescing can be different
- **Yoav:** Other browser vendors, any hunches regarding feasibility?
- **Alex:** Sounds difficult and potentially expensive to implement, but no fundamental problems. Those are big hurdles.
- **Benjamin:** We're behind on Long Tasks, so I choose not to speculate here.
- **Yoav:** Regarding cross-origin concern, we can follow up with security folks
- **Patrick:** What would be a reasonable next step?
- **Npm:** Can you file an issue on the LT repo to show the use cases you're addressing?
- **Patrick:** SG
- **Tim:** As Alex mentioned it's not cheap for us to implement, so want clear answers to cross-origin concerns. Having more data to gain confidence would encourage us to say it's worth further investment
- **Npm:** Cost is unclear to me as well. Ulan - who can we ask?
- **Ulan:** <name dropping>
- **Tim:** Also talk to Scott on Scheduling
- **Michal:** Given that every macrotask already attributes its source?
- **Patrick:** With tracing enabled
- **Michal:** Need to check if the data that goes into the trace is available

- **Npm:** Yeah, need to investigate

JS self-profiling update - Andrew Comminos

- **Andrew:** API to sample client-side JavaScript
- ... Just like DevTools
- ... Chrome OT M78-80
- ... API changed to now require COOP/COEP
- ... Can contain stack frames from X-O resources
- ... [example trace]

Trace example


```
{
   "stacks" : [
      { "frameId" : 0 },
      { "frameId" : 1, "parentId" : 0 },
      { "frameId" : 2, "parentId" : 1 },
   ],
   "samples" : [
      { "timestamp" : 1551.73499998637, "stackId": 2 },
      { "timestamp" : 1576.83999999426, "stackId": 1 },
      { "timestamp" : 1601.90499993041 },
   ],
   "frames" : [
      {
         "name" : "b",
         "uri" : "https://static.xx.fbcdn.net/rsrc.php/v3/yW/r/ZgaPtFDHPeq.js",
         "line" : 23,
         "column" : 169,
      },
      {
         "name" : "l",
         "uri" : "https://static.xx.fbcdn.net/rsrc.php/v3iMKu4/yW/l/en_US-i/gSq3sO3PcU1.js",
         "line" : 313,
         "column" : 468,
      },
      {
         "uri" : "https://static.xx.fbcdn.net/rsrc.php/v3iMKu4/yW/l/en_US-i/gSq3sO3PcU1.js",
         "line" : 313,
         "name" : "a",
         "column" : 1325
      },
   ]
}
```


- ... Github repo: https://meet.google.com/linkredirect?authuser=0&dest=https%3A%2F%2Fgithub.com%2FWICG%2Fjs-self-profiling
- ... Origin Trial feedback: Found and resolved a ton of performance issues at Facebook
- ... Commonly useful for interactions
- ... Testimonials in Github: https://github.com/WICG/js-self-profiling/issues/21, https://github.com/WICG/js-self-profiling/issues/24
- ... Highlighted need for API to be very performant
- ... Activation: Profiling requires work and we need to figure out when to do it
- ... Wanted to use Feature/Permissions Policy, but lack of support in workers and no “disabled by default” option shut that down
- ... Options:
    - Script-driven warm up? Would need to be done on the main thread. A lot cheaper to do this online than stopping the world and re-doing.

    - Amend Permission Policy. Needs to support disabled-by-default params, worker propagation
    - Add a new header
- **Yoav:** What are the blockers for Permissions Policy?
- **Andrew:** disabled-by-default use-case is fine, but shared worker issues could open up dialog
- **Yoav:** Take AI to start dialog
- **Andrew:** We can take that offline then for now
- ... Also want to solve the hardest problem in computer science
- ... Naming is hard
- ... No other spec uses the label "JS"
- ... What is "self-profiling" anyways?
- ... "JavaScript Sampling API" proposed
- **Michal:** Would like to think about what someone outside of this context would think of the name. Profiling sees bad.
- **Andrew:** To answer Carnine's question "sampling does not mean anything precise to me", "sampling" in isolation maybe doesn't mean anything, JavaScript Sampling itself is pretty unambiguous to what it refers to
- Michael Smith: The word "Self" isn't in any other of these candidates
- ... Because it's fundamental to the whole feature, the application has APIs to measure its own performance
- ... Don't have a strong opinion
- **Yoav:** The "What is self profiling anyways" comment resonated with me
- (some other naming suggestions in chat)
- Paul Irish: Does the API differentiate between idle time and spent in recalc style/layout
- **Andrew:** It's just script for now
- **Paul:** In the spec right now there is a Profiler Trace dictionary, may want to change the name of that interface
- ... Anyone who engages with the API can see the result is samples
- ... Priority would be communicating it's just JavaScript (and not layout, idle, everything else)
- **Boris:** It really depends on whether you want to limit the potential of the API. If in the future it also allows to view layout profiling, Performance Profile API sounds better.
- **Andrew:** In the future we may want to have potential of other things to sample
- ... I like the idea of Performance Profiler API
- ... TAG review has finished
- ... Would like to see comments in Origin Trials survey
- **Michal:** I was confused by "self" in the name as well, is it there's a special "self" time or that a web developer can trigger it via an API
- Michael Smith: Encouraged by the API itself, how it can be useful for developers, but concerned Apple would not be willing to implement it at all.
- **Ryosuke:** Our position has not changed
- **Andrew:** Is the concern around smaller sites?
- **Ryosuke:** API could incur too much runtime costs

