Philip Tellis
@bluesmoon

Nic Jansma
@nicj

https://github.com/SOASTA/boomerang

http://www.soasta.com/mpulse

https://github.com/SOASTA/measuring-continuity

SOASTA

Measuring Continuity

2016-06-22

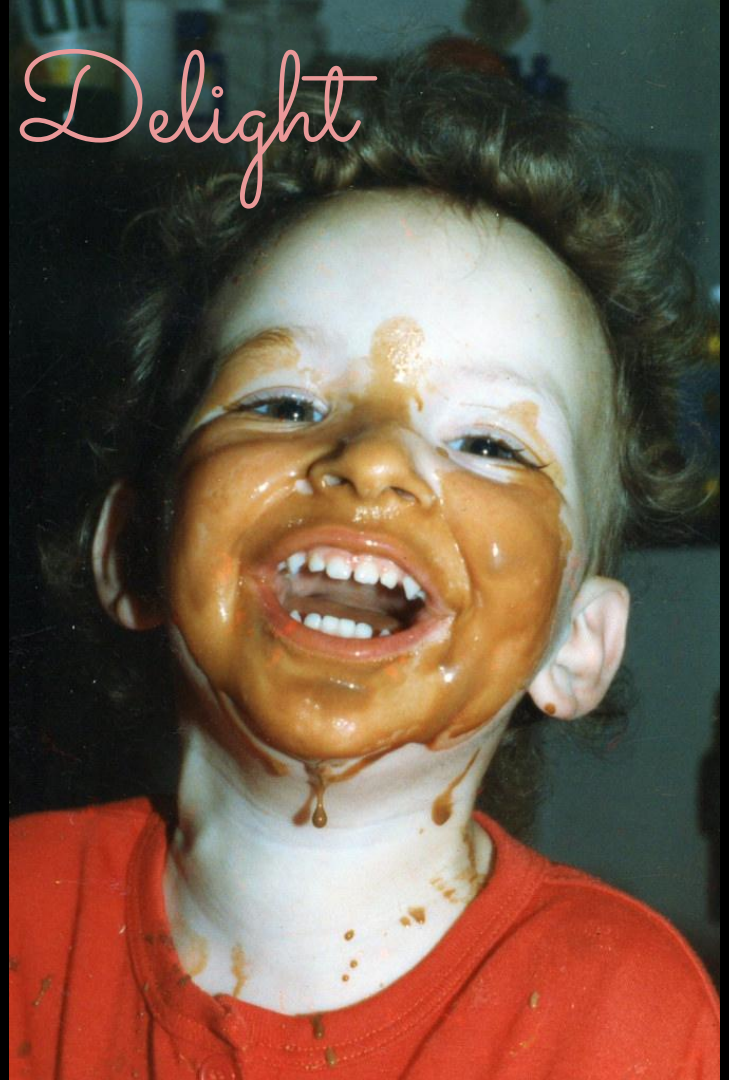
#VELOCITYCONF 2016

boomerang

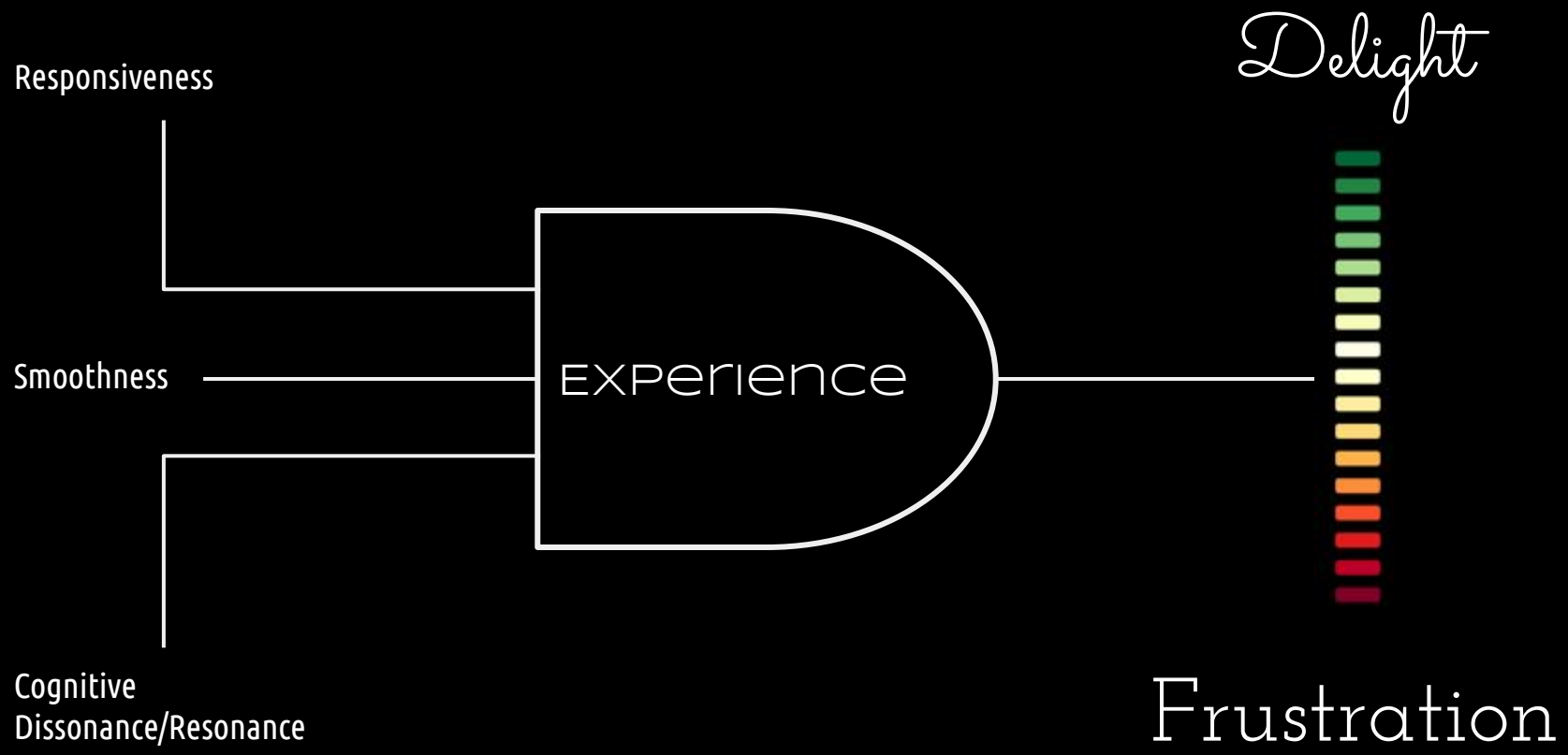
https://github.com/SOASTA/boomerang

WARNING: HUMANS WHO SUBMIT PULL REQUESTS ARE KNOWN TO HAVE BEEN HIRED

WHY



Or


Responsiveness
Smoothness
Cognitive
Dissonance/Resonance
experience
Delight
Frustration

RUM today

- We measure everything up to navigation complete (page load or SPA nav)
- We measure whether users bounce or convert

But

- The bulk of user interaction and experience happens after navigation has completed

Which continuous variables can we measure and how?

