MAKE IT FAST

Using Modern Brower APIs to Monitor and Improve the Performance of your Web Applications

nic jansma | nicj.net | @nicj

WHO AM I?

Nic Jansma

nic@nicj.net

@nicj

http://nicj.net

SOASTA

- SOASTA (current)
- Microsoft (2005-2011)
- Founding member of W3C WebPerf Working Group

STATE OF PERFORMANCE MEASUREMENT

How do we measure performance?

SERVER

- HTTP logs (apache, nginx, haproxy)
- Server monitoring (top, iostat, vmstat, cacti, mrtg, nagios, new relic)
- Profiling (timestamps, xdebug, xhprof)
- Load testing (ab, jmeter, soasta, blazemeter, loadrunner)

DEVELOPER

- Browser developer tools (ie, chrome, ff, opera, safari)
- Network monitoring (fiddler, wireshark, tcpdump)

BUT...

- Measuring performance from the server and developer perspective is not the full story
- The only thing that really matters is what your end-user sees
- Measuring real-world performance of your end-users is tough

USER

(circa 2010)

- `var elapsedTime = Date.now() - startTime;`
- Boomerang: github.com/lognormal/boomerang

W3C WEBPERF WORKING GROUP

www.w3.org/2010/webperf

Founded 2010 to give developers the ability to assess and understand performance characteristics of their web apps

> *The mission of the Web Performance Working Group is to provide methods to measure aspects of application performance of user agent features and APIs*

Microsoft, Google, Mozilla, Opera, Facebook, Netflix, etc

WORKING GROUP GOALS

- Expose information that was not previously available
- Give developers the tools they need to make their applications more efficient
- Little to no overhead
- Easy to understand APIs

PUBLISHED SPECS

- **Navigation Timing** (NT): Page load timings
- **Resource Timing** (RT): Resource load timings
- **User Timing** (UT): Custom site events and measurements
- **Performance Timeline**: Access NT/RT/UT and future timings from one API
- **High Resolution Time**: Better `Date.now()`

PUBLISHED SPECS (PT 2)

- **Page Visibility**: Visibility state of document
- **Timing control for script-based animations**: `requestAnimationFrame()`
- **Efficient Script Yielding**: More efficient than `setTimeout(...,0)`: `setImmediate()`

UPCOMING SPECS

- Beacon: Async send data (even after page is closed)
- Resource Hints: `rel="preconnect" rel="preload"`
- Resource Priorities: `lazyload`
- Frame Timing: Animation timings
- Navigation Error Logging: For failed navigations

PARTICIPATE!

www.w3.org/2010/webperf

public-web-perf@w3.org

github.com/w3c/web-performance

NAVIGATIONTIMING

www.w3.org/TR/navigation-timing

Goal: Expose accurate performance metrics describing your visitor's page load experience

Current status: Recommendation

Upcoming: NavigationTiming2

HOW IT WAS DONE BEFORE

(this isn't accurate)

```
<html><head><script>
var start = new Date().getTime();
function onLoad {
    var pageLoadTime = (new Date().getTime()) - start;
}
body.addEventListener(“load”, onLoad, false);
</script>...</html>
```

WHAT'S WRONG WITH THIS?

- It only measures the time from when the HTML gets parsed to when the last sub-resource is downloaded
- It misses the initial DNS lookup, TCP connection and HTTP request wait time
- `Date().getTime()` is not reliable

INTERLUDE

`DOMHighResTimeStamp`

	`Date`	**`DOMHighResTimeStamp`**
Accessed Via	`Date().getTime()`	`performance.now()`
Resolution	millisecond	sub-millisecond
Start	Unix epoch	`navigationStart`
Monotonically Non-decreasing	No	Yes
Affected by user's clock	Yes	No
Example	`1420147524606`	`3392.275999998674`

NAVIGATIONTIMING

window.performance.navigation

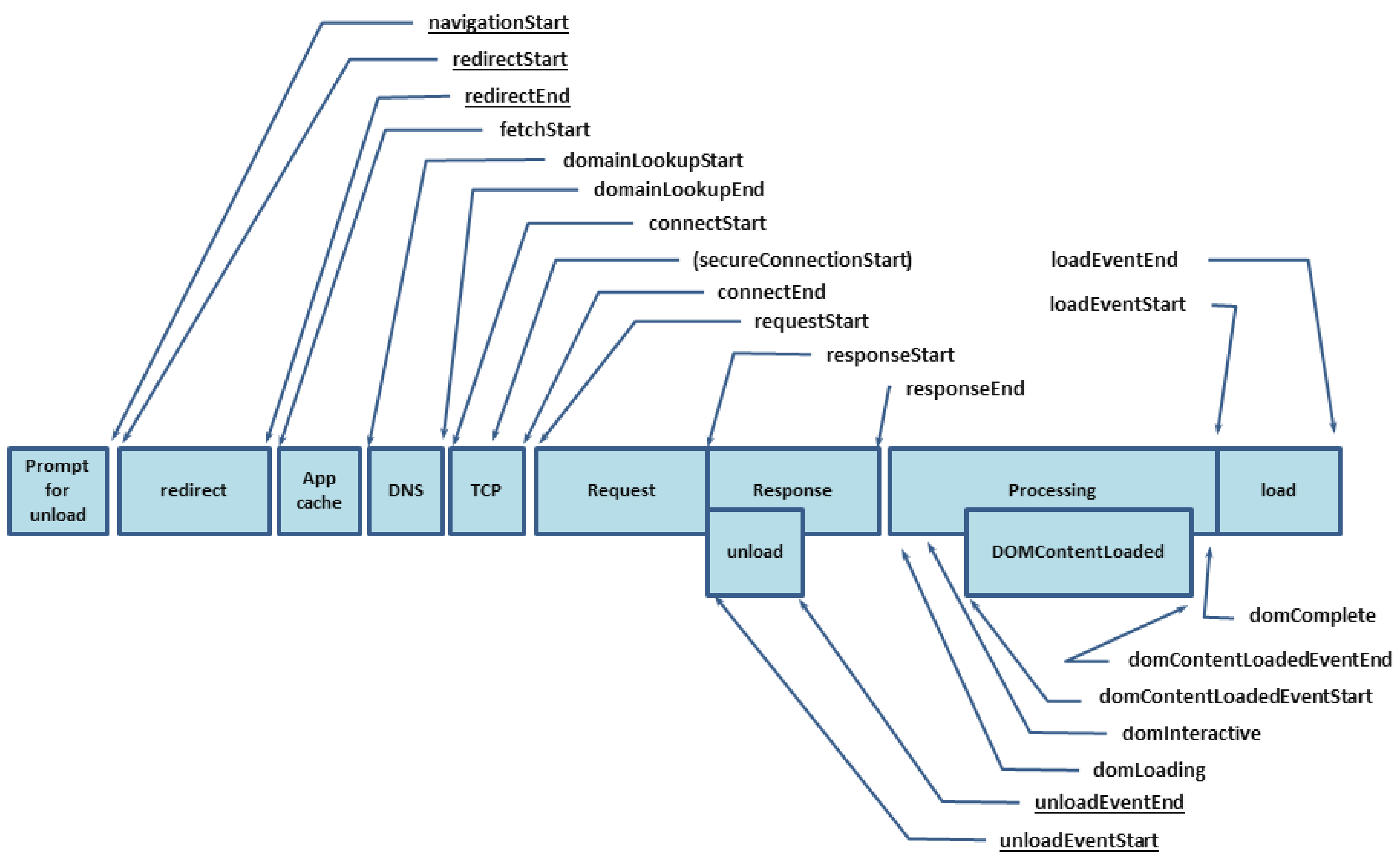
```
interface PerformanceNavigation {
    const unsigned short TYPE_NAVIGATE = 0;
    const unsigned short TYPE_RELOAD = 1;
    const unsigned short TYPE_BACK_FORWARD = 2;
    const unsigned short TYPE_RESERVED = 255;
    readonly attribute unsigned short type;
    readonly attribute unsigned short redirectCount;
};
```

NAVIGATIONTIMING

`window.performance.timing`

```
interface PerformanceTiming {
    readonly attribute unsigned long long navigationStart;
    readonly attribute unsigned long long unloadEventStart;
    readonly attribute unsigned long long unloadEventEnd;
    readonly attribute unsigned long long redirectStart;
    readonly attribute unsigned long long redirectEnd;
    readonly attribute unsigned long long fetchStart;
    readonly attribute unsigned long long domainLookupStart;
    readonly attribute unsigned long long domainLookupEnd;
    readonly attribute unsigned long long connectStart;
    readonly attribute unsigned long long connectEnd;
    readonly attribute unsigned long long secureConnectionStart;
    readonly attribute unsigned long long requestStart;
    readonly attribute unsigned long long responseStart;
    readonly attribute unsigned long long responseEnd;
    readonly attribute unsigned long long domLoading;
    readonly attribute unsigned long long domInteractive;
    readonly attribute unsigned long long domContentLoadedEventStart;
    readonly attribute unsigned long long domContentLoadedEventEnd;
    readonly attribute unsigned long long domComplete;
    readonly attribute unsigned long long loadEventStart;
    readonly attribute unsigned long long loadEventEnd;
};
```

NAVIGATIONTIMING



HOW TO USE

```
function onLoad() {
    if ('performance' in window && 'timing' in window.performance) {
        setTimeout(function() {
            var t = window.performance.timing;
            var ntData = {
                redirect: t.redirectEnd - t.redirectStart,
                dns: t.domainLookupEnd - t.domainLookupStart,
                connect: t.connectEnd - t.connectStart,
                ssl: t.secureConnectionStart ? (t.connectEnd - secureConnectio
                request: t.responseStart - t.requestStart,
                response: t.responseEnd - t.responseStart,
                dom: t.loadEventStart - t.responseEnd,
                total: t.loadEventEnd - t.navigationStart
            };
        }, 0);
    }
}
```

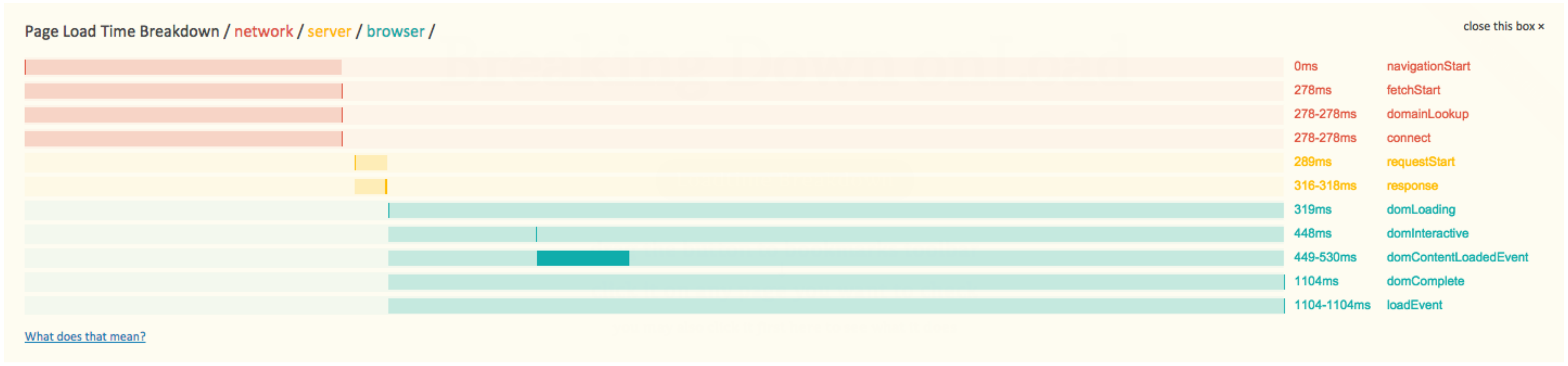
THEN WHAT?

DIY / Open Source

- Send this data to your backend for logging
- Show any page's timings via a bookmarklet: kaaes.github.io/timing
- Boomerang: github.com/lognormal/boomerang
- Boomcatch: cruft.io/posts/introducing-boomcatch
- BoomerangExpress: github.com/andreas-marschke/boomerang-express
- SiteSpeed.io: www.sitespeed.io
- Piwik: github.com/piwik/piwik

KAAES TIMING

kaaes.github.io/timing



Page Load Time Breakdown / network / server / browser /

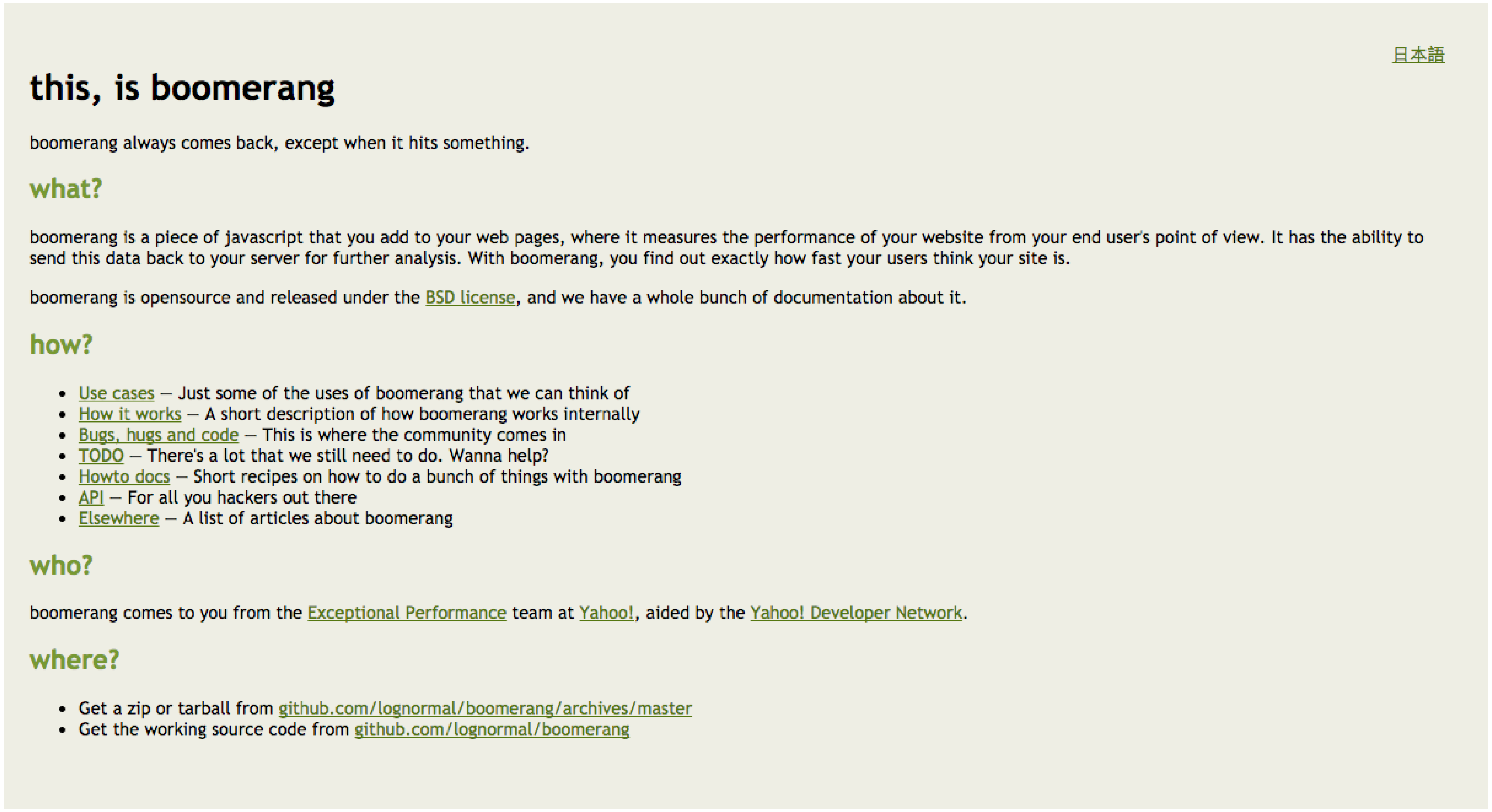
close this box ×

0ms	navigationStart
278ms	fetchStart
278-278ms	domainLookup
278-278ms	connect
289ms	requestStart
316-318ms	response
319ms	domLoading
448ms	domInteractive
449-530ms	domContentLoadedEvent
1104ms	domComplete
1104-1104ms	loadEvent

What does that mean?