- **Andrew:** Cost for startup, or runtime?
- **Ryosuke:** This is not something we will implement, we will strongly object
- **Michael:** I trust your sense of the cost of this, are you confident the cost is so much it's a non-starter.
- **Ryosuke:** We're talking about something that could be a cost of 10%+ and a power cost. We're not adding a feature that drains a user's battery.
- **Andrew:** You can get a lot of signal from a small percentage of users, and from that small sample you can fix major performance problems.
- **Ryosuke:** I understand that and disagree with that proposition
- **Alex:** There are some websites with smaller users than Facebook
- **Andrew:** In which case they would not need this API
- **Ryosuke:** Or they enable it once and forget about it
- **Andrew:** There are a lot of APIs that are footguns
- **Yoav:** The COEP requirement in the short-run is going to be a hurdle for any site that incorporates third-parties or X-O resources
- ... If the concern is around developers setting and forgetting it, we could incorporate sampling into the API itself.
- ... As a developer it could say I want it on N% of visitors or users
- **Ryosuke:** We don't want to encourage A/B sampling behavior
- **Patrick:** I don't know if this is the case for Webkit, there's probably a non-zero cost to having that code in there at all, versus having the code in the engine
- **Andrew:** We managed to mitigate a lot of this because we legeraged the sampling profiler already built in
- **Michael:** There are a number of sites that are on roughly the same scale that could benefit from this. If we're trying to fix experiences for users of the web, similar websites from other companies can help for a massive number of users.
- ... It would be nice to see if there's a way we can get this implemented for the web platform
- **Ryosuke:** Our approach to perf is not to let websites optimize for our engine, it's for our engine to optimize for content out there.
- ... We just don't do this type of A/B testing in general in our products
- **Andrew:** A/B could be otherwise considered we are granting the website this much resource or signal to gather this data
- **Ryosuke:** Which is what we don't want to do
- **Andrew:** Just speaking from a product POV we've found this to be massively useful
- **Gilles:** It would be useful for us at Wikipedia since we're seeing such variations in JS users are running, we even let users running their on JS on the site. So depending on what article or user, the difference can be significant.
- ... This would allow us to sample random pages and look at patterns
- ... I think for any website that has different users getting different experiences, at scale, this could be useful
- **Yoav:** Wikipedia is another good example where they can abide by COEP COOP restrictions due to a lack of third-parties
- ... With that, we're out of time, see everyone tomorrow!

Thursday October 22

TAO and CORS/CORP opt-ins - Yoav

- **Yoav:** We have a lot of opt-ins
- ... They're not well layered, and developers may need to set all of them
- ... Timing-Allow-Oigin, Cross-Origin-Resource-Policy and Cross-Origin-Resource-Sharing
- ... Information types: For resources, we have resource-level timing, origin-level timing and user-network-level timing
- ... Resource sizes
- ... Caching information (via transferSize)
- ... In the future, we have feature requests for new info, like HTTP status codes or content type
- ... Plus reporting of frame timings to its parent
- ... Proposal: Timings: Provided by TAO, could also be provided if CORS, may need a separate host-level opt-in
- ... Sizes information: Don't want to allow for TAO enabled resources, but do want to allow for CORP and CORS
- ... Information already provided via CORP and memory APIs
- ... Caching: Expose via CORP
- ... Metadata such as status code and content type: CORP maybe
- ... Parent frame reporting - something else entirely
- ... nextHopProtocol is behind TAO, but that doesn't make sense because it's information already observable and exposes information about the user's network (e.g. proxy downgrade)
- **Ryosuke:** How is it observable?
- **Yoav:** If you trigger more than 6 requests and look at the network characteristics
- ... e.g. HTTP/1 vs. HTTP/2 characteristics are different from another
- ... HTTP/3 vs. HTTP/2 may be more complex
- **Nic:** For clarification, are you saying if CORS, then you'd have access to timing, sizes, caching, metadata.
- **Yoav:** Correct
- **Camille:** So CORS becomes a mode where you say that the resource becomes similar to a same-origin resource for the specified origin.
- **Yoav:** In a way. Since CORS provides byte-for-byte access to these resources
- ... All of this information could be polyfilled without ResourceTiming
- … Calling it same-origin may confuse people regarding the credentials handling
- **Camille:** In terms of information about the resource it would behave similarly to a same-origin resource