Developer Tools

Elements Console Sources Network Timeline Resources Security Audits Profiles HTTPS Everywhere EditThisCookie Adblock Plus

Capture: JS Profile Memory Paint Screenshots

500 ms 1000 ms 1500 ms 2000 ms 2500 ms 3000 ms 3500 ms 4000 ms 4500 ms

FPS CPU NET HEAP 28.8 MB – 71.8 MB

600 ms 800 ms 1000 ms 1200 ms 1400 ms 1600 ms 1800 ms 2000 ms 2200 ms 2400 ms 2600 ms 2800 ms 3000 ms 3200 ms 3400 ms 3600 ms 3800 ms 4000 ms 4200 ms 4400 ms 4600 ms 4800 ms

431.0 ms 185.8 ms 295.4 ms 342.6 ms 163.9 ms 165.7 ms

Interactions

Main

Par...]) P...) Evalua...js:1) E...

C... (an...on) (...

re...js

re...re

init

JS Heap [28.8 MB – 71.8 MB] Documents [13 – 36] Nodes [2 342 – 5 064] Listeners [95 – 391] GPU Memory

Summary Bottom-Up Call Tree Event Log

Range: 385 ms – 4.81 s

168.2 ms	Loading
2136.1 ms	Scripting
450.1 ms	Rendering
41.0 ms	Painting
554.2 ms	Other
1075.0 ms	Idle

Total: 4.42 s

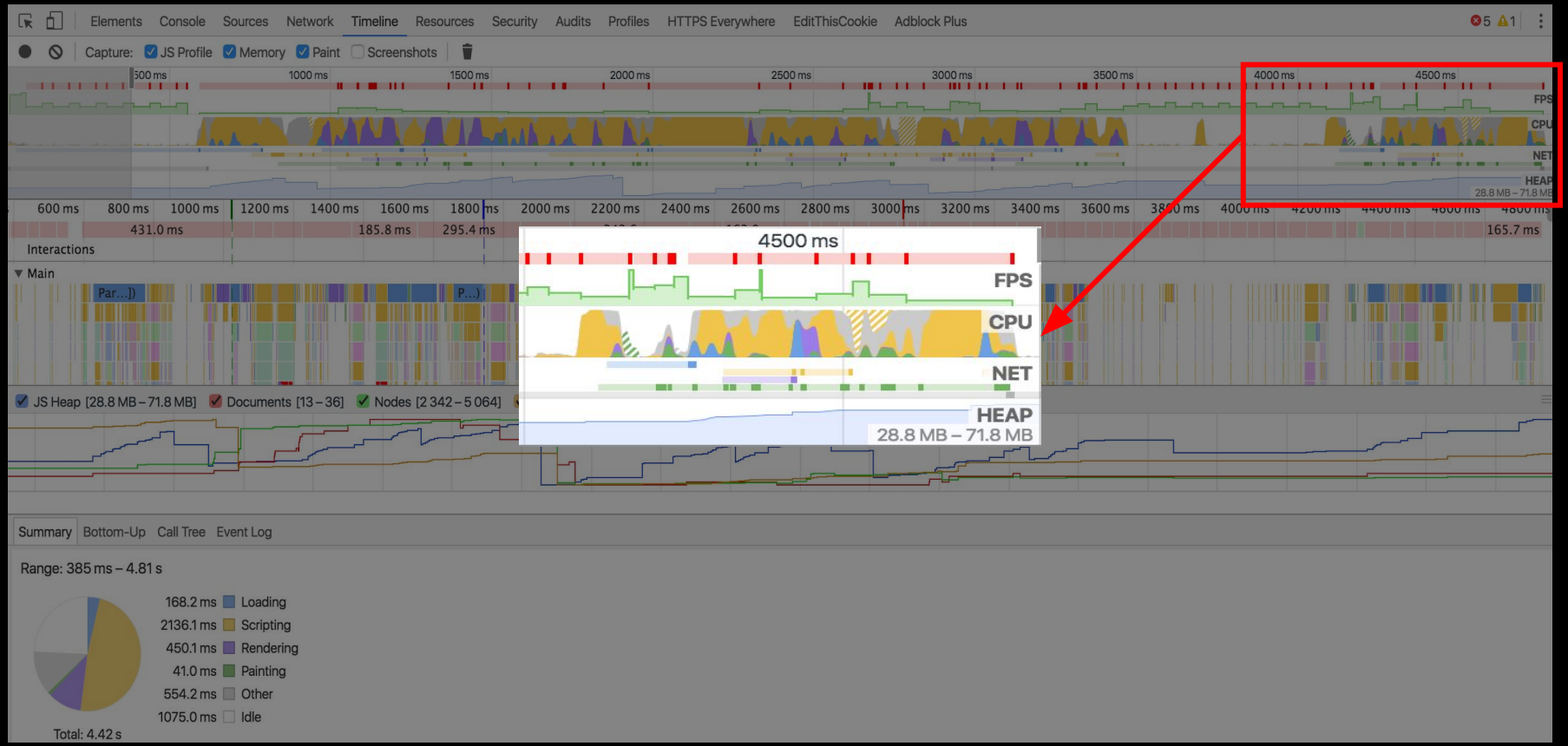
Developer Tools

"The fact that something is possible to measure, and may even be highly desirable and useful to expose to developers, does not mean that it can be exposed as runtime JavaScript API in the browser, due to various privacy and security constraints"

– Performance APIs, Security and Privacy

https://w3c.github.io/perf-security-privacy/

Continuity Metrics



FPS - Frames Per Second

- `requestAnimationFrame(callback)`
- Callback is run before the next paint

```
1 // total frames seen this second
2 var frames = 0;
3
4 function measureFps() {
5   frames++;
6
7   // request a callback before the next frame
8   window.requestAnimationFrame(measureFps);
9 }
10
11 // start measuring
12 window.requestAnimationFrame(measureFps);
13
14 // report on frame rate (FPS) once a second
15 setInterval(function() {
16   console.log("FPS: " + frames);
17   frames = 0;
18 }, 1000);
```