BOOMERANG

github.com/lognormal/boomerang



日本語

this, is boomerang

boomerang always comes back, except when it hits something.

what?

boomerang is a piece of javascript that you add to your web pages, where it measures the performance of your website from your end user's point of view. It has the ability to send this data back to your server for further analysis. With boomerang, you find out exactly how fast your users think your site is.

boomerang is opensource and released under the BSD license, and we have a whole bunch of documentation about it.

how?

- Use cases — Just some of the uses of boomerang that we can think of
- How it works — A short description of how boomerang works internally
- Bugs, hugs and code — This is where the community comes in
- TODO — There's a lot that we still need to do. Wanna help?
- Howto docs — Short recipes on how to do a bunch of things with boomerang
- API — For all you hackers out there
- Elsewhere — A list of articles about boomerang

who?

boomerang comes to you from the Exceptional Performance team at Yahoo!, aided by the Yahoo! Developer Network.

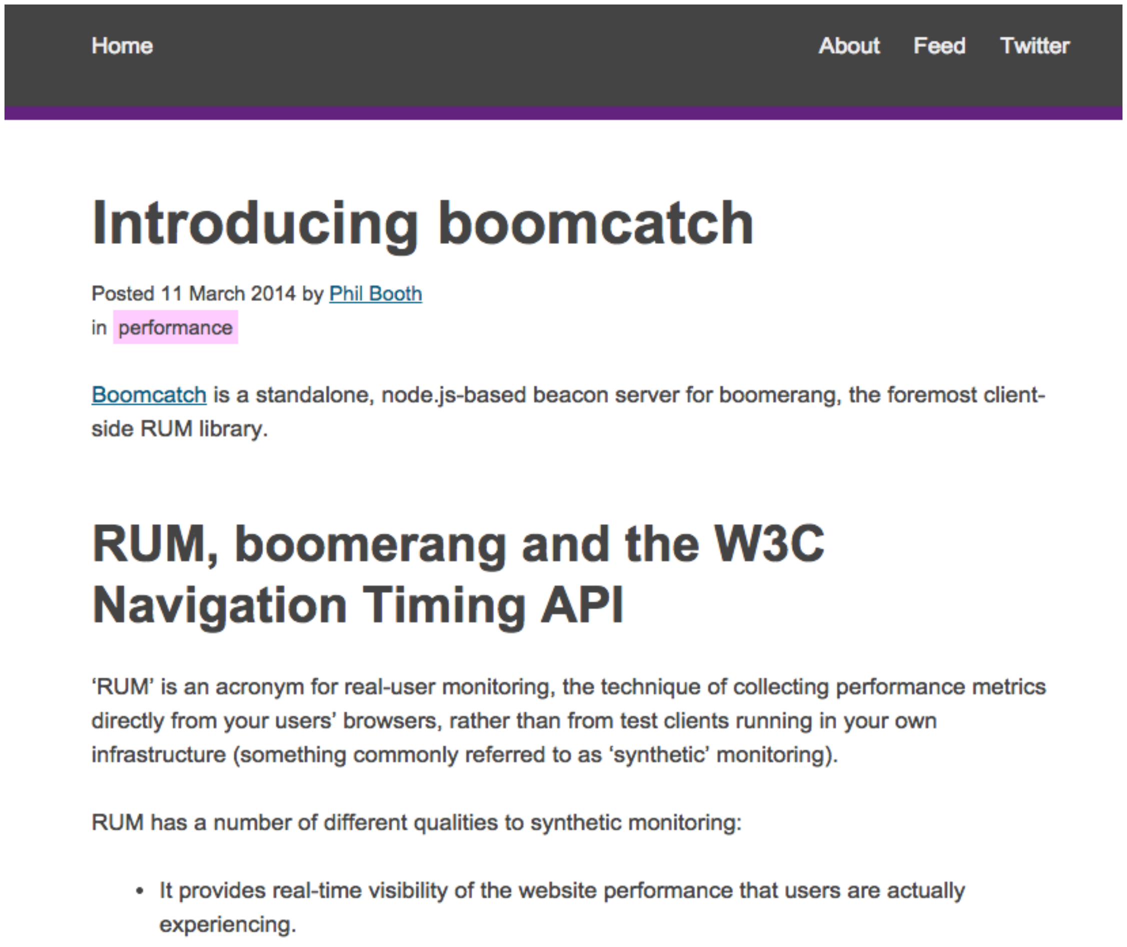
where?

- Get a zip or tarball from github.com/lognormal/boomerang/archives/master
- Get the working source code from github.com/lognormal/boomerang

BOOMCATCH

Collects beacons + maps (statsd) + forwards (extensible)

cruft.io/posts/introducing-boomcatch



Introducing boomcatch

Posted 11 March 2014 by Phil Booth
in performance

Boomcatch is a standalone, node.js-based beacon server for boomerang, the foremost client-side RUM library.

RUM, boomerang and the W3C Navigation Timing API

'RUM' is an acronym for real-user monitoring, the technique of collecting performance metrics directly from your users' browsers, rather than from test clients running in your own infrastructure (something commonly referred to as 'synthetic' monitoring).

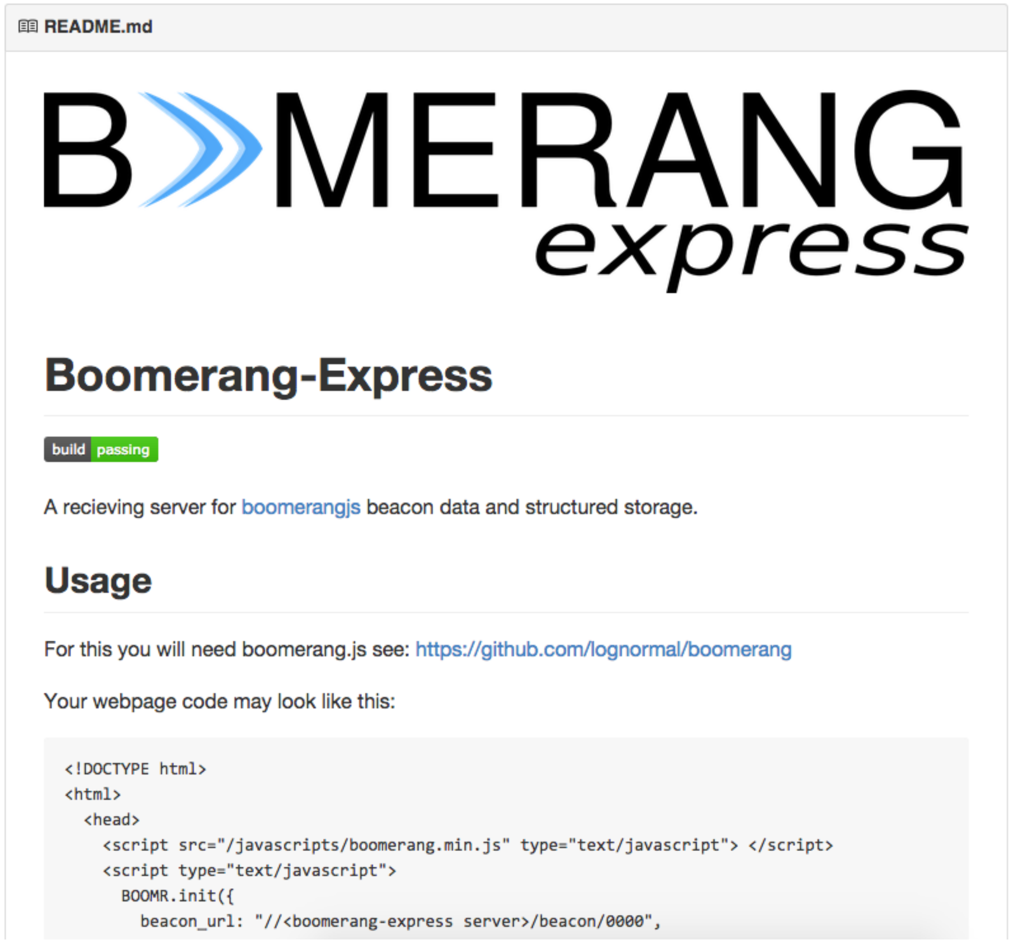
RUM has a number of different qualities to synthetic monitoring:

- It provides real-time visibility of the website performance that users are actually experiencing.

BOOMERANGEXPRESS

Collects beacons

github.com/andreas-marschke/boomerang-express



README.md

Boomerang-Express

build passing

A recieving server for boomerangjs beacon data and structured storage.

Usage

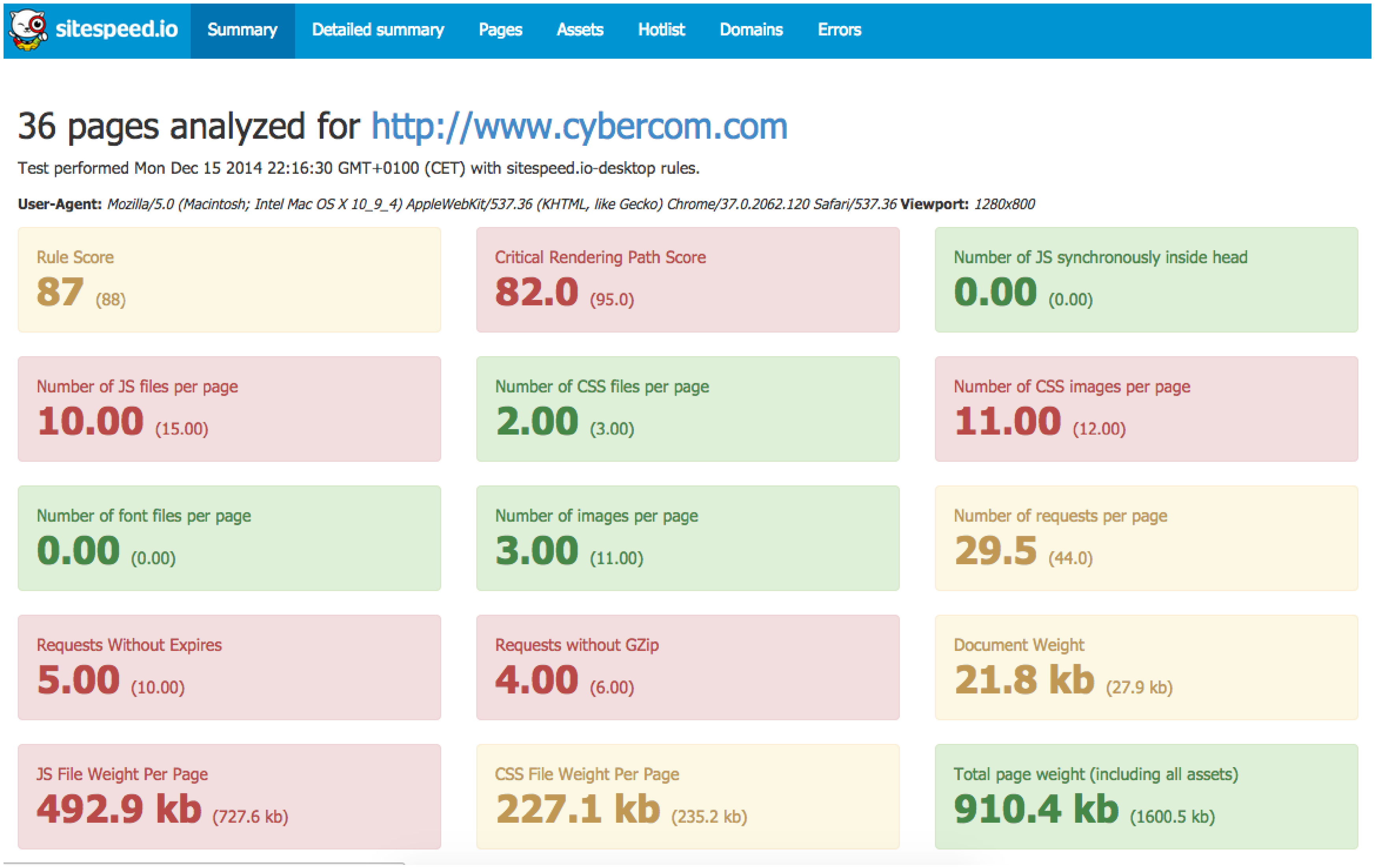
For this you will need boomerang.js see: https://github.com/lognormal/boomerang

Your webpage code may look like this:

```
<!DOCTYPE html>
<html>
  <head>
    <script src="/javascripts/boomerang.min.js" type="text/javascript"> </script>
    <script type="text/javascript">
      BOOMR.init({
        beacon_url: "//<boomerang-express server>/beacon/0000",
```

SITESPEED.IO

www.sitespeed.io



36 pages analyzed for http://www.cybercom.com

Test performed Mon Dec 15 2014 22:16:30 GMT+0100 (CET) with sitespeed.io-desktop rules.