- **Ryosuke:** I'm skeptical that because CORS allows resource to be read, that all these things are observable
- ... Things like size for some types aren't observable
- **Yoav:** Let's talk about timings first maybe
- ... responseStart and all those points in-between, related to resource-level loading
- **Ryosuke:** Opt-ins for CORS never agreed to that timing information
- ... CORS has existed for decades, and it's problematic to say this thing now provides a new thing. Bad to have an existing feature do more stuff and make it less secure.
- **Yoav:** My claim is it's not less secure because it's already exposing all those things, and you can see when each byte has hit the client
- **Artur:** Could you talk about the delta between what's exposed right now and the difference between CORS
- ... If we see everything behind TAO is also observable via CORS
- **Yoav:** Looking at ResourceTiming and points in time it enables
- ... redirectStart/end, domainLookupStart/end, connectStart/End, secureConnectionStart, requestStart, responseStart, responseEnd
- ... Without TAO all we have is startTime, fetchStart and responseEnd
- ... TAO gives us opt-in to everything in-between
- ... If we're talking about what does CORS enable us to observe, I believe it will enable us to observe responseStart,
- **Ryosuke:** I think it's problematic to expose how much TLS or DNS took
- **Camille:** Seems to me as well it'd be hard to see how CORS would enable that
- **Mike:** There's information about what the server is doing, versus how the network responds. CORS wouldn't give worker, app cache, DNS, TCP.
- ... Network considerations that seem different from server considerations
- ... CORS lets you get access to all the bytes
- ... Not necessarily to how to get all that information to you
- ... Doesn't seem CORS gives you any more work to have access to network conditions than how TAO does
- **Camille:** One more question: if TAO only gives you timings, that seems like a simple enough mental model vs. having implication on parts of TAO that seem much more complicated
- **Yoav:** the motivation is that we have low adoption of TAO even for static, public resources, and introducing CORS into the equation would significantly increase our visibility to what 3Ps are doing, without compromising security
- **Mike:** Agree for things that the server has control over, but disagree on things where the server is just an end point.
- … Characteristics of how bits come to you, that's CORS
- ... Different than DNS, which has nothing to do how long the resource took
- **Yoav:** Agreed those are different types of information (resource-leve, origin-level, user-network level)
- **Mike:** Concerned more about network level characteristics than origin-level
- ... The DNS-level data seems different nor that CORS would let you see that data
- ... I think that's also true of nextHopProtocol

- **Yoav:** Yep
- **Camille:** If we're thinking about this for opt-ins, and if you have TAO, and without it, you don't have timings. If I want timings, I have TAO.
- ... versus the implications if you have CORS that you get parts of timing
- **Yoav:** Motivation after many years we still have low adoption of TAO
- ... Even if we have static resources that are not sensitive and no one cares about timing
- ... Having information to these shared public resources would give better insight into third-party visibility
- **Mike:** I agree for things the server has control over (resource-level), origin-level we could talk about, user-network level I don't see the justification for that
- **Yoav:** Moving to sizes, sizes are exposed with TAO, and what I'm proposing is that we would stop, and would enable browsers that haven't implemented sizes would be able to.
- ... We would move away from a TAO opt-in to a CORP opt-in. Not sure if on its own it would give you the size for any random attacker
- **Mike:** The reason CORP is used for cross-origin isolation and not CORS, is we wanted to create a distinction between reading all the bits and content, versus the ability to use that on your website.
- ... Ability to include on your page, but not understand what it is
- ... So both sizes and metadata concern me with access-leaks repository. They use things like size and timings and metadata, and distinctions between failed request and successful request, side-effects from 500 and 404 error, for determining a particular resource was for a logged in user and not logged in, or what a particular user has access to.
- ... So I'm a little worried about blurring the line between reading a resource and including a resource
- ... Then CORP can't give you size, and CORP can't give you metadata. It can only give you access to pull in a resource. For size in particular we may already be over that line with measureMemory() work.
- ... Wondering if that framing makes any sense
- **Yoav:** Reason I thought it's OK was because of memory measurement, already exposing those sizes.
- ... JS profiling goes beyond embeddability of the resource and enables profiling of JS resources
- ... Doesn't give byte-by-byte, but gives characteristics
- **Mike:** My understanding was only related to the size of the JavaScript, does it include other things like images and fonts
- **Ulan:** Initially it was about JavaScript, but after cross-origin isolation we thought we could expand it to more memory
- **Camille:** None of that gives you metadata though
- **Artur:** I think the answer to what Mike mentioned. One is explicitly APIs like measureMemory() and by setting CORP you're allowing it to be loaded into address space of cross-origin renderer. In practice it allows you to put resource into render, so it can read the specific bytes, size of resource, and the metadata in that resource.