FPS - Long Frames

Frames > 16.6 ms lead to < 60 FPS

```
1  var lastFrame = performance.now();
2  var longFrames = 0;
3
4  function measureFps() {
5    var now = performance.now();
6
7    // calculate how long this frame took
8    if (now - lastFrame >= 18) { longFrames++; }
9
10   lastFrame = now;
11
12   window.requestAnimationFrame(measureFps);
13 }
14 window.requestAnimationFrame(measureFps);
15
16 // report on long frames once a second
17 setInterval(function() {
18   console.log("Long frames: " + longFrames);
19   longFrames = 0;
20 }, 1000);
```

FPS - Video

HTML5 VIDEO metrics (Chrome/FF)

```
var latestFrame = 0;
var latestReportedFrame = 0;

setInterval(function() {
  // find the first VIDEO element on the page
  var vids = document.getElementsByTagName("video");
  if (vids && vids.length) {
    var vid = vids[0];
    if (vid.webkitDecodedFrameCount || vid.mozPaintedFrames) {
      latestFrame = vid.webkitDecodedFrameCount || vid.mozPaintedFrames;
    }
  }

  console.log("Video FPS: "
    + Math.max(latestFrame - latestReportedFrame, 0));

  // reset count
  latestReportedFrame = latestFrame;
}, 1000);
```

CPU - Page Busy

- Browser doesn’t expose CPU metrics directly
- Detect Busy by running a function at a regular interval
- See if the callback runs at the time we expect
- If the callback was delayed, the page was Busy
- Busy can be caused by other JavaScript, layout, render, etc

CPU - Page Busy

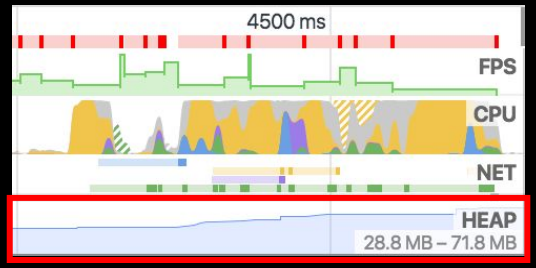

```
11 setInterval(function() {
12   var now = performance.now();
13   var delta = now - last;
14   last = now;
15
16   // if we're more than 2x the poll
17   // + deviation, we missed one period completely
18   while (delta > ((POLLING_INTERVAL * 2)
19     + ALLOWED_DEVIATION_MS)) {
20     total++;
21     late++;
22     delta -= POLLING_INTERVAL; // adjust, try again
23   }
24
25   total++;
26
27   if (delta > (POLLING_INTERVAL + ALLOWED_DEVIATION_MS)) {
28     late++;
29   }
30 }, POLLING_INTERVAL);
```

NET - Resources

- ResourceTiming
- Bytes available in ResourceTiming2


```
var resources =
  window.performance.getEntriesByType("resource");

// number of resources fetched
var resourceCount = resources.length;

// number of bytes
var bytesOverWire = 0;
resources.forEach(function(res) {
  bytesOverWire +=
    res.transferSize ? res.transferSize : 0;
});

console.log("Resources: " + resourceCount
  + " " + bytesOverWire + "b");
```

HEAP - Memory Usage



- Non-standard
- Reduced precision to avoid privacy concerns

```
// report on JS object memory once a second
setInterval(function() {
  var mem = window.performance
    && window.performance.memory
    && window.performance.memory.usedJSHeapSize;

  console.log("Memory usage: " + mem);
}, 1000);
```

Battery

- Monitor your visitor's battery state
- Reduce work on low battery

```
1 setInterval(function() {
2   navigator.getBattery().then(function(batt) {
3     console.log(batt.level);
4   });
5 }, 1000);
```

Interactions

Elements Console Sources Network Timeline Resources Security Audits Profiles HTTPS Everywhere EditThisCookie Adblock Plus

Capture: JS Profile Memory Paint Screenshots

500 ms 1000 ms 1500 ms 2000 ms 2500 ms 3000 ms 3500 ms 4000 ms 4500 ms 5000 ms 5500 ms 6000 ms 6500 ms 7000 ms

FPS CPU NET

Interactions: 389.4 ms Scroll; 1377.1 ms; Scroll; Scroll; 550.8 ms; 384.6 ms Scroll; 517.7 ms; 588.4 ms Response; 274.4 ms

Main

Event (DOM...ntLoaded) Functio...js:14) d ready fireWith l (anony...ction) trigger each lh.a.each each (anony...ction) trigger m.handle

(a...n) re...s re...e init e...e each (a...) (a...) (a...) e...e e...e each (a...) (a...) e...e

Funct...s:6) s.onload load (anon...ion) u.run i.run i.start u

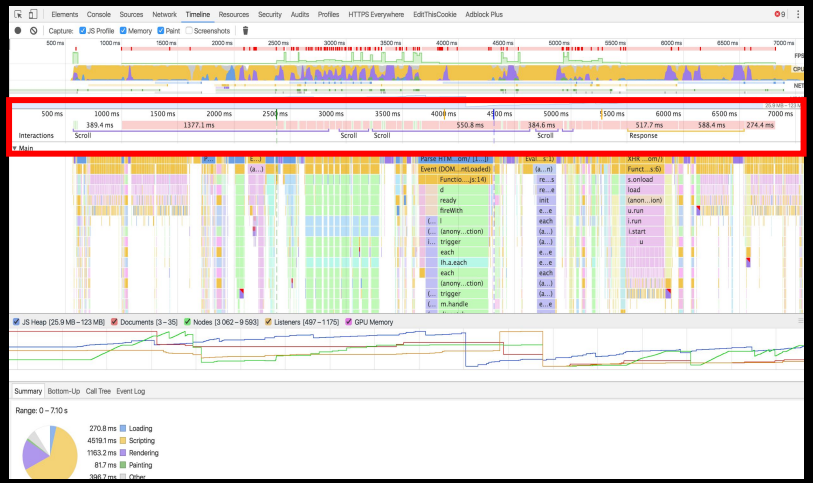
JS Heap [25.9 MB – 123 MB] Documents [3 – 35] Nodes [3 062 – 9 593] Listeners [497 – 1 175] GPU Memory