User-Agent: *Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/37.0.2062.120 Safari/537.36* **Viewport:** *1280x800*

Rule Score
87 (88)

Critical Rendering Path Score
82.0 (95.0)

Number of JS synchronously inside head
0.00 (0.00)

Number of JS files per page
10.00 (15.00)

Number of CSS files per page
2.00 (3.00)

Number of CSS images per page
11.00 (12.00)

Number of font files per page
0.00 (0.00)

Number of images per page
3.00 (11.00)

Number of requests per page
29.5 (44.0)

Requests Without Expires
5.00 (10.00)

Requests without GZip
4.00 (6.00)

Document Weight
21.8 kb (27.9 kb)

JS File Weight Per Page
492.9 kb (727.6 kb)

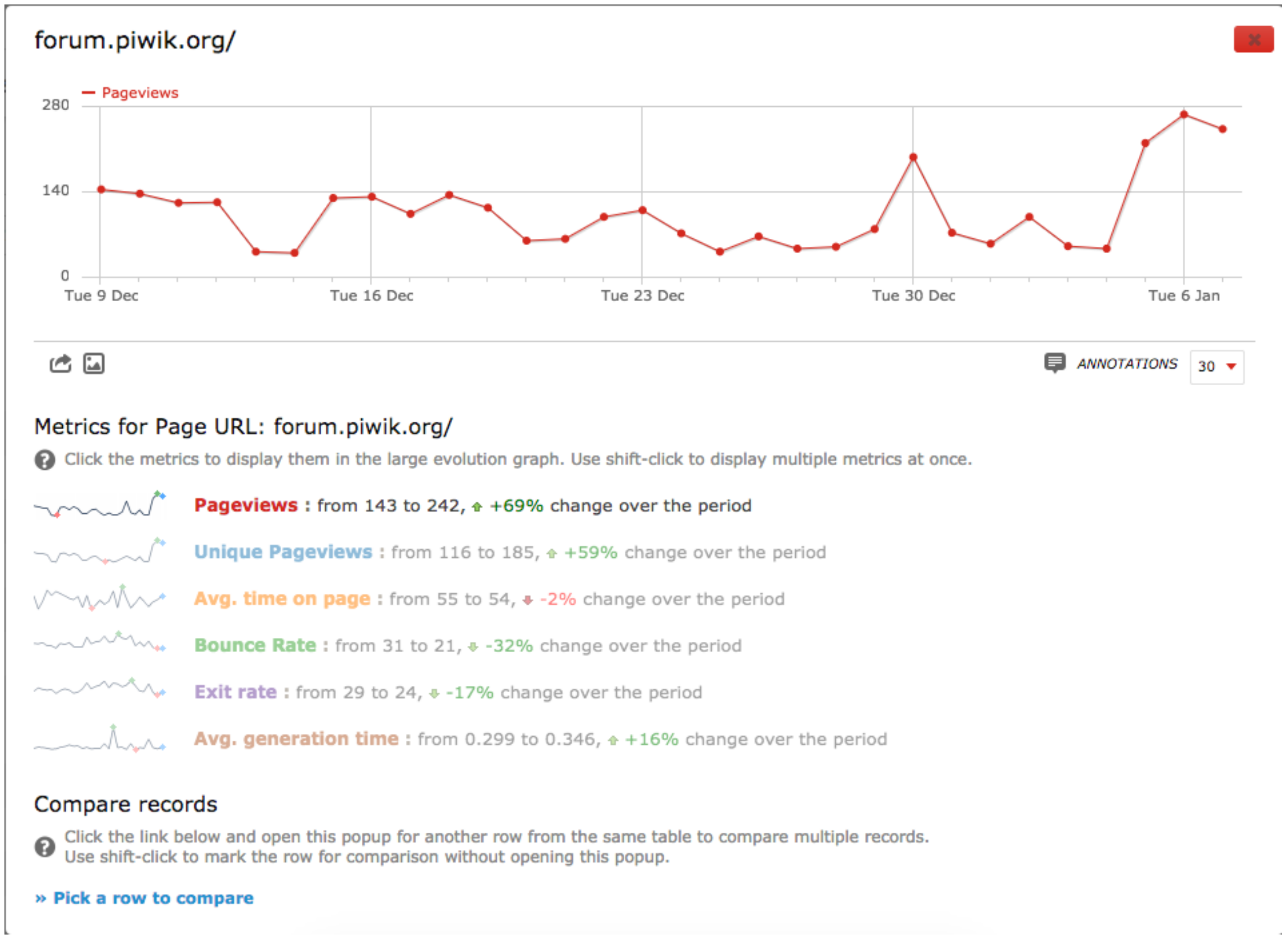
CSS File Weight Per Page
227.1 kb (235.2 kb)

Total page weight (including all assets)
910.4 kb (1600.5 kb)

PIWIK

"generation time" = `responseEnd - requestStart`

github.com/piwik/piwik



Metrics for Page URL: forum.piwik.org/

Click the metrics to display them in the large evolution graph. Use shift-click to display multiple metrics at once.

Pageviews : from 143 to 242, +69% change over the period

Unique Pageviews : from 116 to 185, +59% change over the period

Avg. time on page : from 55 to 54, -2% change over the period

Bounce Rate : from 31 to 21, -32% change over the period

Exit rate : from 29 to 24, -17% change over the period

Avg. generation time : from 0.299 to 0.346, +16% change over the period

Compare records

Click the link below and open this popup for another row from the same table to compare multiple records.
Use shift-click to mark the row for comparison without opening this popup.

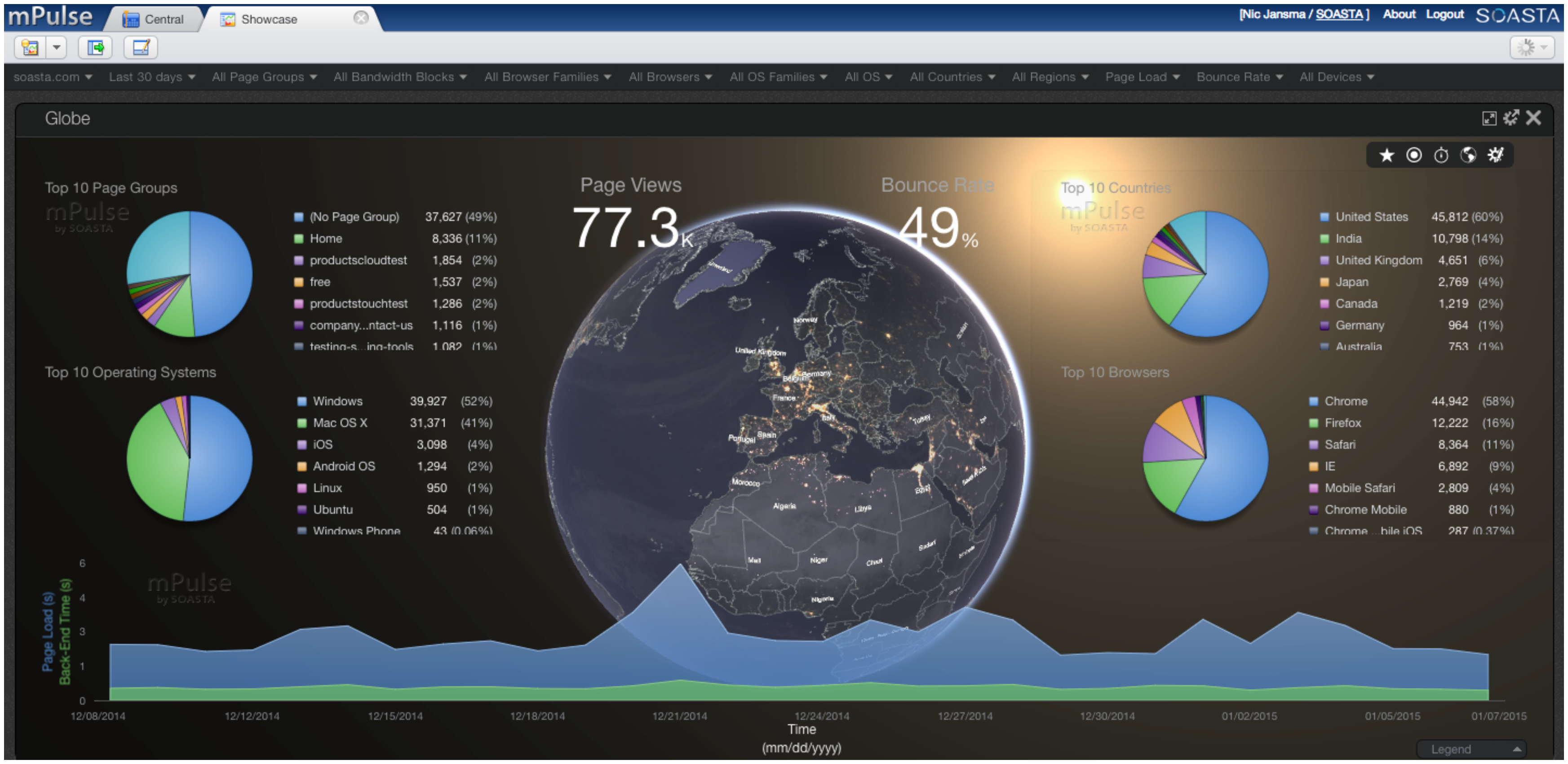
» **Pick a row to compare**

COMMERCIAL

- SOASTA mPulse: soasta.com
- Google Analytics Site Speed: google.com/analytics
- New Relic Browser: newrelic.com/browser-monitoring
- NeuStar WPM: neustar.biz
- SpeedCurve: speedcurve.com

SOASTA MPULSE

soasta.com



Top 10 Page Groups

🟦 (No Page Group)	37,627	(49%)
🟩 Home	8,336	(11%)
🟪 productscloudtest	1,854	(2%)
🟧 free	1,537	(2%)
🟪 productstouchtest	1,286	(2%)
🟪 company...ntact-us	1,116	(1%)
⬛ testing-s...ing-tools	1,082	(1%)

Top 10 Operating Systems

🟦 Windows	39,927	(52%)
🟩 Mac OS X	31,371	(41%)
🟪 iOS	3,098	(4%)
🟧 Android OS	1,294	(2%)
🟪 Linux	950	(1%)
🟪 Ubuntu	504	(1%)
⬛ Windows Phone	43	(0.06%)

Top 10 Countries

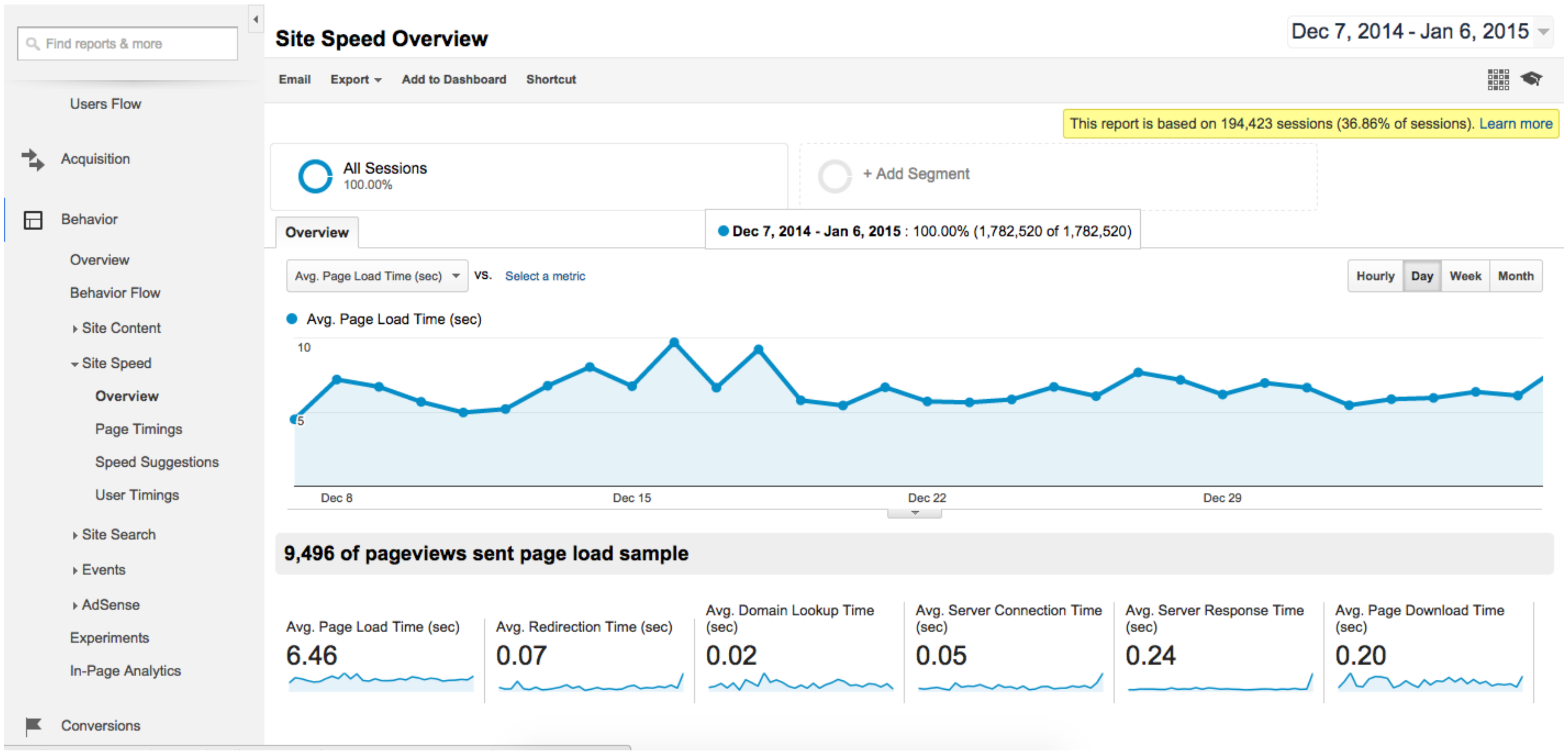
🟦 United States	45,812	(60%)
🟩 India	10,798	(14%)
🟪 United Kingdom	4,651	(6%)
🟧 Japan	2,769	(4%)
🟪 Canada	1,219	(2%)
🟪 Germany	964	(1%)
⬛ Australia	753	(1%)

Top 10 Browsers

🟦 Chrome	44,942	(58%)
🟩 Firefox	12,222	(16%)
🟪 Safari	8,364	(11%)
🟧 IE	6,892	(9%)
🟪 Mobile Safari	2,809	(4%)
🟪 Chrome Mobile	880	(1%)
⬛ Chrome ...bile iOS	287	(0.37%)

GOOGLE ANALYTICS SITE SPEED

google.com/analytics



Site Speed Overview

Dec 7, 2014 - Jan 6, 2015

This report is based on 194,423 sessions (36.86% of sessions). Learn more

All Sessions
100.00%

Dec 7, 2014 - Jan 6, 2015 : 100.00% (1,782,520 of 1,782,520)

Avg. Page Load Time (sec) vs. Select a metric

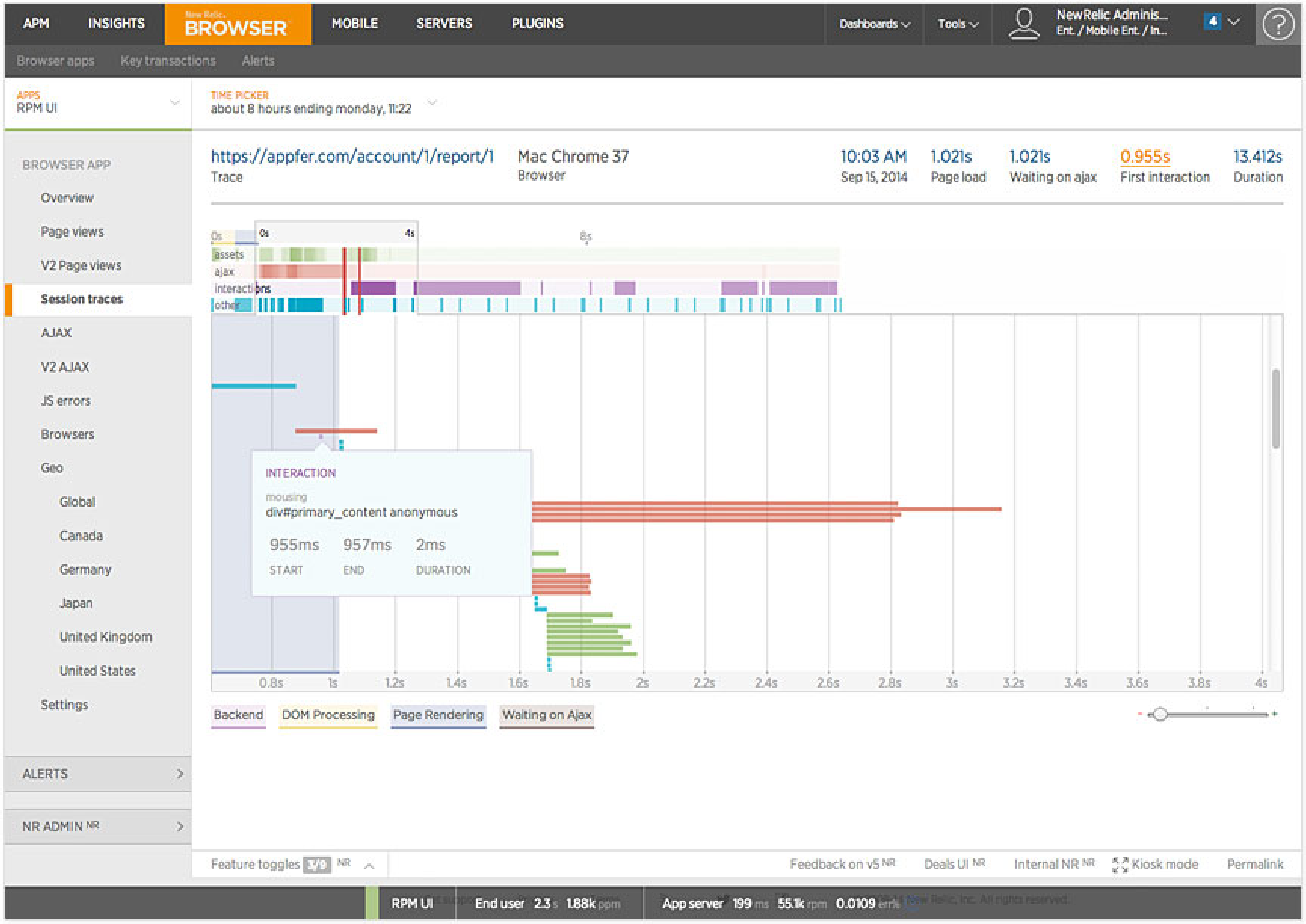
● Avg. Page Load Time (sec)