- ... So if you're a website that allows your resource to opt-in in a way that allows this, it's hard to make a argument that it's unsafe to expose that information
- **Mike:** The way I was thinking about this a year ago, is to make the attacker to do the attack. There is risk a malicious entity could gain information. CORS grants that access, an explicit mechanism to give access. There's a meaningful difference in intent between a server that wants its resources to be included versus a resource that's going to be read.
- ... We know that's not the case today, like Spectre attacks and other side-channels reveal this info, but we don't need to bake that into the APIs.
- **Camille:** But in that case, why do we say it's ok to be cross-origin isolated and not require that all subresources in those contexts go through CORS rather than CORP
- **Mike:** I think that's the conclusion you would reach if Artur is correct, but I disagree
- **Camille:** From an implementation perspective, I have no guarantee that cross-origin resources would not be attacked in a cross-origin context
- **Mike:** Agreed, there is value in making the attacker perform the attack
- **Camille:** In particular for size, if you can read measureMemory() API they can easily do that, and there's not much value in hiding it vs. having attackers perform it
- **Mike:** Surprised measureMemory() can read more than the JS heap
- **Ryosuke:** Raised this concern yesterday, and we will not be implementing the JS profiling API, so these precedents are not valid for us
- **Yoav:** In my view, it's unclear what we would tell developers they would need to do in what resources they can set CORP to beyond just public resources that expose nothing confidential or network information.
- ... Anything beyond that is something tricky and subtle and something they may get wrong.
- ... Mike do you have a proposal for an opt-in for ...
- **Mike:** It would be valuable for servers to have a signal that it is not a resource ever changed by the user requesting it. Not affected by cookies, IP, etc.
- ... Seems like we would be able to do some things if we had an assertion from the server that a resource would never change based on the user requesting it
- ... Not sure what all those changes would be
- ... But a mechanism for the server to make this assertion, or the site has content that changes
- ... I think that's a good idea apart from whether it gives any ability via timing and other proposals here
- ... I wonder whether there are things we can do with caching and various levels of the stack if we had an assertion that this particular resource is never personalized
- ... Servers could abuse this or there's some messaging potential
- **Yoav:** Just unclear to me what the delta would be between this and CORP
- **Mike:** CORP uses cookies
- **Camille:** Could we have this an extension of CORP - credential-less mode?
- **Mike:** My claim is if you accept there is a distinction between CORP and CORS, then there is a set of resources that can be included in your page, versus included+read. That

latter group can be split about things that are user specific, but can be read across origins, where you need an ACAO header with a specific origin.

- … There's another kind that's not user specific. We have that in CORS with ACAO:* but it's hard to use, needs to be used in anonymous mode
- ... If we reset expectations and set a flag that this is a public resource, that's a different category
- **Yoav:** In my slide here, if we had that magical mode today, you could put that instead of CORP here. It's just unclear to me what the difference is.
- **Nic:** For RUM providers, things like sizes is important to us from a pageweight PoV. We're reporting sizes on all of the content on our customers pages. It's not complete due to opt-in, so our assertions about page size are not accurate. If customers would be able to measure size for 3Ps, they'd be able to better assess their page size
- … From a CDN pov, we could improve the network … but a lot of it is around page weight
- … status codes and content types can help us catch errors
- **Yoav:** visibility on type of resources? credentialed/user specific/etc?
- **Nic:** did a crawl and many resources include CORS, but no breakdown of types of resources. 25% TAO + 25% more with CORS
- **Mike:** hard to know from the outside - application specific if the resource doesn something interesting with user data
- **Yoav:** From a developer perspective the difficulty about CORS is a) preflights and b) the fact that you can't opt-in all your resources to CORS and having to deal with the crossorigin attribute. I see the appeal in a "not personalized" kind of header, even if it overlaps with CORS.
- **Mike:** The place I would like to get is a place where the defaults are different from where they are today. i.e. credentialed request mode is not enabled by default.
- ... Changing that default would give us different characteristics on the web than what we see today
- ... And some kinds of requests are easier to reason about. People using IP addresses for ACLs and intranets would still be an issue. But it would help if resources using credentials became more rare.
- … That comes in two steps: introduce a new COEP: "don't send my credentials". We could then introduce a server side header indicating that the resource is not personalized. But that's not yet spelled out.
- **Artur:** I am still hung up on the distinction between the mode that would replace CORP (credential-less mode), this is exactly what CORP was meant to do. Only difference is semantic where CORP cross-origin I allow a resource to be read in isolated contexts.
- ... To me CORP cross-origin encompass already what you are talking about
- **Mike:** That could be the case. I'm willing to be convinced that this distinction is not one that makes any difference, and that it's not worth trying to introduce complexity around those concepts. Not what I had in mind when we were designing these.
- ... Seems to me in the status quo there are resources that want to change the content by user, but do not want that content to be exposed to the embedder