Summary Bottom-Up Call Tree Event Log

Range: 0 – 7.10 s

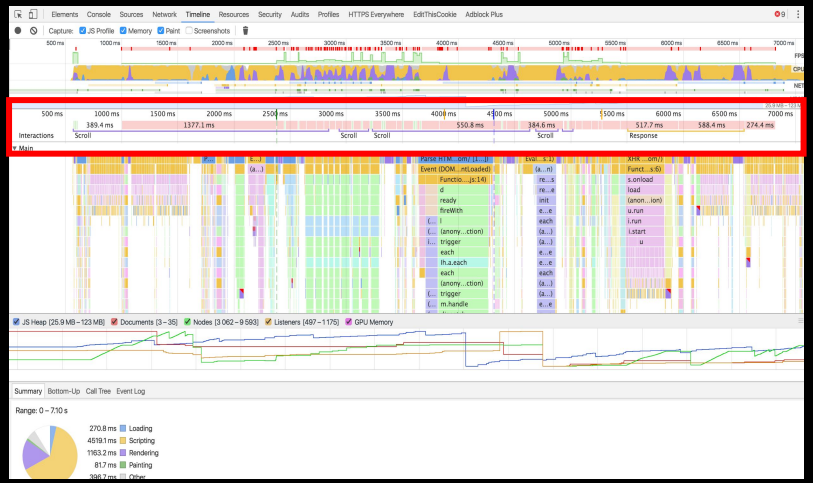
270.8 ms	Loading
4519.1 ms	Scripting
1163.2 ms	Rendering
81.7 ms	Painting
396.7 ms	Other

Interactions - User Input



- scroll
- mousemove
- click
- keydown

Interactions - Visibility



Window's visibility state

```
1 document.addEventListener("visibilitychange", function() {
2   console.log(document.hidden ? "hidden" : "visible");
3 }, false);
```

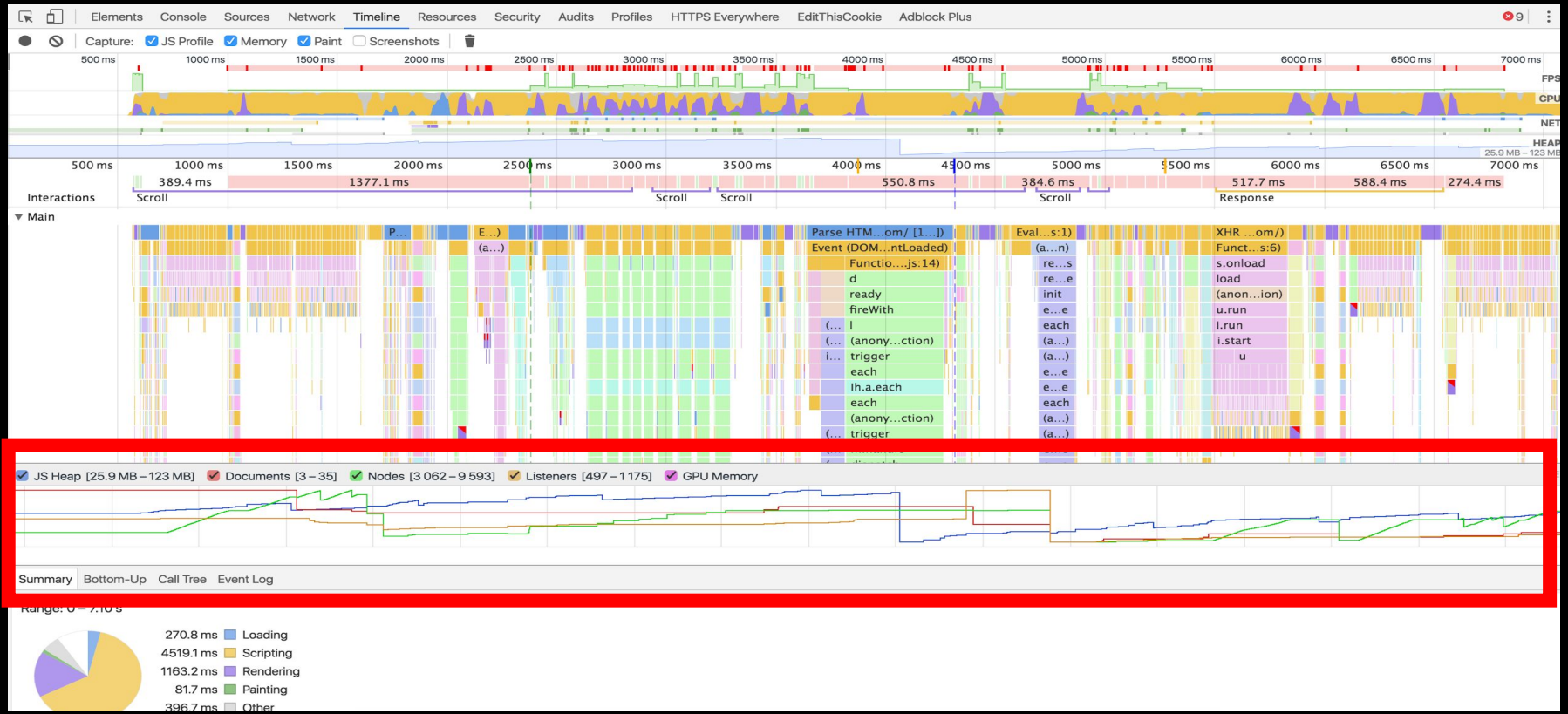
Also look at the `IntersectionObserver`

Interactions - Orientation

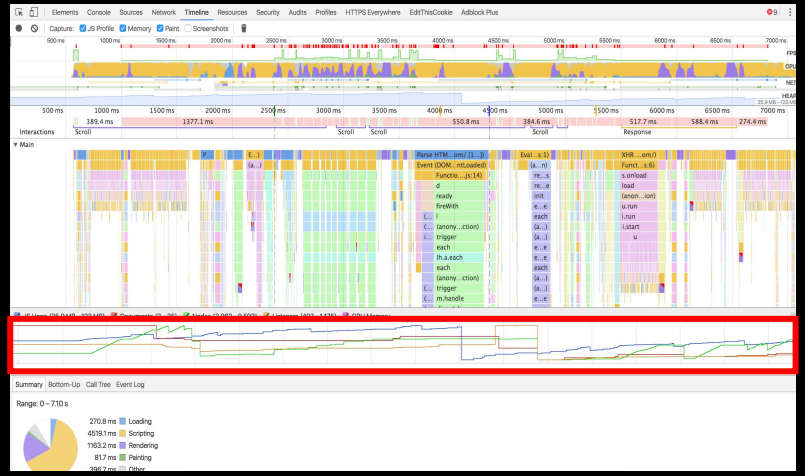
How the device is being held

```
window.addEventListener("orientationchange", function() {
  console.log("orientation: " + screen.orientation.angle);
});
```