9,496 of pageviews sent page load sample

Avg. Page Load Time (sec)	Avg. Redirection Time (sec)	Avg. Domain Lookup Time (sec)	Avg. Server Connection Time (sec)	Avg. Server Response Time (sec)	Avg. Page Download Time (sec)
6.46	0.07	0.02	0.05	0.24	0.20

NEW RELIC BROWSER

newrelic.com/browser-monitoring



APPS RPM UI

TIME PICKER about 8 hours ending monday, 11:22

https://appfer.com/account/1/report/1 Trace

Mac Chrome 37 Browser

10:03 AM	1.021s	1.021s	0.955s	13.412s
Sep 15, 2014	Page load	Waiting on ajax	First interaction	Duration

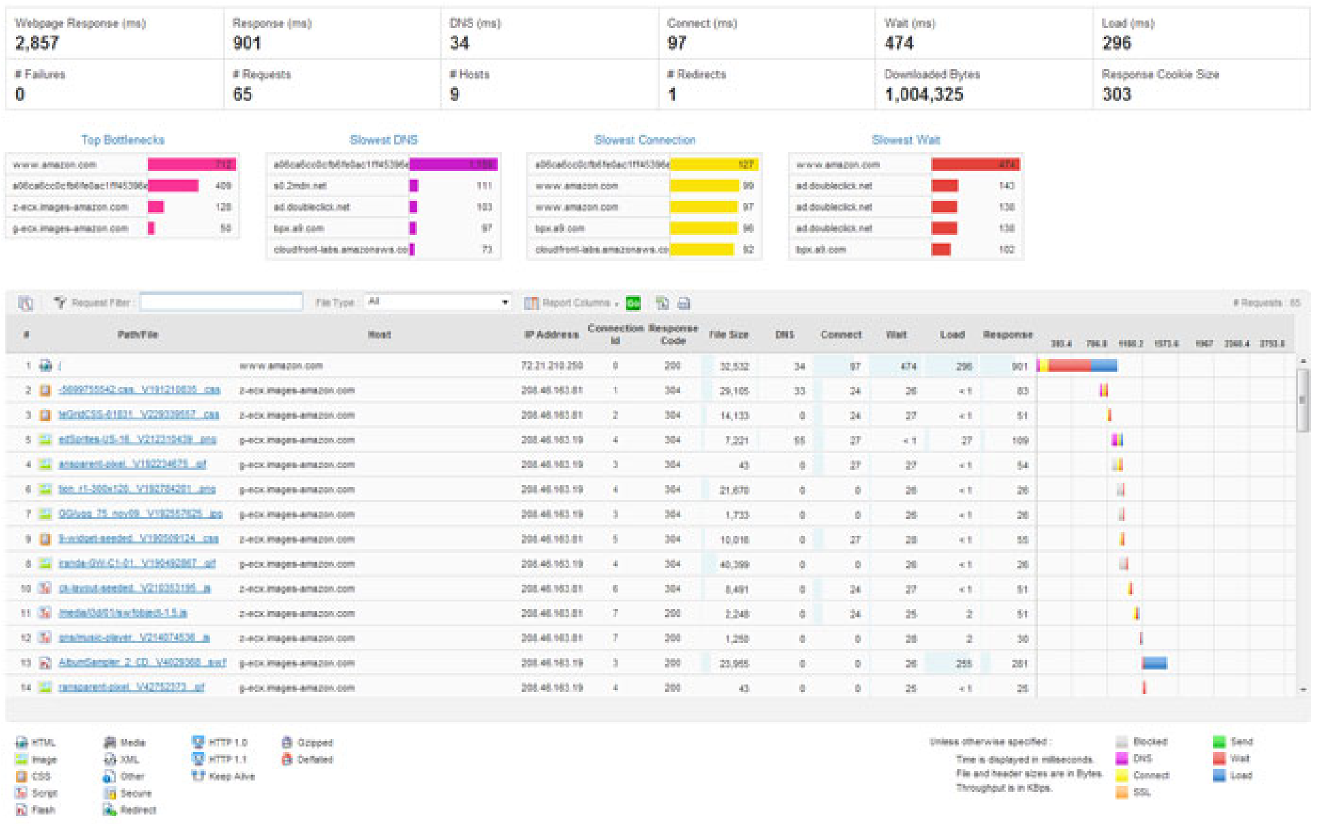
INTERACTION
mousing
div#primary_content anonymous

955ms	957ms	2ms
START	END	DURATION

RPM UI | End user 2.3s 1.88k ppm | App server 199 ms 55.1k rpm 0.0109 err%

NEUSTAR WPM

neustar.biz



Webpage Response (ms)	Response (ms)	DNS (ms)	Connect (ms)	Wait (ms)	Load (ms)
2,857	901	34	97	474	296
# Failures	**# Requests**	**# Hosts**	**# Redirects**	**Downloaded Bytes**	**Response Cookie Size**
0	65	9	1	1,004,325	303

Top Bottlenecks

www.amazon.com	712
a06ca6cc0cfb6fe0ac1ff45396e	409
z-ecx.images-amazon.com	128
g-ecx.images-amazon.com	58

Slowest DNS

a06ca6cc0cfb6fe0ac1ff45396e	1,559
a0.2mdn.net	111
ad.doubleclick.net	103
bpx.a9.com	97
cloudfront-labs.amazonaws.co	73

Slowest Connection

a06ca6cc0cfb6fe0ac1ff45396e	127
www.amazon.com	99
www.amazon.com	97
bpx.a9.com	96
cloudfront-labs.amazonaws.co	92

Slowest Wait

www.amazon.com	474
ad.doubleclick.net	143
ad.doubleclick.net	138
ad.doubleclick.net	138
bpx.a9.com	102

Requests : 65

#	Path/File	Host	IP Address	Connection Id	Response Code	File Size	DNS	Connect	Wait	Load	Response
1	/	www.amazon.com	72.21.210.250	0	200	32,532	34	97	474	296	901
2	-5699755542-css._V191210635_.css	z-ecx.images-amazon.com	208.46.163.81	1	304	29,105	33	24	26	< 1	83
3	teGridCSS-61831._V229339557_.css	z-ecx.images-amazon.com	208.46.163.81	2	304	14,133	0	24	27	< 1	51
5	edSprites-US-16._V212310439_.png	g-ecx.images-amazon.com	208.46.163.19	4	304	7,221	55	27	< 1	27	109
4	ansparent-pixel._V192234675_.gif	g-ecx.images-amazon.com	208.46.163.19	3	304	43	0	27	27	< 1	54
6	tion_r1-300x120._V192784201_.png	g-ecx.images-amazon.com	208.46.163.19	4	304	21,670	0	0	26	< 1	26
7	QGlugg_75_nov09._V192567625_.jpg	g-ecx.images-amazon.com	208.46.163.19	3	304	1,733	0	0	26	< 1	26
9	ll-widget-seeded._V190509124_.css	z-ecx.images-amazon.com	208.46.163.81	5	304	10,016	0	27	28	< 1	55
8	randa-GW-C1-01._V190492867_.gif	g-ecx.images-amazon.com	208.46.163.19	4	304	40,399	0	0	26	< 1	26
10	ck-layout-seeded._V210353195_.js	z-ecx.images-amazon.com	208.46.163.81	6	304	8,491	0	24	27	< 1	51
11	/media/0d/01/swfobject-1.5.js	z-ecx.images-amazon.com	208.46.163.81	7	200	2,248	0	24	25	2	51
12	gns/music-player._V214074536_.js	z-ecx.images-amazon.com	208.46.163.81	7	200	1,250	0	0	28	2	30
13	AlbumSampler_2_CD._V4029368_.swf	g-ecx.images-amazon.com	208.46.163.19	3	200	23,955	0	0	26	255	281
14	ransparent-pixel._V42752373_.gif	g-ecx.images-amazon.com	208.46.163.19	4	200	43	0	0	25	< 1	25

HTML	Media	HTTP 1.0	Gzipped
Image	XML	HTTP 1.1	Deflated
CSS	Other	Keep Alive	
Script	Secure		
Flash	Redirect		

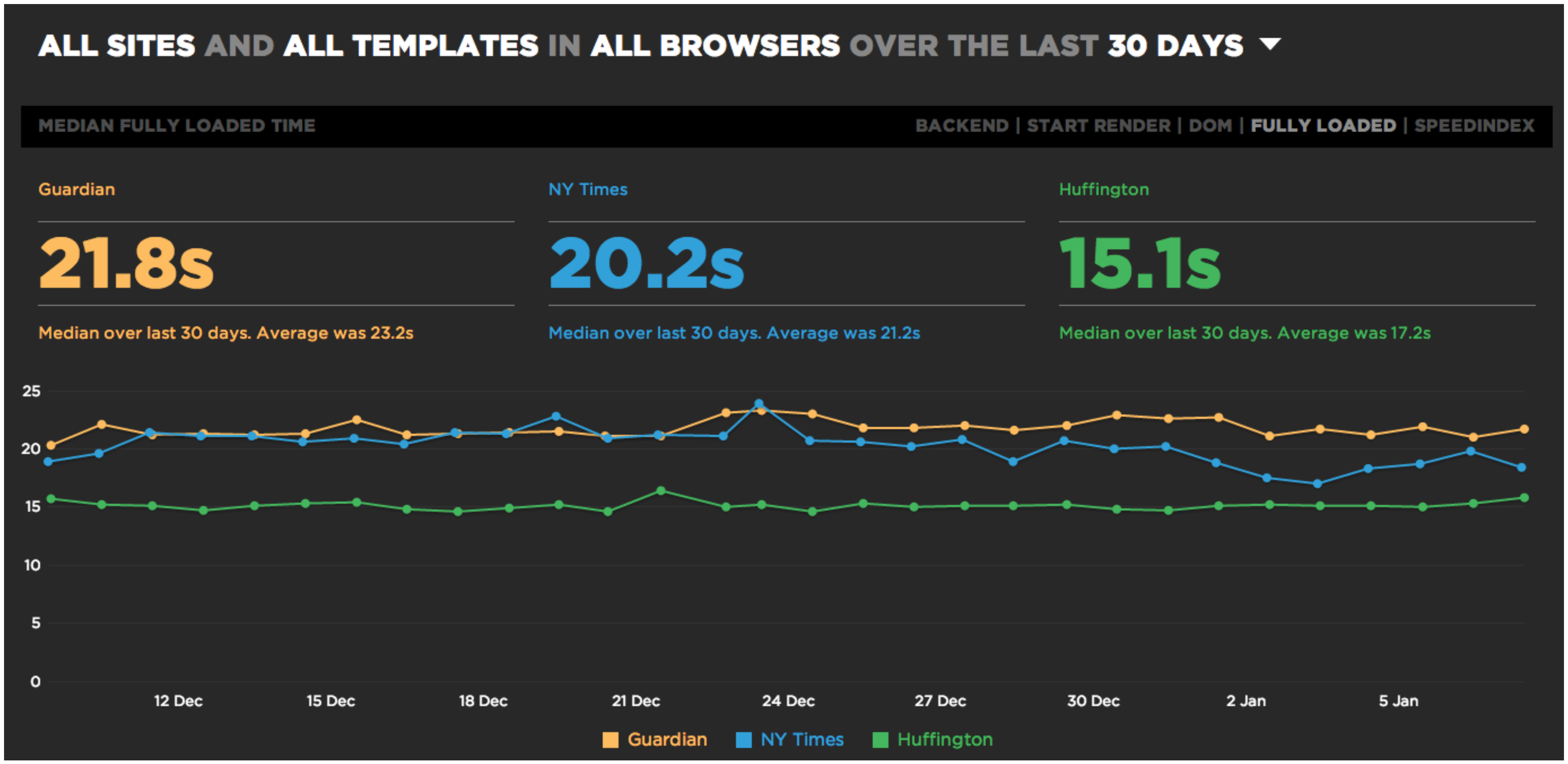
Unless otherwise specified :
Time is displayed in milliseconds.
File and header sizes are in Bytes.
Throughput is in KBps.