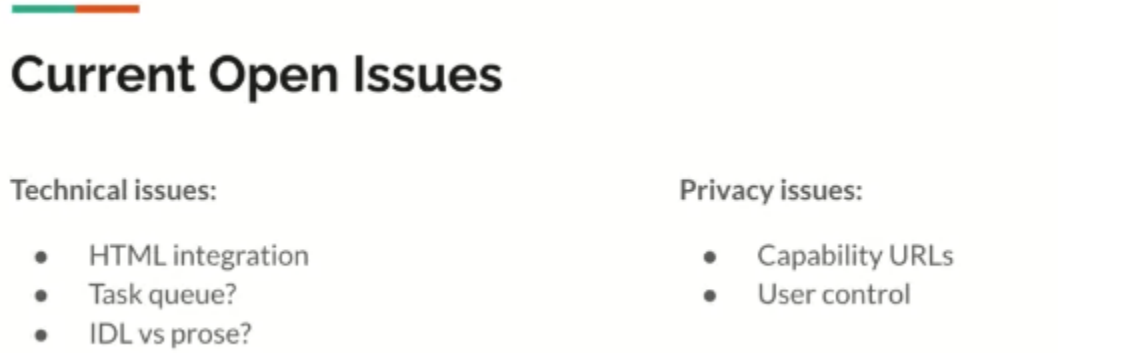
- ... That might not be possible and we'd have to align the web's API surface to that. But I'm not quite willing to give that up yet.
- **Yoav:** What concerns me about drawing that line is that people may trip over it and fall and break their face
- ... In my view it's better to have a line that splits personalized vs. non-personalized than a more subtle one
- **Mike:** Part of the line I think might be worth holding is whether the content may be legible. Artur may have more context as he's dealing with resource Google sets CORP on. If it's the case for all of those resources we would also be comfortable using CORS across origins, that's a strong argument towards the stance that we shouldn't make a distinction.
- **Artur:** I think it is, but it's conflated by how I've been thinking about CORP is where we roll it out
- ... If we were to set CORP cross-origin on a resource that has authenticated information, we would treat this as a vulnerability, because it could end up in someone else's process.
- **Mike:** One point that might make this more complicated is that in Chrome we have the ability to load IFRAMEs out of process, where that may not be in other browsers
- ... We require you to set CORP in isolated contexts. Not clear to me that every resource that wants to be embedded in a page also wants to be read by its embedder
- **Camille:** For example on Android iframes are not necessarily in a different process
- **Mike:** Browsers are not yet at a point where out-of-process iframes are universal. In that world there are HTML documents that would not want to be legible to their embedder yet want to be embedded. That's a line we want to hold, or otherwise, we need to change some iframes.
- **Camille:** That nore an issue implementation problem issue of cross-origin isolated and what it includes
- **Mike:** It's an implementation problem, but will be us for a while
- **Camille:** If you want to be embedded in a context where you won't be legible, you don't set COEP on yourself, so you'd still be able to be embedded across-origin.
- **Mike:** We're over time, what's the right place to continue this discussion
- **Yoav:** I'll create a new issue on ResourceTiming and do my best to summarize this issue
- **Ryosuke:** Want to echo the point that we have to hold the line that resources may want to be embedded but not read. We've been thinking about this. Hardest ones are scripts. For images other than size don't need to be readable via CORP.
- ... Some assumptions discussed here are not all agreed upon by browsers