Size Metrics



Size - Nodes



- HTML size (bytes)
- Overall Node count
- IFRAME, IMG, SCRIPT, etc., node count

Size - DOM Changes

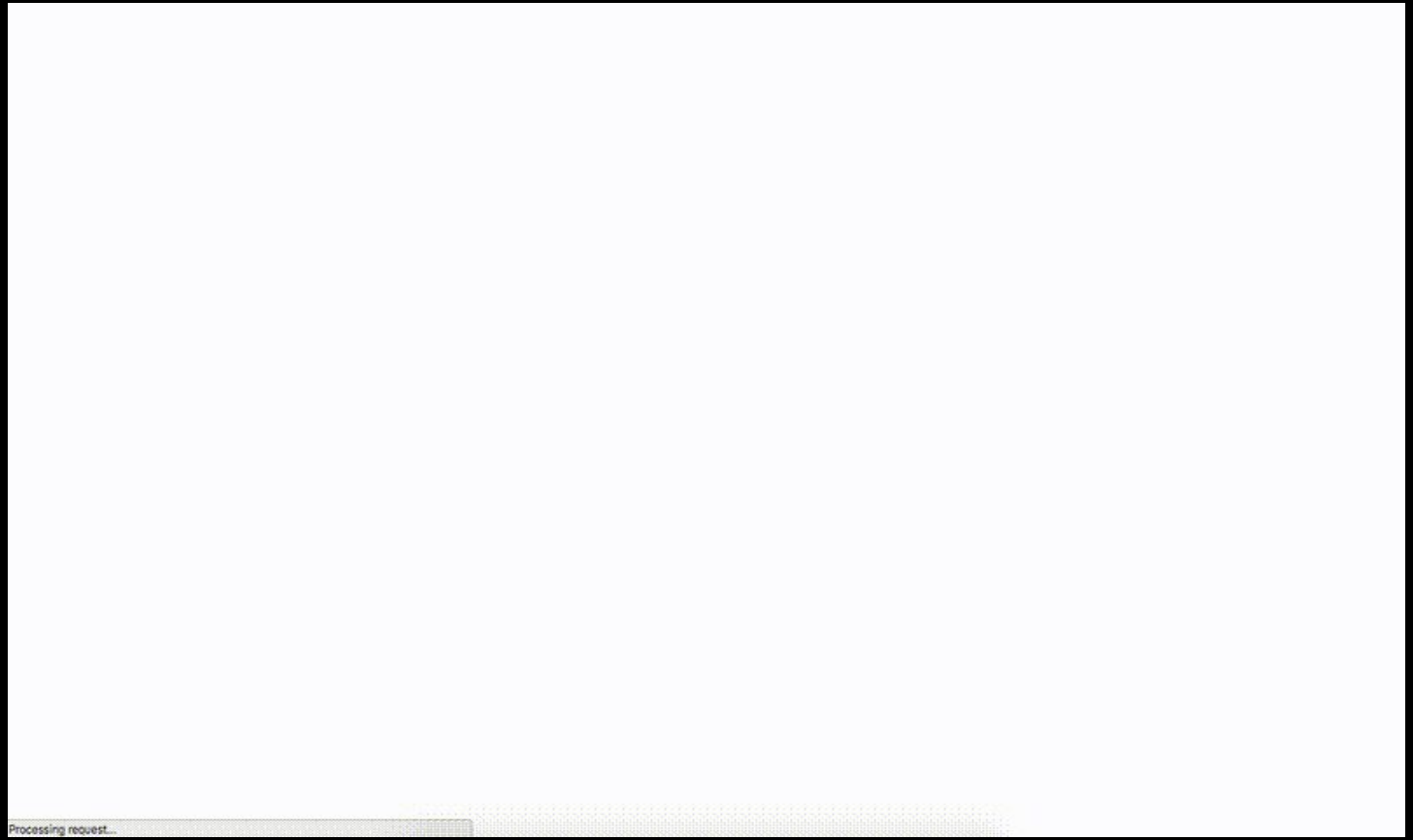
MutationObserver == change over time

```
var d = document;
var mutationCount = 0;
var domLength =
  d.getElementsByTagName("*").length;

// create an observer instance
var observer = new MutationObserver(function(mutations) {
  mutations.forEach(function(mutation) {
    if (mutation.type !== "childList") { return; }
    for (var i = 0; i < mutation.addedNodes.length; i++) {
      var node = mutation.addedNodes[i];
      mutationCount++;
      mutationCount += node.getElementsByTagName ?
        node.getElementsByTagName("*").length : 0;
    }
  });
});

// configure the observer
observer.observe(d, { childList: true, subtree: true });
```

Errors

```
1 var errorCount = 0;
2
3 window.onerror = function () {
4   errorCount++;
5 }
6
7 setInterval(function() {
8   console.log("Errors: " + errorCount);
9   errorCount = 0;
10 }, 1000);
```