⬜ Blocked	🟩 Send
🟪 DNS	🟥 Wait
🟨 Connect	🟦 Load
🟧 SSL	

SPEEDCURVE

Runs on top of WebPageTest

speedcurve.com



ALL SITES AND **ALL TEMPLATES** IN **ALL BROWSERS** OVER THE LAST **30 DAYS** ▼

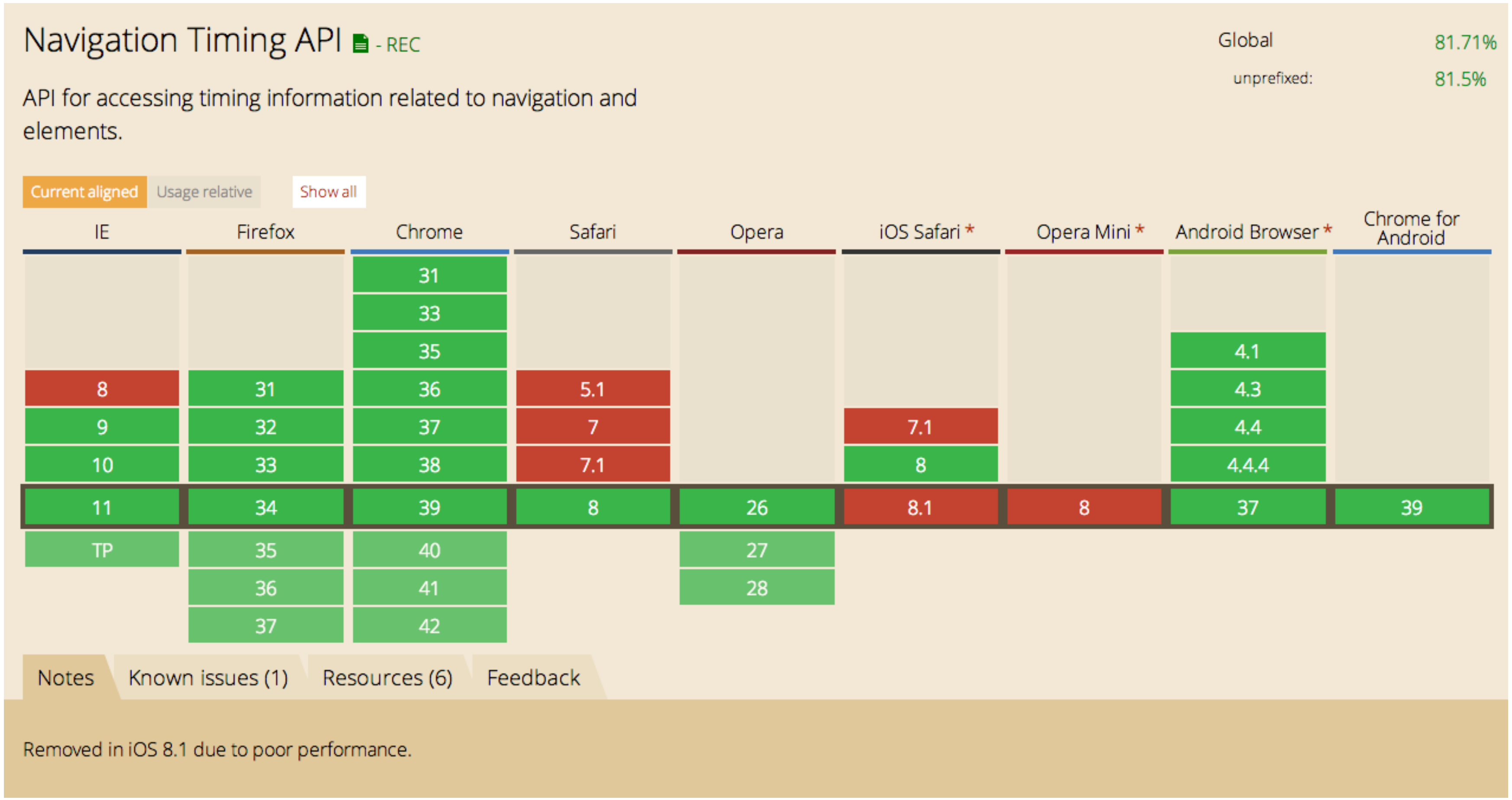
MEDIAN FULLY LOADED TIME

BACKEND | START RENDER | DOM | **FULLY LOADED** | SPEEDINDEX

Guardian	NY Times	Huffington
21.8s	20.2s	15.1s
Median over last 30 days. Average was 23.2s	Median over last 30 days. Average was 21.2s	Median over last 30 days. Average was 17.2s

NAVIGATIONTIMING

caniuse.com/#feat=nav-timing



Navigation Timing API - REC

Global 81.71%
unprefixed: 81.5%

API for accessing timing information related to navigation and elements.

Current aligned | Usage relative | Show all

IE	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
		31						
		33						
		35					4.1	
8	31	36	5.1				4.3	
9	32	37	7		7.1		4.4	
10	33	38	7.1		8		4.4.4	
11	34	39	8	26	8.1	8	37	39
TP	35	40		27				
	36	41		28				
	37	42						

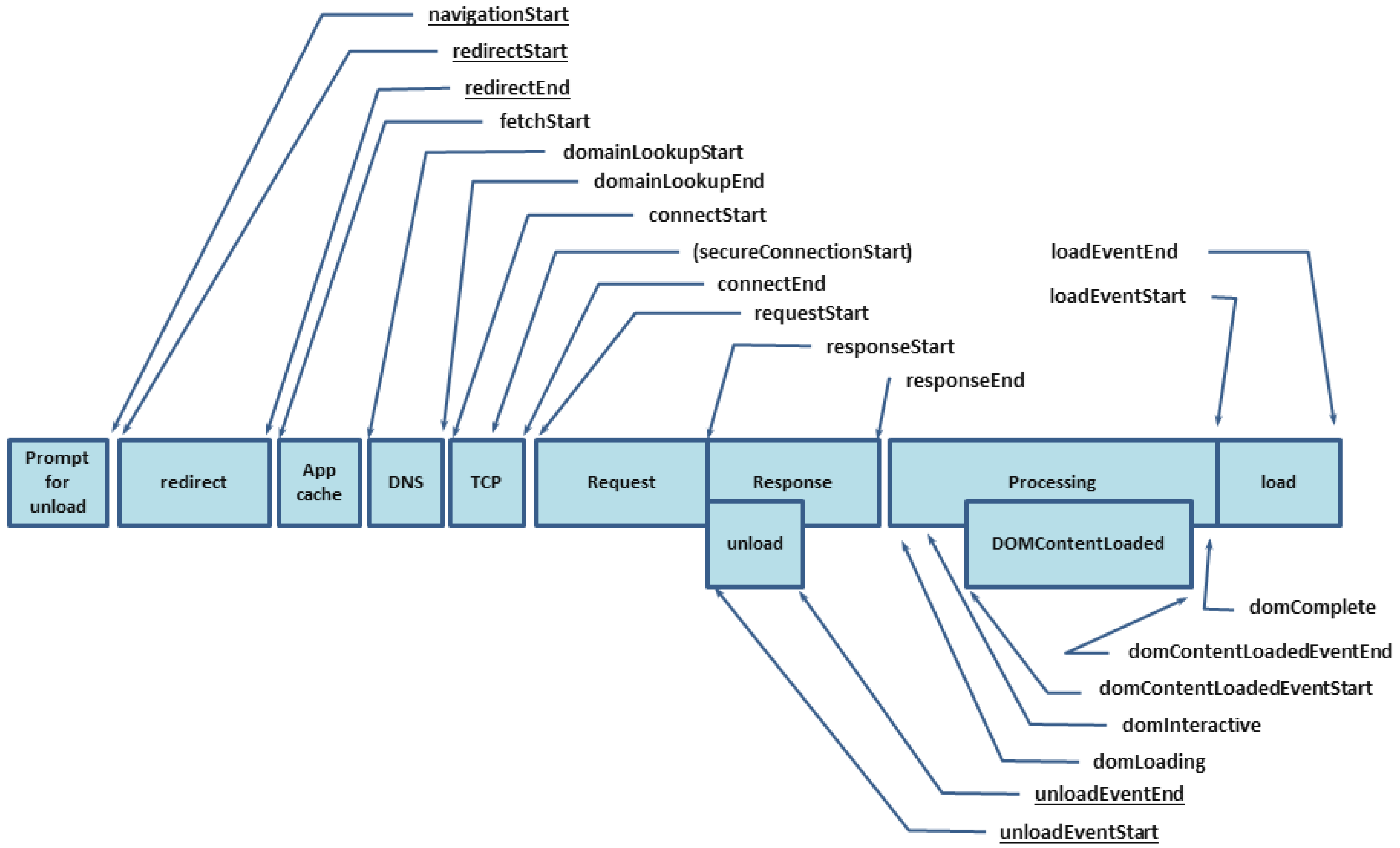
Notes | Known issues (1) | Resources (6) | Feedback

Removed in iOS 8.1 due to poor performance.

TIPS

- Use `fetchStart` instead of `navigationStart` unless you're interested in redirects, tab init time, etc
- `loadEventEnd` will be 0 until *after* the body's `load` event has finished (so you can't measure it in the `load` event)
- We don't have an accurate way to measure the "request time", as "`requestEnd`" is invisible to us (the server sees it)
- `secureConnectionStart` isn't available in IE

TIPS


navigationStart
redirectStart
redirectEnd
fetchStart
domainLookupStart
domainLookupEnd
connectStart
(secureConnectionStart)
connectEnd
requestStart
responseStart
responseEnd
loadEventEnd
loadEventStart
Prompt for unload
redirect
App cache
DNS
TCP
Request
Response
Processing
load
unload
DOMContentLoaded
domComplete
domContentLoadedEventEnd
domContentLoadedEventStart
domInteractive
domLoading
unloadEventEnd
unloadEventStart

TIPS (PT 2)

- iOS still doesn't have support
- Home page scenarios: Timestamps up through `responseEnd` event may be 0 duration because some browsers speculatively pre-fetch home pages (and don't report the correct timings)
- If possbile, do any beaconing of the data as soon as possible. Browser `onbeforeunload` isn't 100% reliable for sending data
- Single-Page Apps: You'll need a different solution for "navigations" (Boomerang + plugin coming soon)

NAVIGATIONTIMING2

www.w3.org/TR/navigation-timing-2

DRAFT

Builds on NavigationTiming:

- Support for Performance Timeline
- Support for High Resolution Time
- timing information for link negotiation
- timing information for prerender

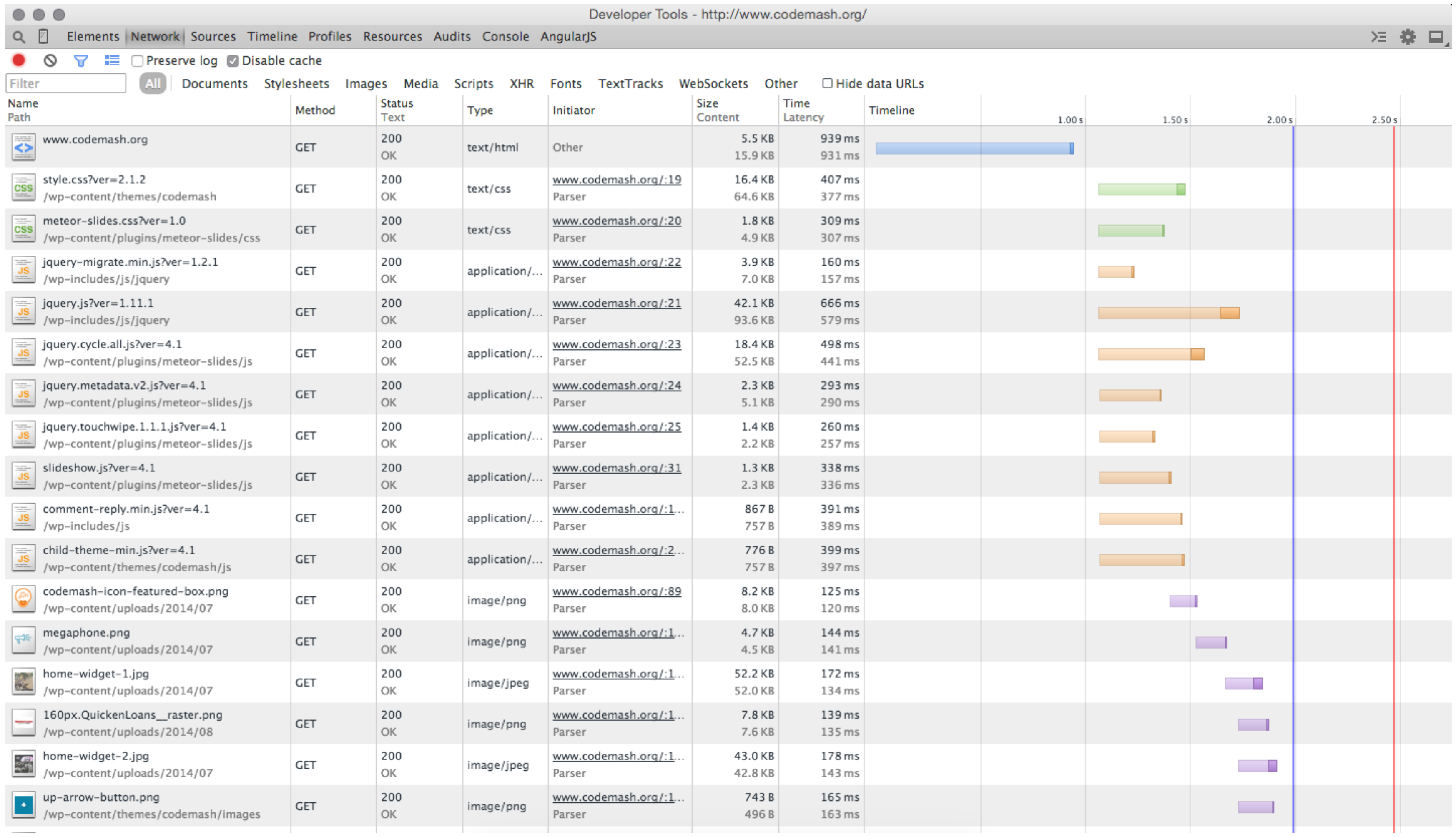
RESOURCETIMING

www.w3.org/TR/resource-timing

Goal: Expose sub-resource performance metrics

Current status: Working Draft

INSPIRATION



☐ Preserve log ☑ Disable cache ☐ Hide data URLs

Name Path	Method	Status Text	Type	Initiator	Size Content	Time Latency
www.codemash.org	GET	200 OK	text/html	Other	5.5 KB 15.9 KB	939 ms 931 ms
style.css?ver=2.1.2 /wp-content/themes/codemash	GET	200 OK	text/css	www.codemash.org/:19 Parser	16.4 KB 64.6 KB	407 ms 377 ms
meteor-slides.css?ver=1.0 /wp-content/plugins/meteor-slides/css	GET	200 OK	text/css	www.codemash.org/:20 Parser	1.8 KB 4.9 KB	309 ms 307 ms
jquery-migrate.min.js?ver=1.2.1 /wp-includes/js/jquery	GET	200 OK	application/…	www.codemash.org/:22 Parser	3.9 KB 7.0 KB	160 ms 157 ms
jquery.js?ver=1.11.1 /wp-includes/js/jquery	GET	200 OK	application/…	www.codemash.org/:21 Parser	42.1 KB 93.6 KB	666 ms 579 ms
jquery.cycle.all.js?ver=4.1 /wp-content/plugins/meteor-slides/js	GET	200 OK	application/…	www.codemash.org/:23 Parser	18.4 KB 52.5 KB	498 ms 441 ms
jquery.metadata.v2.js?ver=4.1 /wp-content/plugins/meteor-slides/js	GET	200 OK	application/…	www.codemash.org/:24 Parser	2.3 KB 5.1 KB	293 ms 290 ms
jquery.touchwipe.1.1.1.js?ver=4.1 /wp-content/plugins/meteor-slides/js	GET	200 OK	application/…	www.codemash.org/:25 Parser	1.4 KB 2.2 KB	260 ms 257 ms
slideshow.js?ver=4.1 /wp-content/plugins/meteor-slides/js	GET	200 OK	application/…	www.codemash.org/:31 Parser	1.3 KB 2.3 KB	338 ms 336 ms
comment-reply.min.js?ver=4.1 /wp-includes/js	GET	200 OK	application/…	www.codemash.org/:1… Parser	867 B 757 B	391 ms 389 ms
child-theme-min.js?ver=4.1 /wp-content/themes/codemash/js	GET	200 OK	application/…	www.codemash.org/:2… Parser	776 B 757 B	399 ms 397 ms
codemash-icon-featured-box.png /wp-content/uploads/2014/07	GET	200 OK	image/png	www.codemash.org/:89 Parser	8.2 KB 8.0 KB	125 ms 120 ms
megaphone.png /wp-content/uploads/2014/07	GET	200 OK	image/png	www.codemash.org/:1… Parser	4.7 KB 4.5 KB	144 ms 141 ms
home-widget-1.jpg /wp-content/uploads/2014/07	GET	200 OK	image/jpeg	www.codemash.org/:1… Parser	52.2 KB 52.0 KB	172 ms 134 ms
160px.QuickenLoans__raster.png /wp-content/uploads/2014/08	GET	200 OK	image/png	www.codemash.org/:1… Parser	7.8 KB 7.6 KB	139 ms 135 ms
home-widget-2.jpg /wp-content/uploads/2014/07	GET	200 OK	image/jpeg	www.codemash.org/:1… Parser	43.0 KB 42.8 KB	178 ms 143 ms
up-arrow-button.png /wp-content/themes/codemash/images	GET	200 OK	image/png	www.codemash.org/:1… Parser	743 B 496 B	165 ms 163 ms

HOW IT WAS DONE BEFORE

For dynamically inserted content, you could time how long it took from DOM insertion to the element’s onLoad event

HOW IT WAS DONE BEFORE

(this isn't practical for all content)

```
var start = new Date().getTime();
var image1 = new Image();
var resourceTiming = function() {
    var now = new Date().getTime();
    var latency = now - start;
    alert("End to end resource fetch: " + latency);
};

image1.onload = resourceTiming;
image1.src = 'http://www.w3.org/Icons/w3c_main.png';
```

WHAT'S WRONG WITH THIS?