Reporting API updates - Ian Clelland

- **Ian:** currently Chrome is the only implementer

- … support for

- … reporting API has been split into Reporting and Network Reporting
- … different scopes, data and privacy implications
- … Spun off crash, deprecation and interventions reports to their own documents
- … the Reporting API defines structure, report format, end points. Defines reports that are tied to the lifetime of the Document. Also defined ReportingObserver
- … Recently added worker support, credentials no longer sent across-origins, updates to relative URL handling
- … Network Reporting API - covers out-of-band reporting, like NEL
- … includes endpoint groups for improved reliability
- … Previously report-to couldn't been used as an identifier, so we moved it to an origin level opt-in - Origin Policy
- … Also, defined as infrastructure without any report types
- Open issues



- **Yoav:** Just as a reminder, the reasoning for splitting up the previous reporting API into 2 separate infrastructure specs was that Mozilla was interested in implementing one but not the other
- **Ian:** Roughly matches my understanding, they wanted reporting for COOP/COEP, but not interested in reporting beyond the document

- **Yoav:** On the Origin Policy front, any movement? Saw open issues regarding the performance implications of Origin Policy dependencies that from the network reporting perspective may not be awful.
- **Ian:** Probably not as bad. The 2 biggest issues I recall is the first load problem, and concerns around the Origin Policy itself being a cookie-like mechanism. What we need for network reporting is some sort of origin-level configuration, where Origin Policy seems like a good candidate for that, but it may not be the only one.
- … We could tie it to DNS/SRV records
- **Yoav:** That would introduce operational complexity
- **Ian:** Interested in what other implementers think
- **Sean:** Not too sure about Mozilla. Need to check
- **Alex:** No strong feelings, but haven't followed it closely enough
- **Ian:** For the privacy issues, should we take it to the privacy CG?
- **Yoav:** Normally, we tackle issues internally until we think the privacy story looks good. Can involve privacy oriented folks in those discussions. So can bring in folks from the Privacy CG to review bits and pieces
- **Carine:** Privacy is part of wide review, but maybe we can meet with them and get ideas
- **Yoav:** wide-review is rather late, but we can kick off an earlier one with the PING
- **Yoav:** the capability URLs point seem more boolean than the user control one. Seems like we don't want reports to maintain state beyond cookies, and if they do we should fix it. The user control question seems like something more opinionated, with various right answers depending on the UA and the user's opinions.
- **Ian:** I'd prefer to leave that to implementers and there's room for different browsers to have different opinions, but there are people who disagree.
- ...On the capability URL front, the browser cannot prevent capability URLs from leaking other than advising site owners against doing that
- … The risk is exposing the actual URLs in reports that get shared with 3P (e.g. login session identifiers, change password, etc)
- **Yoav:** One way to tackle it is to expose only a part of the actual URL and only expose hashes to enable the first party find the URL through URL to hash mapping.
- **Alex:** the game of guessing which parts of URLs are sensitive is a losing game. We should stick to "any portion of a url is sensitive and can be used to steal information".
- **Ian:** Probably true in the generic case

Reporting API and performance metrics - Yoav Weiss

- **Yoav:** Today we suggest developers and analytics to use page visibility data to send beacons before the user leaves
- ... Existing events such as pageshow, pagehide, beforeunload, unload are not reliable hooks to ensure data can be sent
- ... Visibility events are an unintuitive proxy for "send your beacons here"
- ... Requires backend stitching for everything beyond the first visibility session
- ... Still not common, skewed reporting for continuous metrics
- ... Many reports are biased towards "onload" time or other early stages in page lifecycle