Demo

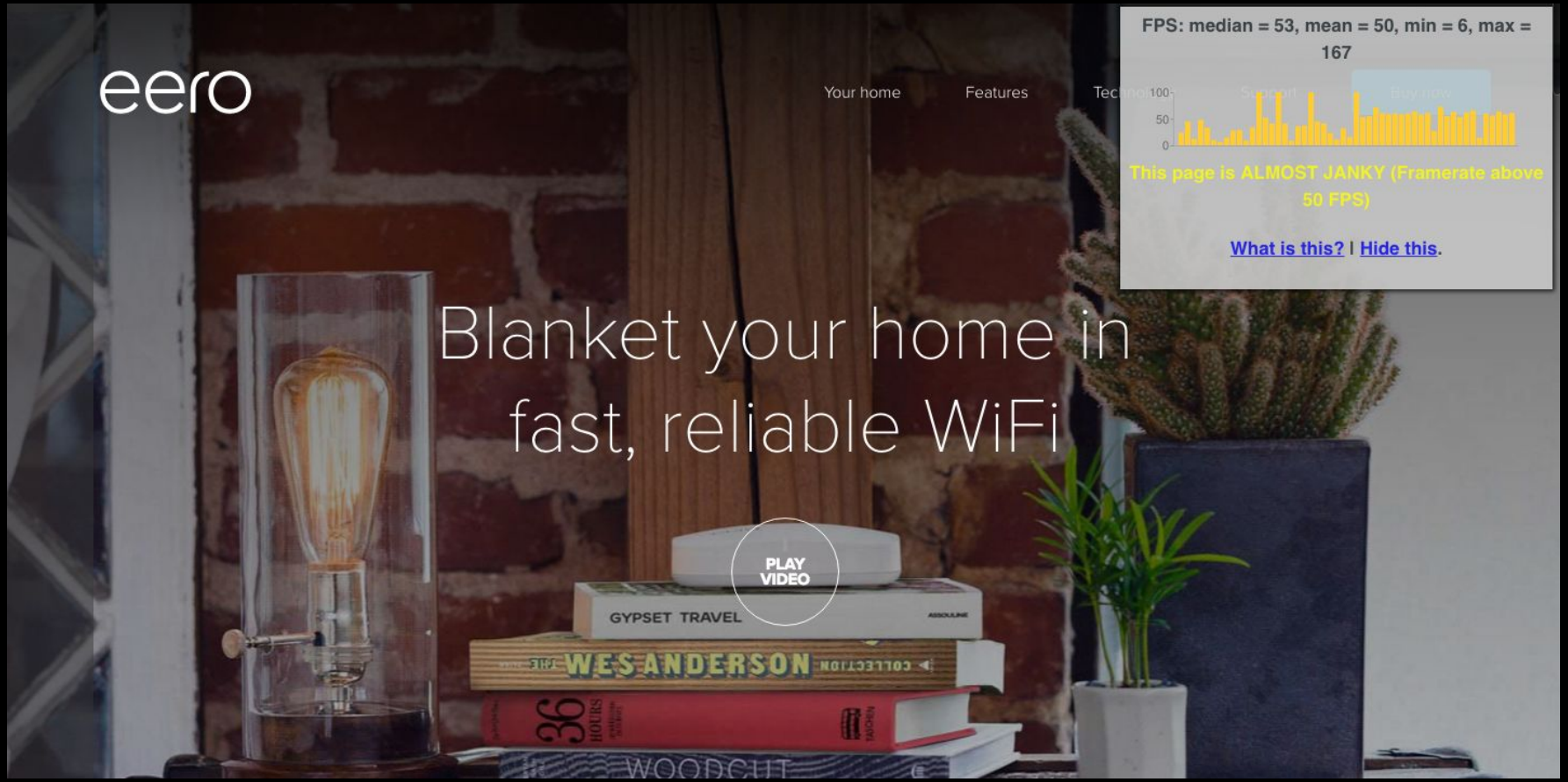


https://github.com/SOASTA/measuring-continuity

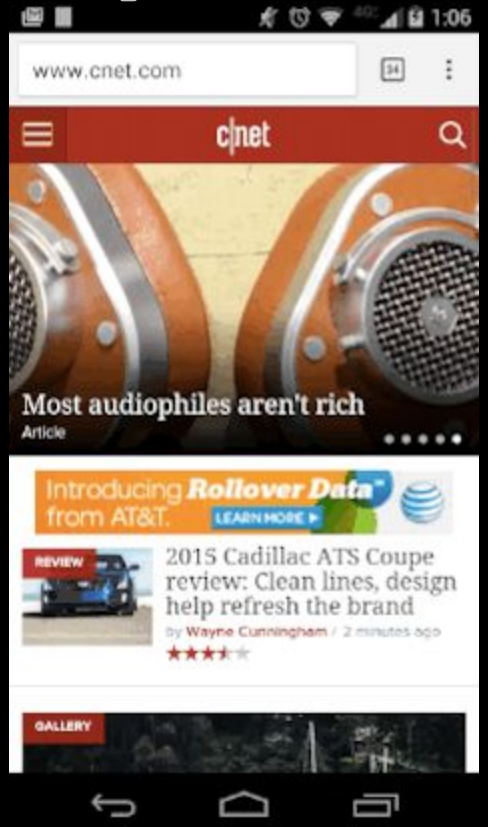
So what?

- Raw data != useful metrics
- Let’s measure the user experience
  - Smoothness
  - Responsiveness
  - Reliability
  - Emotion

Smoothness - FPS during scroll



Smoothness - FPS after interaction



Responsiveness

- How long it takes for the site to respond to input?
  - `requestAnimationFrame` to detect next paint
  - MutationObserver to detect DOM changes
- UserTiming to monitor your own code
- SPA instrumentation via `boomerang`

Responsiveness

```
1 document.addEventListener("click", function(e) {
2   var start = performance.now();
3   requestAnimationFrame(function() {
4      var delta = performance.now() - start;
5      console.log("Click responsiveness: " + delta);
6   });
7 }, false);
```

Reliability

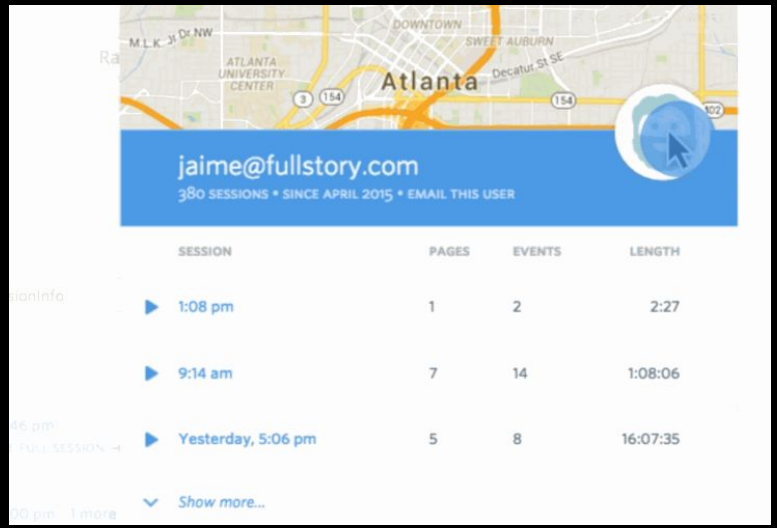
- JavaScript errors
- Leaks:
  - JavaScript memory usage over time
  - DOM size increase over time

Tracking Emotion

Rage Clicks

Rage clicks are series of clicks in which your users are pummeling their mouse buttons in frustration. It's like punching your site in the face, usually because it's not doing what the user wants or expects it to.