- It measures end-to-end download time plus rendering time
- Not practical if you want to measure every resource on the page (IMG, SCRIPT, LINK rel="css", etc)
- `Date().getTime()` is not reliable

RESOURCETIMING

`window.performance.getEntries()`

```
interface PerformanceEntry {
    readonly attribute DOMString name;
    readonly attribute DOMString entryType;
    readonly attribute DOMHighResTimeStamp startTime;
    readonly attribute DOMHighResTimeStamp duration;
};

interface PerformanceResourceTiming : PerformanceEntry {
    readonly attribute DOMString initiatorType;

    readonly attribute DOMHighResTimeStamp redirectStart;
    readonly attribute DOMHighResTimeStamp redirectEnd;
    readonly attribute DOMHighResTimeStamp fetchStart;
    readonly attribute DOMHighResTimeStamp domainLookupStart;
    readonly attribute DOMHighResTimeStamp domainLookupEnd;
    readonly attribute DOMHighResTimeStamp connectStart;
    readonly attribute DOMHighResTimeStamp connectEnd;
    readonly attribute DOMHighResTimeStamp secureConnectionStart;
    readonly attribute DOMHighResTimeStamp requestStart;
    readonly attribute DOMHighResTimeStamp responseStart;
    readonly attribute DOMHighResTimeStamp responseEnd;
};
```

INTERLUDE: PERFORMANCETIMELINE

www.w3.org/TR/performance-timeline

Goal: Unifying interface to access and retrieve performance metrics

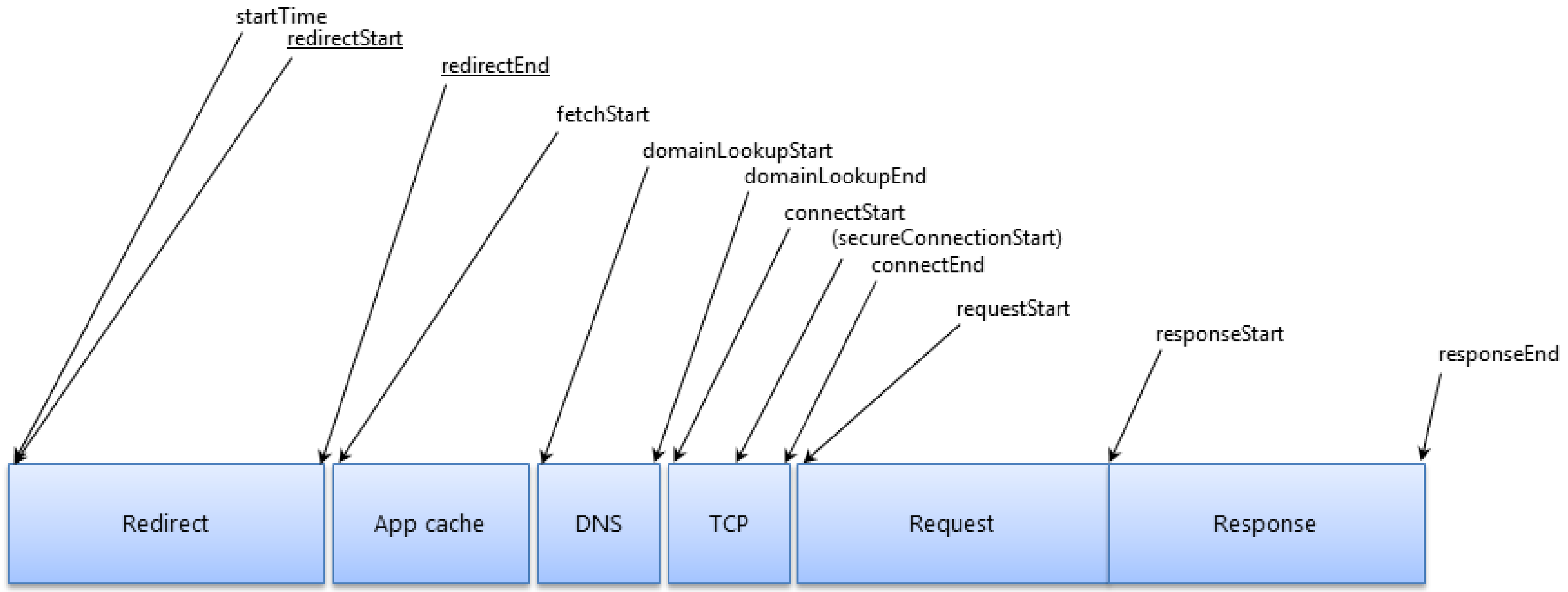
Current status: Recommendation

PERFORMANCETIMELINE

`window.performance`

- `getEntries()`: Gets all entries in the timeline
- `getEntriesByType(type)`: Gets all entries of the specified type (eg `resource`, `mark`, `measure`)
- `getEntriesByName(name)`: Gets all entries with the specified name (eg URL or mark name)

RESOURCETIMING



HOW TO USE

`window.performance.getEntriesByType("resource")[0]`

```
{
    connectEnd: 566.357000003336,
    connectStart: 566.357000003336,
    domainLookupEnd: 566.357000003336,
    domainLookupStart: 566.357000003336,
    duration: 4.275999992387369,
    entryType: "resource",
    fetchStart: 566.357000003336,
    initiatorType: "img",
    name: "https://www.foo.com/foo.png",
    redirectEnd: 0,
    redirectStart: 0,
    requestStart: 568.4959999925923,
    responseEnd: 570.6329999957234,
    responseStart: 569.4220000004862,
    secureConnectionStart: 0,
    startTime: 566.357000003336
}
```

INITIATORTYPE

`localName` of that element:

- `img`
- `link`
- `script`
- css: `url()`, `@import`
- `xmlhttprequest`

USE CASES

- Send all resource timings to your backend analytics
- Raise an analytics event if any resource takes over X seconds to download (and trend this data)
- Watch specific resources (eg third-party ads or analytics) and complain if they are slow

BUFFER

- There is a ResourceTiming buffer (per IFRAME) that stops filling after its size limit is reached (default: 150 entries)
- Listen for the `onresourcetimingbufferfull` event
- `setResourceTimingBufferSize(n)` and `clearResourceTimings()` can be used to modify it
- Don't just: `setResourceTimingBufferSize(99999999)` as this can lead to browser memory growing unbound

COMPRESSING

- Each resource is ~ 500 bytes `JSON.stringify()`'d
- HTTP Archive tells us there's 99 HTTP resources on average, per page, with an average URL length of 85 bytes
- That means you could expect around 45 KB of ResourceTiming data per page load
- Compress it: nicj.net/compressing-resourcetiming

COMPRESSING

Converts:

```
{
    "responseEnd":323.1100000002698,
    "responseStart":300.5000000000000,
    "requestStart":252.68599999981234,
    "secureConnectionStart":0,
    "connectEnd":0,
    "connectStart":0,
    "domainLookupEnd":0,
    "domainLookupStart":0,
    "fetchStart":252.68599999981234,
    "redirectEnd":0,
    "redirectStart":0,
    "duration":71.42400000045745,
    "startTime":252.68599999981234,
    "entryType":"resource",
    "initiatorType":"script",
    "name":"http://foo.com/js/foo.js"
}
```

COMPRESSING

To:

```
{
    "http://": {
        "foo.com/": {
            "js/foo.js": "370,1z,1c",
            "css/foo.css": "48c,5k,14"
        },
        "moo.com/moo.gif": "312,34,56"
    }
}
```

Overall, compresses ResourceTiming data down to 15% of its original size

github.com/nicjansma/resourcetiming-compression.js

TIMING-ALLOW-ORIGIN

- By default, cross-origin resources expose timestamps for only the `fetchStart` and `responseEnd` attributes
- This is to protect your privacy (attacker can't load random URLs to see where you've been)
- Override by setting `Timing-Allow-Origin` header
- `Timing-Allow-Origin = "Timing-Allow-Origin" ":" origin-list-or-null | "*"`
- If you have a CDN, **use this**
- **Note**: Third-party libraries (ads, analytics, etc) must set this on their servers. 5% do according to HTTP Archive. Google, Facebook, Disqus, mPulse, etc.

BLOCKING TIME

- Browsers will open a limited number of connections to each unique origin (protocol/server name/port)
- If there are more resources than the # of connections, the later resources will be "blocking", waiting for their turn to download
- `duration` includes Blocking time!
- So in general, don't use `duration`, but this is all you get with cross-origin resources.

BLOCKING TIME

Calculate:

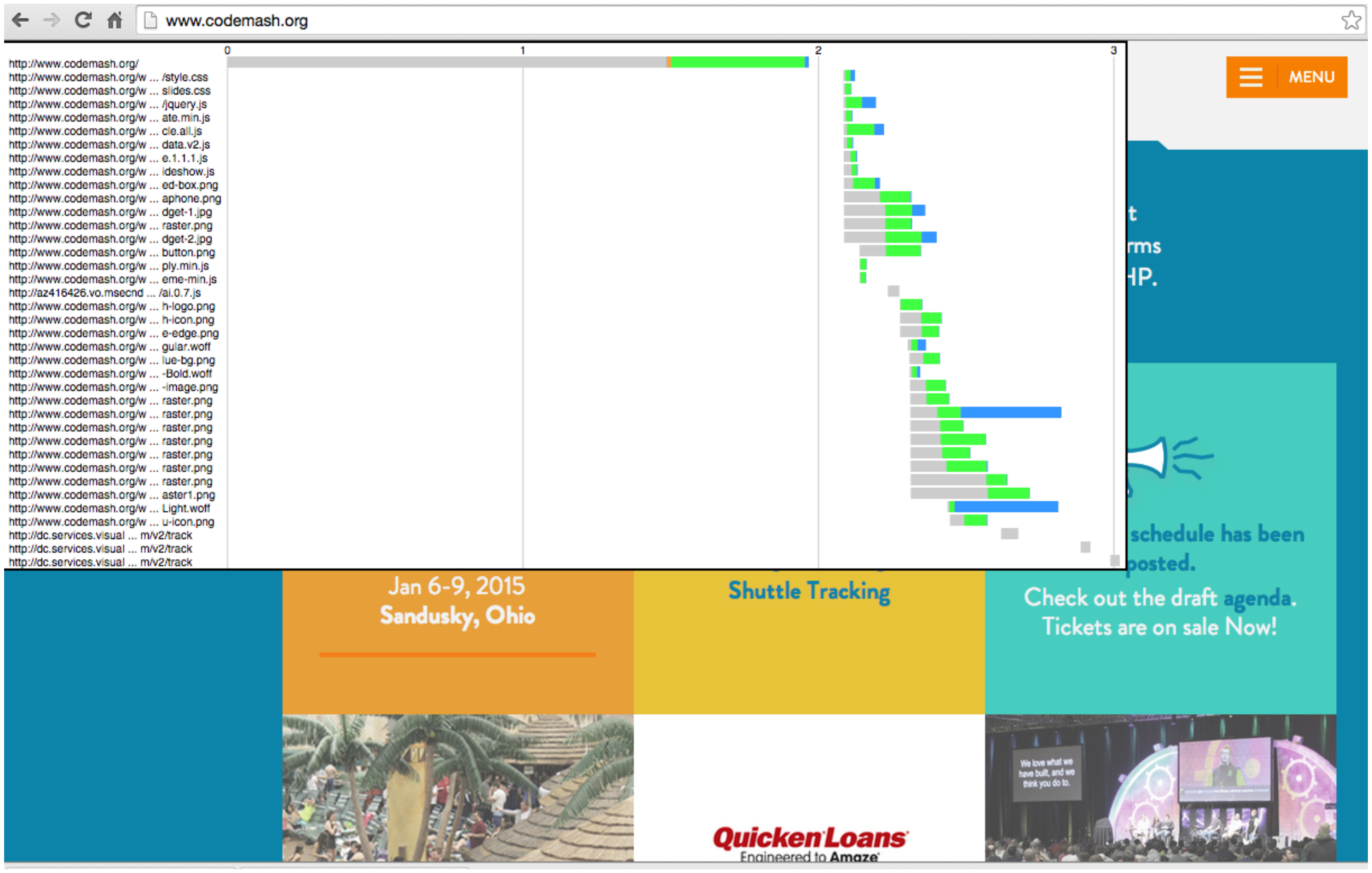
```
var waitTime = 0;
if (res.connectEnd && res.connectEnd === res.fetchStart)
{
    waitTime = res.requestStart - res.connectEnd;
}
else if (res.domainLookupStart)
{
    waitTime = res.domainLookupStart - res.fetchStart;
}
```

DIY / OPEN SOURCE

- Compress + send this data to your backend for logging
- Show any page's resources via a bookmarklet: github.com/andydavies/waterfall
- Heatmap bookmarklet / Chrome extension: github.com/zeman/perfmap
- Nurun's Performance Bookmarklet: github.com/nurun/performance-bookmarklet
- Boomerang supports ResourceTiming: github.com/lognormal/boomerang