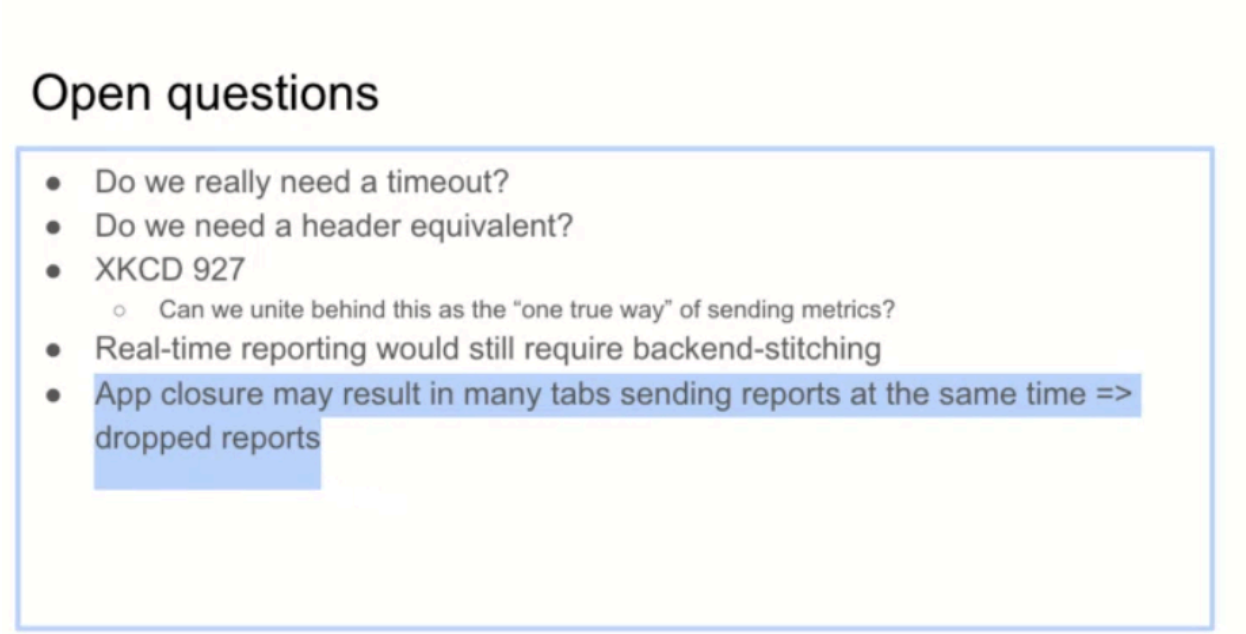
- ... Target use case: "I wish I could just tell browser the metrics I want to collect and send it when tab is gone"
- ... Using Reporting API
- ... Non-goal: I wish there was a way to collect known metrics without running scripts
- ... Rough sketch:

-
- ... Developer has a blog of various metrics, an endpoint and timeout (so reports don't lag too much)
- ... Register report
- ... Later-on, the user can add metrics to the blob
- ... Report send by browser when timeout expires or renderer dies
- ... Interesting?
- ... Slightly better ergonomics?
- **Steven:** I don't think we would switch because today we send a beacon, SPA, we send one payload per SPA page transition. Doing at the end of the session would be hours of data.
- **Yoav:** What do you do for cases where the user goes away?
- **Steven:** We implemented our own version of TTI, wait for activity to die down, send a beacon once that is reached. We don't wait for visibility change.
- **Michal:** Makes sense for a long-lived SPA, so a problem around cutting up sessions in a load. Even around one route or soft-nav, you may want to send out a report early. Isn't it true you'd want to have a summary at the end of that.
- ... With this proposal, you would send as late as possible with as much data as possible.
- Is the backend stitching work just something that hasn’t happened yet, or something more fundamental?
- **Nic:** I can talk of our use cases, but wonder - if this type of API would enable you to send the data either now, after a timeout or when the page unload, then you kinda get

the best of all worlds where you can accumulate it as long as you want, but if you need to force it out due to soft navigation you could.

- … We are constantly struggling between capturing as much data as we can and sending it out right away. Want to send right away due to both past reliability issues as well as for real-time reporting. That contradicts with collection not-just-loading data: JS errors, frame rates, and other continuous metrics.
- … Today we send the majority of the data around onload, but there is data we want to collect later. If we had a guarantee that we can send all that data in a single beacon, we wouldn't send it at onload, but later, when we have more data.
- … It kinda like an improvement to beacon, where we want to pile up more data and have the browser handle the transport. Would be very useful for us from a reliability point of view, and would enable us to be less load-focused.
- **Yoav:** Will you use it and if so, when?
- **Nic:** We would probably migrate to that straight away once available, as it would give some customers more data and improve its reliability and accuracy
- … Would definitely be a priority to use it right away
- **Ian:** The reporting APIs try to coalesce reports when possible and delay them to save user bandwidth. I heard you want to send immediately. We would in that case give the ability to snapshot the data.
- **Yoav:** in the model that I had in mind, you still have the object, so you can always cancel the report and sendBeacon it if you need to force data out.
- **Ian:** Would it be important to force it out, and would queueing be acceptable?
- **Nic:** We would like to guarantee real-timeness of the data. So waiting ~ minutes would hurt that.
- **Ian:** Currently wait a minute
- **Scott:** This has come up when talking about task scheduling, so want to schedule analytics in an idle task, but want guarantee that the task would eventually run. Currently track the tasks manually as well as scheduling them, but want an option to guarantee they run e.g. on page hide. So considering an option to postTasks that tells you when those tasks need to run. That gives you the best of both worlds: run if there's nothing else, but not interfere with user input and make sure it runs when user visibility changed. Do we need something generic for that?
- **Yoav:** Report sending doesn't have to happen on main thread.
- … Bits that could be interesting for scheduling pieces is metric collection. Would be good to coordinate those efforts
- **Gilles:** Isn't this a slightly-more-advanced sendBeacon?
- **Yoav:** Different in that you setup an empty blob, then fill in the data later
- ... Gives you data reliability that you're render won't die without sending data
- **Gilles:** Original intention of sendBeacon, have some control of timeouts for when it gets sent
- **Noam:** This sounds that it may be more reliable than sendBeacon. Notion of sending last-minute telemetry data before the session ends. sendBeacon can be reliable if there is a certain threshold of concurrent requests.
- ... If this could be a higher-guaranteed, it would be a nice option