– Caitlin Brett, FullStory



https://blog.fullstory.com/moar-magic-announcing-rage-error-and-dead-clicks-1f19e50a1421

Rage Clicks

```
1 var same = 0, x = 0, y = 0, targ = null;
2
3 document.addEventListener("click", function(e) {
4   var nX = e.clientX; var nY = e.clientY;
5
6   // calculate number of pixels moved
7   var pixels = Math.round(
8     Math.sqrt(Math.pow(y - nY, 2) +
9     Math.pow(x - nX, 2)));
10
11  if (targ == e.target || pixels <= 10) {
12    same++;
13  } else {
14    same = 0;
15  }
16
17  console.log("Same area clicked: " + same);
18
19  x = nX; y = nY; targ = e.target;
20 }, false);
```

Dead Clicks

- Clicking without any meaningful visual (DOM) change
- Might happen during (or right after) page load due to delayed JavaScript

Dead Clicks

Missed Clicks

user clicks near an element, but misses it

Mouse Movement

"People who are angry are more likely to use the mouse in a jerky and sudden, but surprisingly slow fashion.

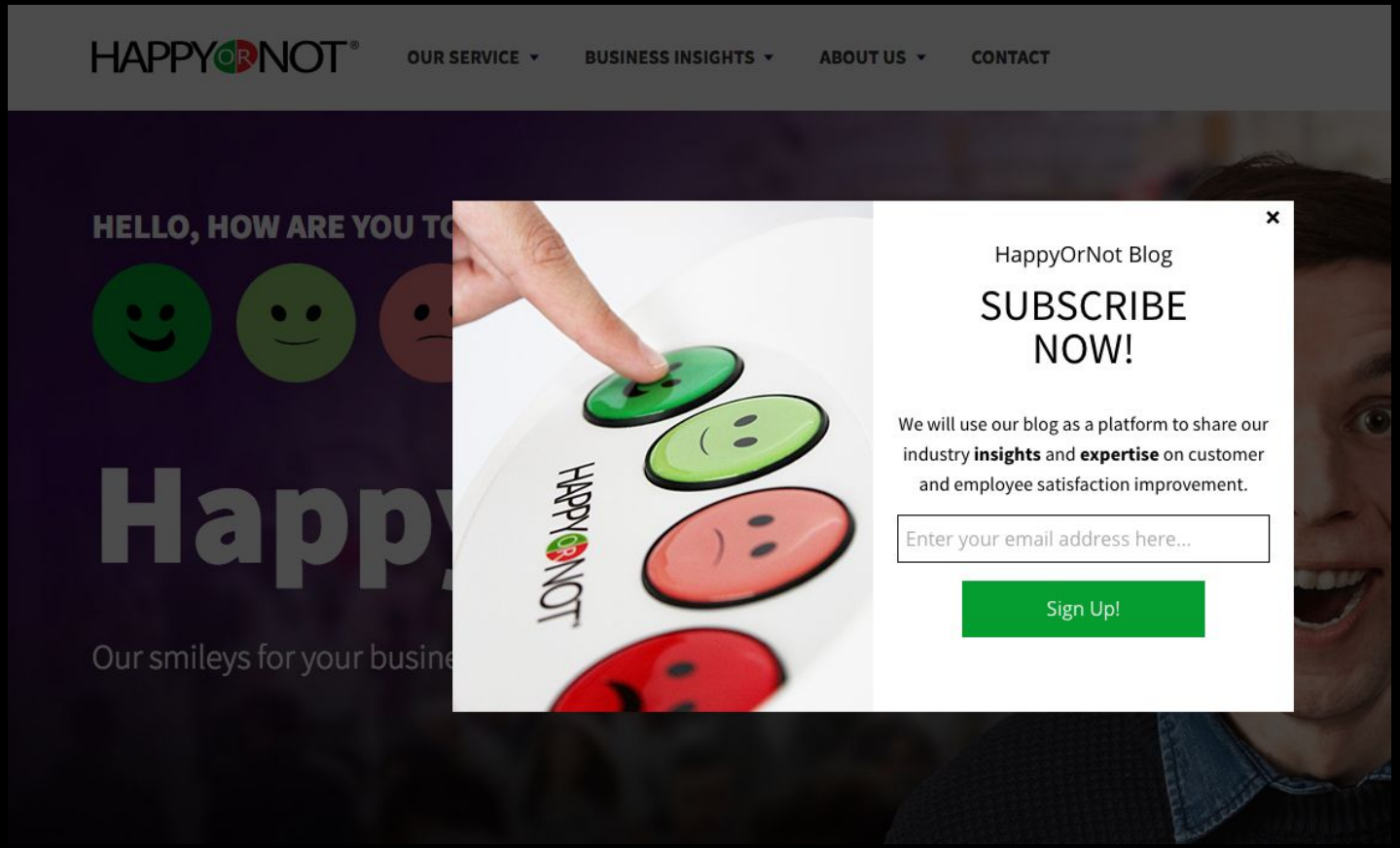
People who feel frustrated, confused or sad are less precise in their mouse movements and move it at different speeds."

– Inferring Negative Emotion from Mouse Cursor Movements

Martin Hibbeln, Jeffrey L. Jenkins, Christoph Schneider, Joseph S. Valacich, and Markus Weinmann