ANDY DAVIES' WATERFALL.JS

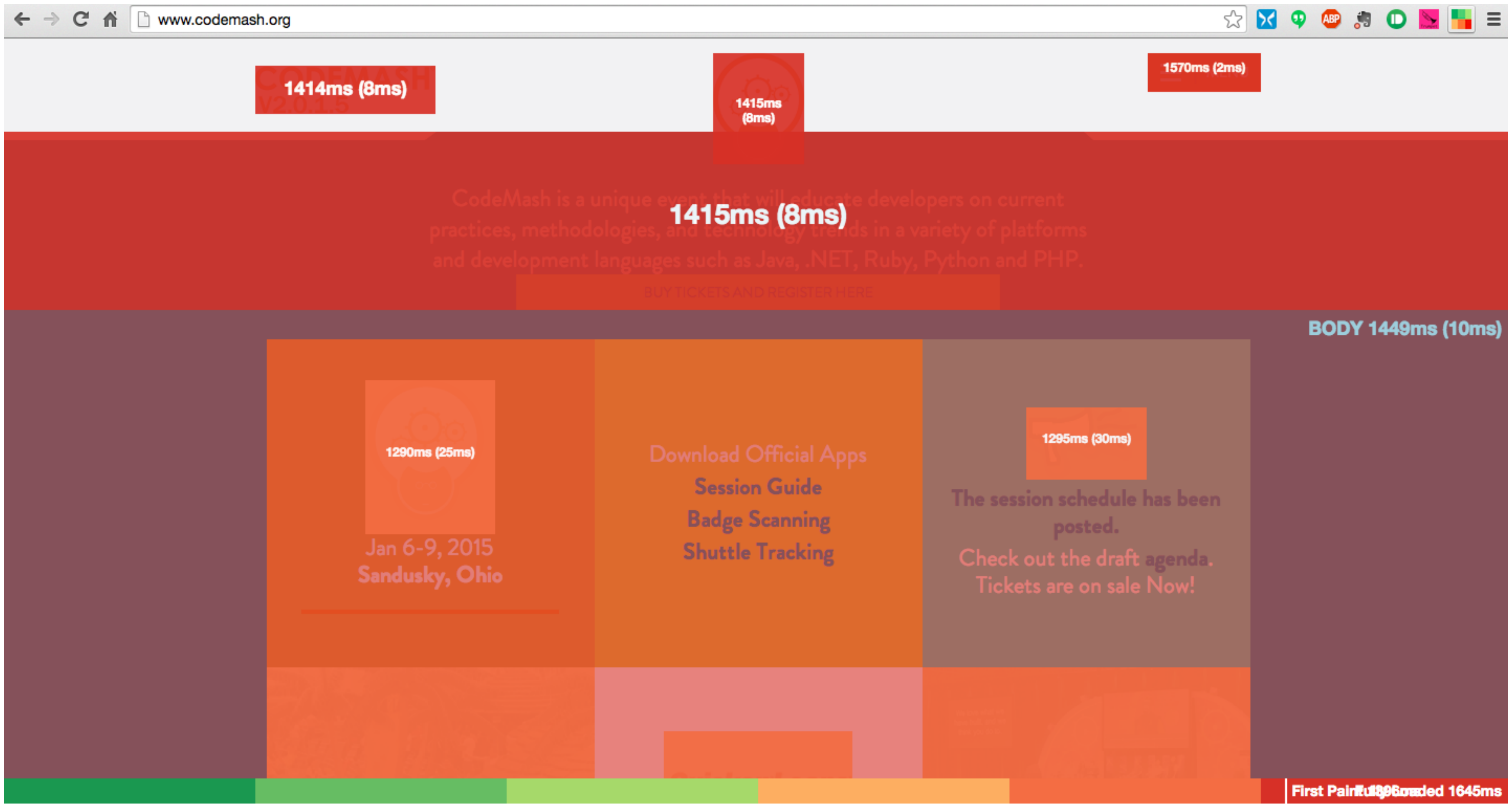
github.com/andydavies/waterfall



```
http://www.codemash.org/
http://www.codemash.org/w ... /style.css
http://www.codemash.org/w ... slides.css
http://www.codemash.org/w ... /jquery.js
http://www.codemash.org/w ... ate.min.js
http://www.codemash.org/w ... cle.all.js
http://www.codemash.org/w ... data.v2.js
http://www.codemash.org/w ... e.1.1.1.js
http://www.codemash.org/w ... ideshow.js
http://www.codemash.org/w ... ed-box.png
http://www.codemash.org/w ... aphone.png
http://www.codemash.org/w ... dget-1.jpg
http://www.codemash.org/w ... raster.png
http://www.codemash.org/w ... dget-2.jpg
http://www.codemash.org/w ... button.png
http://www.codemash.org/w ... ply.min.js
http://www.codemash.org/w ... eme-min.js
http://az416426.vo.msecnd ... /ai.0.7.js
http://www.codemash.org/w ... h-logo.png
http://www.codemash.org/w ... h-icon.png
http://www.codemash.org/w ... e-edge.png
http://www.codemash.org/w ... gular.woff
http://www.codemash.org/w ... lue-bg.png
http://www.codemash.org/w ... -Bold.woff
http://www.codemash.org/w ... -image.png
http://www.codemash.org/w ... raster.png
http://www.codemash.org/w ... raster.png
http://www.codemash.org/w ... raster.png
http://www.codemash.org/w ... raster.png
http://www.codemash.org/w ... raster.png
http://www.codemash.org/w ... raster.png
http://www.codemash.org/w ... raster.png
http://www.codemash.org/w ... aster1.png
http://www.codemash.org/w ... Light.woff
http://www.codemash.org/w ... u-icon.png
http://dc.services.visual ... m/v2/track
http://dc.services.visual ... m/v2/track
http://dc.services.visual ... m/v2/track
```

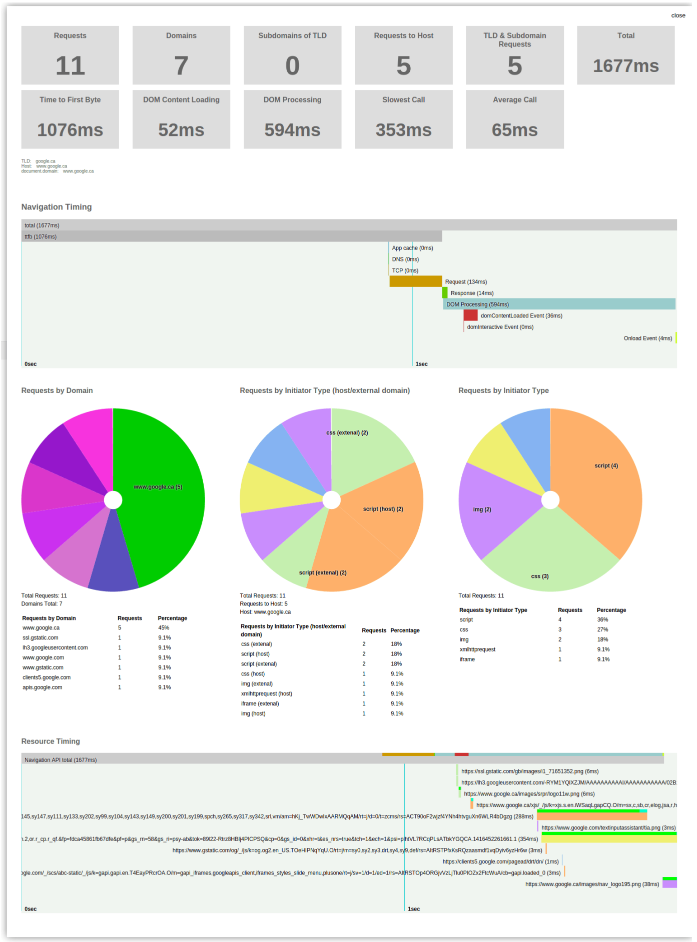
MARK ZEMAN'S PERFMAP

github.com/zeman/perfmap


www.codemash.org
1414ms (8ms)
1415ms (8ms)
1570ms (2ms)
1415ms (8ms)
CodeMash is a unique event that will educate developers on current practices, methodologies, and technology trends in a variety of platforms and development languages such as Java, .NET, Ruby, Python and PHP.
BUY TICKETS AND REGISTER HERE
BODY 1449ms (10ms)
1290ms (25ms)
Jan 6-9, 2015
Sandusky, Ohio
Download Official Apps
Session Guide
Badge Scanning
Shuttle Tracking
1295ms (30ms)
The session schedule has been posted.
Check out the draft agenda.
Tickets are on sale Now!
Fully Loaded 1645ms

NURUN'S PERFORMANCE BOOKMARKLET

github.com/nurun/performance-bookmarklet



close

Requests	Domains	Subdomains of TLD	Requests to Host	TLD & Subdomain Requests	Total
11	7	0	5	5	1677ms

Time to First Byte	DOM Content Loading	DOM Processing	Slowest Call	Average Call
1076ms	52ms	594ms	353ms	65ms

TLD: google.ca
Host: www.google.ca
document.domain: www.google.ca

Navigation Timing

Requests by Domain

Total Requests: 11
Domains Total: 7

Requests by Domain	Requests	Percentage
www.google.ca	5	45%
ssl.gstatic.com	1	9.1%
lh3.googleusercontent.com	1	9.1%
www.google.com	1	9.1%
www.gstatic.com	1	9.1%
clients5.google.com	1	9.1%
apis.google.com	1	9.1%

Requests by Initiator Type (host/external domain)

Total Requests: 11
Requests to Host: 5
Host: www.google.ca

Requests by Initiator Type (host/external domain)	Requests	Percentage
css (extenal)	2	18%
script (host)	2	18%
script (extenal)	2	18%
css (host)	1	9.1%
img (extenal)	1	9.1%
xmlhttprequest (host)	1	9.1%
iframe (extenal)	1	9.1%
img (host)	1	9.1%

Requests by Initiator Type

Total Requests: 11

Requests by Initiator Type	Requests	Percentage
script	4	36%
css	3	27%
img	2	18%
xmlhttprequest	1	9.1%
iframe	1	9.1%

Resource Timing

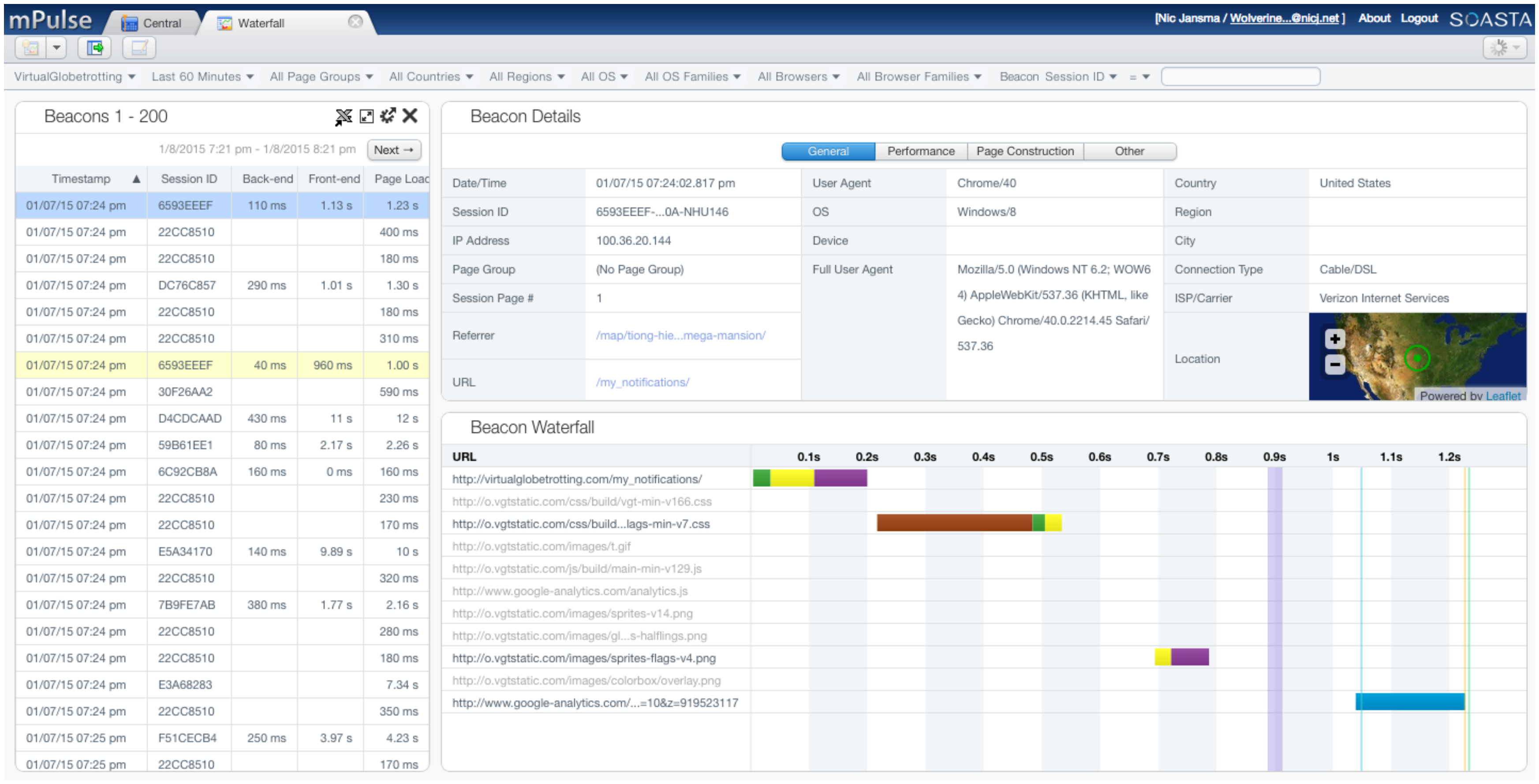
```
Navigation API total (1677ms)
https://ssl.gstatic.com/gb/images/i1_71651352.png (6ms)
https://lh3.googleusercontent.com/-RYM1YQIXZJM/AAAAAAAAAAI/AAAAAAAAAAA/02B
https://www.google.ca/images/srpr/logo11w.png (6ms)
https://www.google.ca/xjs/_/js/k=xjs.s.en.iWSaqLgapCQ.O/m=sx,c,sb,cr,elog,jsa,r,h
145,sy147,sy111,sy133,sy202,sy99,sy104,sy143,sy149,sy200,sy201,sy199,spch,sy265,sy317,sy342,srl,vm/am=hKj_TwWDwIxAARMQqAM/rt=j/d=0/t=zcms/rs=ACT90oF2wjzf4YNh4htvguXn6WLR4bDgzg (288ms)
https://www.google.com/textinputassistant/tia.png (3ms)
.2,or.r_cp.r_qf.&fp=fdca45861fb67dfe&pf=p&gs_rn=58&gs_ri=psy-ab&tok=89l22-Rtrz8HBIj4PICPSQ&cp=0&gs_id=0&xhr=t&es_nrs=true&tch=1&ech=1&psi=plhtVL7RCqPLsATbkYGQCA.1416452261661.1 (354ms)
https://www.gstatic.com/og/_/js/k=og.og2.en_US.TOeHIPNqYqU.O/rt=j/m=sy0,sy2,sy3,drt,sy4,sy9,def/rs=AItRSTPfxKsRQzaasmdf1vqDyiv6yzHr6w (3ms)
https://clients5.google.com/pagead/drt/dn/ (1ms)
ogle.com/_/scs/abc-static/_/js/k=gapi.gapi.en.T4EayPRcrOA.O/m=gapi_iframes,googleapis_client,iframes_styles_slide_menu,plusone/rt=j/sv=1/d=1/ed=1/rs=AItRSTOp4ORGjvVzLjTlu0PIOZx2FtcWuA/cb=gapi.loaded_0 (3ms)
https://www.google.ca/images/nav_logo195.png (38ms)
0sec
1sec
```

COMMERCIAL

- SOASTA mPulse: soasta.com
- New Relic Browser: newrelic.com
- App Dynamics Web EUEM: appdynamics.com

SOASTA MPULSE

soasta.com



mPulse | Central | Waterfall

[Nic Jansma / Wolverine...@nicj.net] About Logout SOASTA

VirtualGlobetrotting ▾ Last 60 Minutes ▾ All Page Groups ▾ All Countries ▾ All Regions ▾ All OS ▾ All OS Families ▾ All Browsers ▾ All Browser Families ▾ Beacon Session ID ▾ = ▾

Beacons 1 - 200

1/8/2015 7:21 pm - 1/8/2015 8:21 pm Next →

Timestamp ▲	Session ID	Back-end	Front-end	Page Loac
01/07/15 07:24 pm	6593EEEF	110 ms	1.13 s	1.23 s
01/07/15 07:24 pm	22CC8510			400 ms
01/07/15 07:24 pm	22CC8510			180 ms
01/07/15 07:24 pm	DC76C857	290 ms	1.01 s	1.30 s
01/07/15 07:24 pm	22CC8510			180 ms
01/07/15 07:24 pm	22CC8510			310 ms
01/07/15 07:24 pm	6593EEEF	40 ms	960 ms	1.00 s
01/07/15 07:24 pm	30F26AA2			590 ms
01/07/15 07:24 pm	D4CDCAAD	430 ms	11 s	12 s
01/07/15 07:24 pm	59B61EE1	80 ms	2.17 s	2.26 s
01/07/15 07:24 pm	6C92CB8A	160 ms	0 ms	160 ms
01/07/15 07:24 pm	22CC8510			230 ms
01/07/15 07:24 pm	22CC8510			170 ms
01/07/15 07:24 pm	E5A34170	140 ms	9.89 s	10 s
01/07/15 07:24 pm	22CC8510			320 ms
01/07/15 07:24 pm	7B9FE7AB	380 ms	1.77 s	2.16 s
01/07/15 07:24 pm	22CC8510			280 ms
01/07/15 07:24 pm	22CC8510			180 ms
01/07/15 07:24 pm	E3A68283			7.34 s
01/07/15 07:24 pm	22CC8510			350 ms
01/07/15 07:25 pm	F51CECB4	250 ms	3.97 s	4.23 s
01/07/15 07:25 pm	22CC8510			170 ms