- **Gilles:** Was original intent of sendBeacon() that it always works, but when it was implemented it wasn't work in always cases
- ... When it's actually implemented there are shortcomings
- **Yoav:** Fundamental difference is there's something you can declare ahead of time, and the browser can prepare. Where sendBeacon() is something you call at the very last instance.
- **Gilles:** But when if you're modifying this at the very last moment, may the data not make it?
- **Yoav:** Potentially, yeah
- Nicolás: From my perspective the main benefit is improving the reliability of sending data, for implementation, we would need to store this in the browser in case the renderer crashes. So it's different from any task the renderer wants to run.
- ... We don't want to store arbitrary data for any arbitrary renderer
- ... Also limiting by size can be important, because if what you're sending gets too big, it reduces the likeness of it succeeding.
- Open questions:



-
- **Yoav:** If there are multiple tabs all with queued reports, and the browser crashes, it will have to send all reports at once
- **Michal:** If some of the concerns around real-time and sending requests earlier, you may still have folks sending some data at visibility changes.
- **Yoav:** You don't even have to cancel it, you take metrics out of it that you send right away
- **Steven:** If this API had a way of send reports now, so you add your data to that object, then if you could send it now, flush buffer, that would be great
- ... In a long session, we could decide when to send it

- **Yoav:** And if a timeout is really what you want, a setTimeout that does that, and if that never happens, that's fine too
- **Yoav:** Also wanted to ask if the concern around real-time and stitching data that comes later would prevent this
- **Benjamin:** Can you walk through the workflow intended by server-side stitching
- **Yoav:** When collecting metrics, have a collection server that receives metrics, where they want to process individually and displayed. If you get multiple reports per session, one example is sending reports on vis state changes, and the user clicks back and forth and you want to report on that single session, then you need at the collecting server or processing pipeline to realize that report A and D are from the same session/user and they need to be stitched together for presentation purposes.
- **Michal:** I'm very interested in metrics after page load
- ... If you want to track the average interaction cost or smoothness, you need to wait to get all of that data
- ... This gets more and more complicated if you have things like BFCache
- ... You may have an eager report and a late report, or you may have a trickle effect.
- ... Summing up might be different per-metric, and there's a whole backend infra issue that doesn't align with how it's built today. Is that correct?
- **Nic:** yeah. Let's say you have a data point on PLT and CLS, there also metadata that goes along with each one of those. If we could group that all together in single DB row, that's way more efficient than having 10 rows with the same metadata but with different timestamps. So it becomes a data processing problem to have data points trickle. From an infra point of view, that's why we try to group all of our data and send it all at once.
- ... A lot of that is outside the concerns of this API, but a lot of RUM analytics providers do the same thing, and it'll help on that front. Makes it easier to process.
- **Yoav:** How do you envision you using this API while still accommodating the real-time constraints? Early report and complete report?
- **Nic:** Maybe classify types of data as real-time vs. continuous.
- **Michal:** I see that this could be appealing as a "lazy sendBeacon". But we talked about the performance timeline to evolve to e.g. soft navs. If this was tied specifically to just the performance timeline, we might be able to help hook those two together. E.g. doing the report automatically every time a session cuts.
- … Is the intention to focus narrowly or will it be up to developers to manually send the reports?
- **Yoav:** Haven't thought about that, but could be interesting to bake in.
- … We could define some notion of sessions where things get sent
- ... Developer-driven and browser-driven sessions
- ... Send this report any time the page goes hidden, or some navigation-y thing happens
- ... We could bake into frameworks, e.g. with the new navigation API
- **Michal:** I suspect you'll always want developers to slice in some manner
- ... I can see the ability to be more eager in slicing and also having the ability to capture the full page lifecycle
- **Yoav:** Major open question, can we unite behind one true way of sending metrics
- Gilles Are you thinking of a standardized report format

- ... Observe and report and decide which fields you want to get
- ... Trend of new APIs like NEL where it's all done automatically
- **Yoav:** I think you'll always need JavaScript to decide what to send
- ... A predefined report format is something else than what we were proposing here