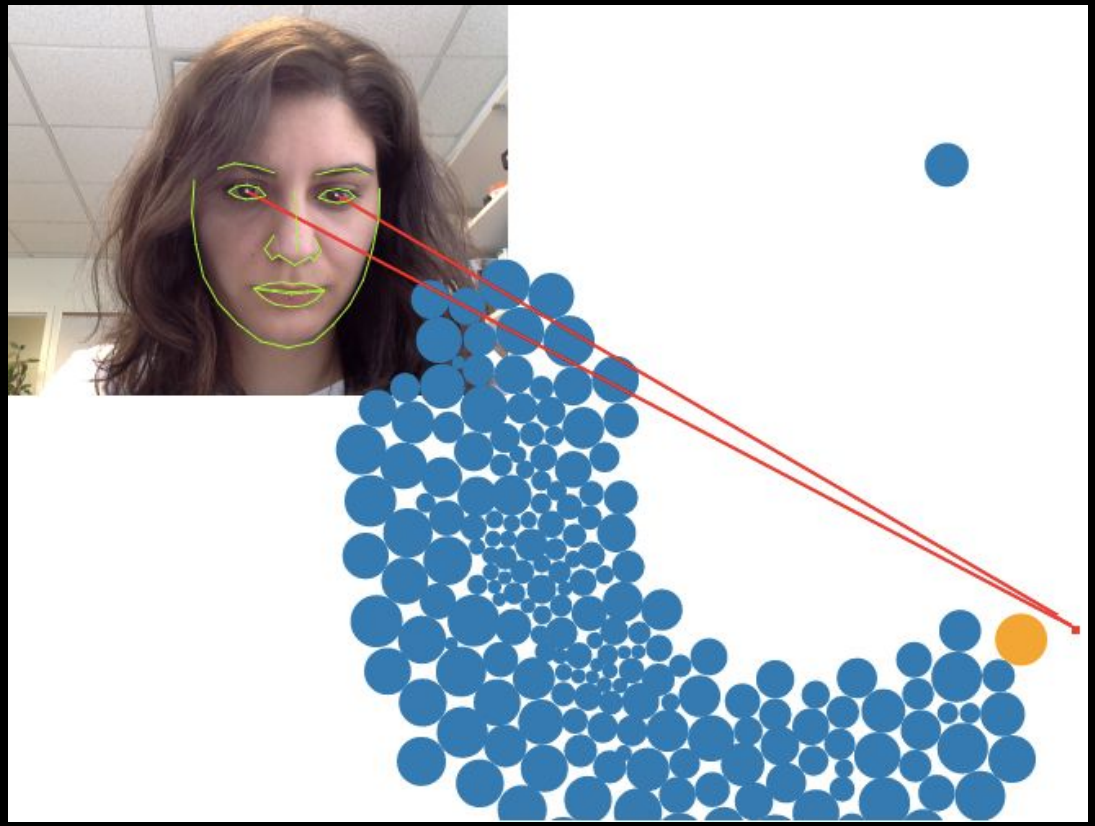
Ask Directly

Rage Clicking



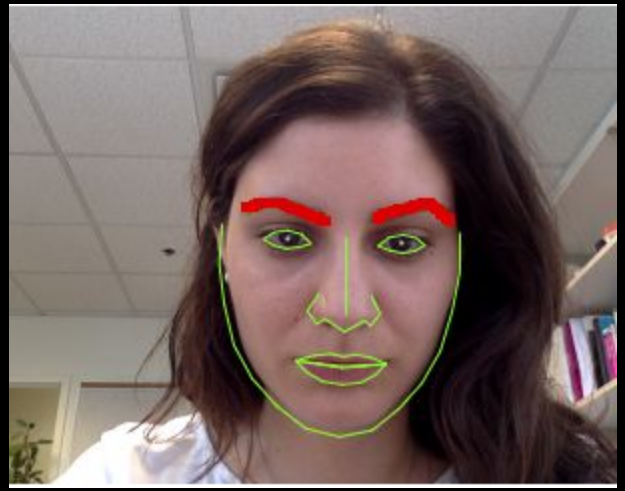
Eye Tracking

https://webgazer.cs.brown.edu/



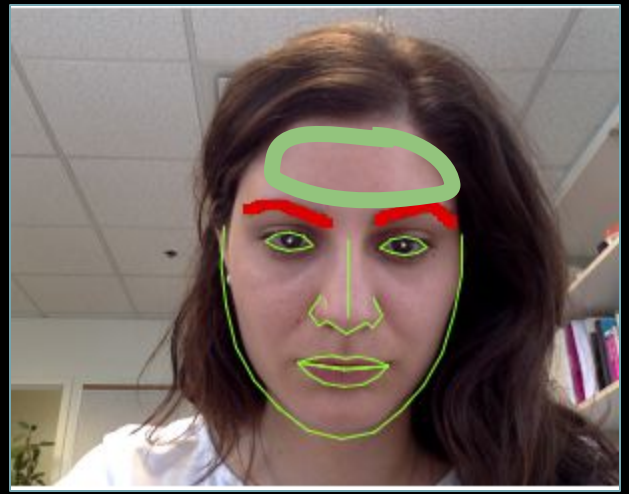
Eyebrow Tracking

https://webgazer.cs.brown.edu/



Emotion - α / β wave Tracking

Mind Reading Markup Language



Further Reading

Rage Clicks
http://blog.fullstory.com/2015/12/reducing-ux-rage-with-fullstorys-rage-clicks/

Inferring Emotion from Mouse Movements
http://www.telegraph.co.uk/technology/news/12050481/Websites-could-read-emotions-by-seeing-how-fast-you-move-your-mouse.html

Scroll Behaviour
http://blog.chartbeat.com/2013/08/12/scroll-behavior-across-the-web/

WebGazer: Eye tracking in JavaScript
http://webgazer.cs.brown.edu/

What JavaScript knows about you
http://webkay.robinlinus.com/

Prerender Events
https://wiki.whatwg.org/wiki/Link_prerender_events

RequestAnimationFrame
https://developer.mozilla.org/en-US/docs/Web/API/window/requestAnimationFrame

RequestIdleCallback
https://developers.google.com/web/updates/2015/08/using-requestidlecallback

IntersectionObserver
https://wicg.github.io/IntersectionObserver/

Video Metrics
https://wiki.whatwg.org/wiki/Video_Metrics

Resource Timing
https://www.w3.org/TR/resource-timing/

The Runtime Performance Checklist
http://calendar.perfplanet.com/2013/the-runtime-performance-checklist/

Jank Meter
https://webperf.ninja/2015/jank-meter/

RAIL Performance Audit of SFGate.com
https://docs.google.com/document/d/1K-mKOqiUiSjqZTEscBLjtjd6E67oiK8H2ztOiq5tigk

Performance: Security & Privacy Considerations
https://w3c.github.io/perf-security-privacy/

Motion Mark Analysis
https://docs.google.com/document/d/1vKNGim07lvPCYL1ctiNss1BqhjfE49t6LwZkwoTkeXU/mobilebasic

Debouncing and Throttling Events
https://css-tricks.com/debouncing-throttling-explained-examples/

Code Examples from this talk
https://github.com/SOASTA/measuring-continuity

Thank You

Photo Credits

Rum on Ice *by* wiserbailey
https://www.flickr.com/photos/25084516@N03/4317148060/

Angel Delight *by* Auntie P
https://www.flickr.com/photos/auntiep/360764980/

Frustrated *by* Kevin Lawver
https://www.flickr.com/photos/kplawver/1903240219/

Philip Tellis
@bluesmoon

Nic Jansma
@nicj

https://github.com/SOASTA/boomerang

http://www.soasta.com/mpulse

https://github.com/SOASTA/measuring-continuity

SOASTA