Beacon Details

General | Performance | Page Construction | Other

Date/Time	01/07/15 07:24:02.817 pm	User Agent	Chrome/40	Country	United States
Session ID	6593EEEF-...0A-NHU146	OS	Windows/8	Region	
IP Address	100.36.20.144	Device		City	
Page Group	(No Page Group)	Full User Agent	Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/40.0.2214.45 Safari/537.36	Connection Type	Cable/DSL
Session Page #	1			ISP/Carrier	Verizon Internet Services
Referrer	/map/tiong-hie...mega-mansion/			Location	Powered by Leaflet
URL	/my_notifications/				

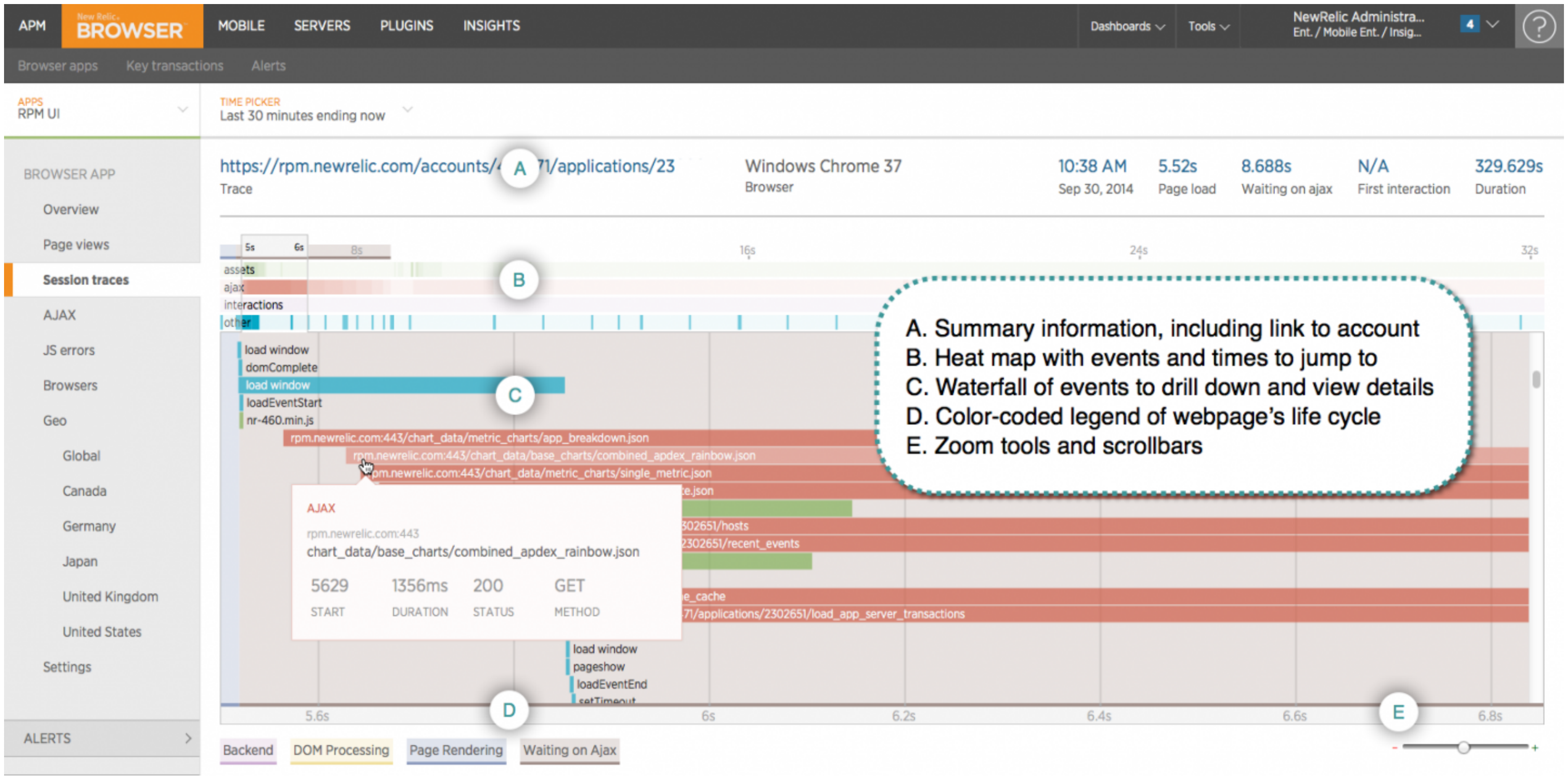
Beacon Waterfall

URL | 0.1s 0.2s 0.3s 0.4s 0.5s 0.6s 0.7s 0.8s 0.9s 1s 1.1s 1.2s

http://virtualglobetrotting.com/my_notifications/
http://o.vgtstatic.com/css/build/vgt-min-v166.css
http://o.vgtstatic.com/css/build...lags-min-v7.css
http://o.vgtstatic.com/images/t.gif
http://o.vgtstatic.com/js/build/main-min-v129.js
http://www.google-analytics.com/analytics.js
http://o.vgtstatic.com/images/sprites-v14.png
http://o.vgtstatic.com/images/gl...s-halflings.png
http://o.vgtstatic.com/images/sprites-flags-v4.png
http://o.vgtstatic.com/images/colorbox/overlay.png
http://www.google-analytics.com/...=10&z=919523117

NEW RELIC BROWSER

newrelic.com/browser-monitoring



https://rpm.newrelic.com/accounts/…71/applications/23	Windows Chrome 37	10:38 AM	5.52s	8.688s	N/A	329.629s
Trace	Browser	Sep 30, 2014	Page load	Waiting on ajax	First interaction	Duration

AJAX

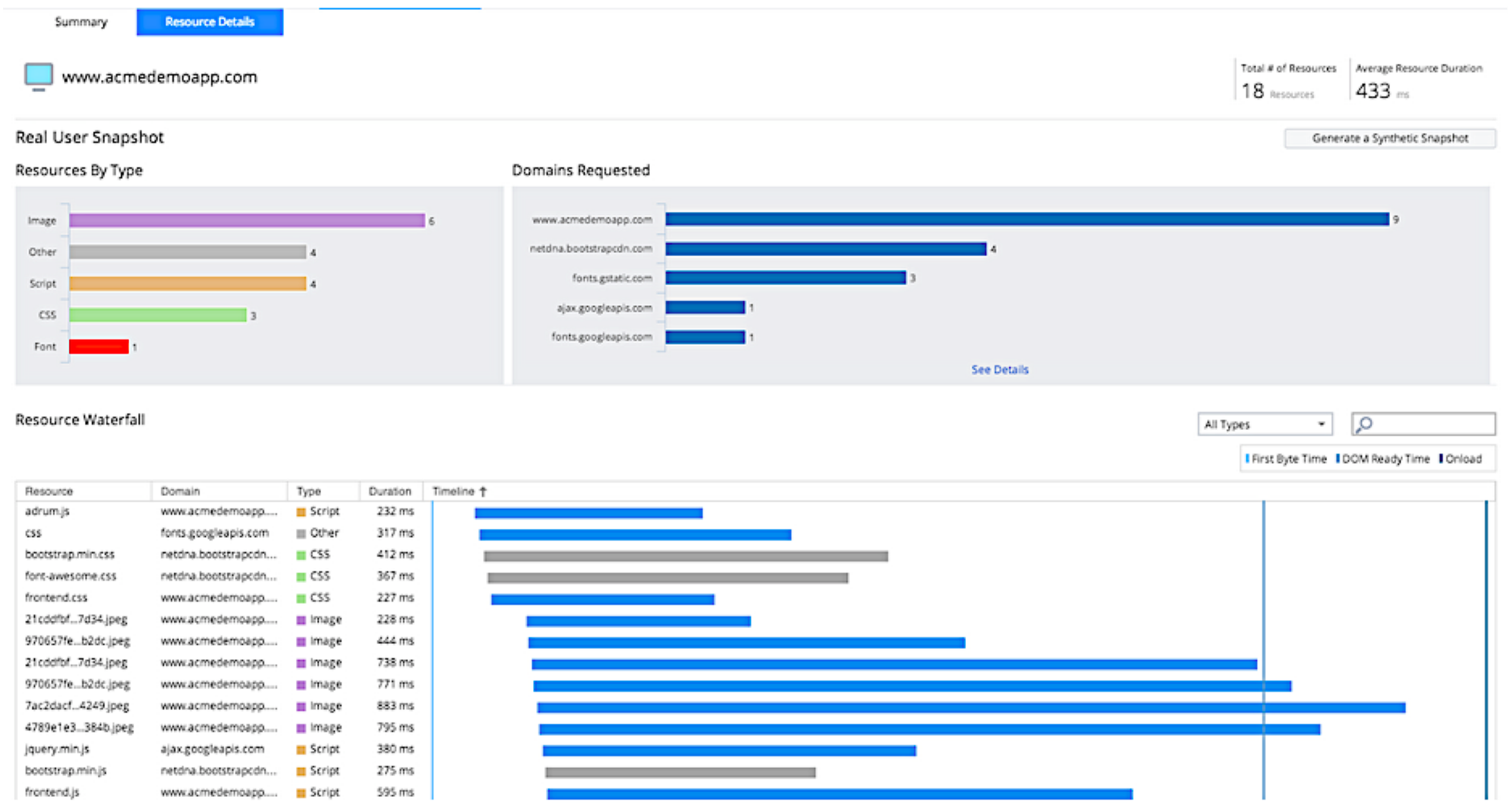
rpm.newrelic.com:443
chart_data/base_charts/combined_apdex_rainbow.json

5629	1356ms	200	GET
START	DURATION	STATUS	METHOD

A. Summary information, including link to account
B. Heat map with events and times to jump to
C. Waterfall of events to drill down and view details
D. Color-coded legend of webpage's life cycle
E. Zoom tools and scrollbars

APP DYNAMICS WEB EUEM

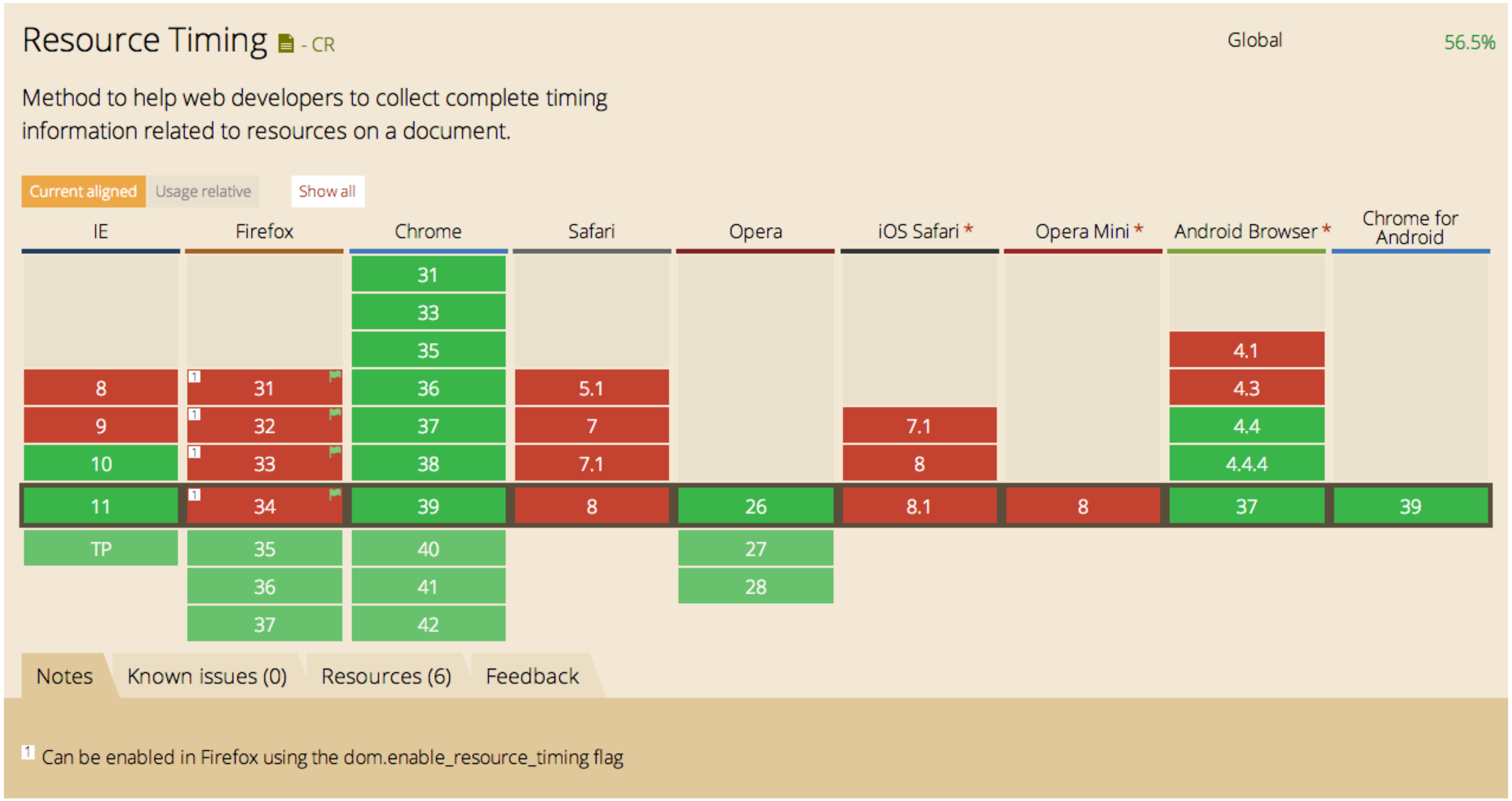
appdynamics.com



Resource	Domain	Type	Duration	Timeline ↑
adrum.js	www.acmedemoapp....	🟧 Script	232 ms	
css	fonts.googleapis.com	⬜ Other	317 ms	
bootstrap.min.css	netdna.bootstrapcdn....	🟩 CSS	412 ms	
font-awesome.css	netdna.bootstrapcdn....	🟩 CSS	367 ms	
frontend.css	www.acmedemoapp....	🟩 CSS	227 ms	
21cddfbf...7d34.jpeg	www.acmedemoapp....	🟪 Image	228 ms	
970657fe...b2dc.jpeg	www.acmedemoapp....	🟪 Image	444 ms	
21cddfbf...7d34.jpeg	www.acmedemoapp....	🟪 Image	738 ms	
970657fe...b2dc.jpeg	www.acmedemoapp....	🟪 Image	771 ms	
7ac2dacf...4249.jpeg	www.acmedemoapp....	🟪 Image	883 ms	
4789e1e3...384b.jpeg	www.acmedemoapp....	🟪 Image	795 ms	
jquery.min.js	ajax.googleapis.com	🟧 Script	380 ms	
bootstrap.min.js	netdna.bootstrapcdn....	🟧 Script	275 ms	
frontend.js	www.acmedemoapp....	🟧 Script	595 ms	

RESOURCETIMING

caniuse.com/#feat=resource-timing



Resource Timing - CR

Global 56.5%

Method to help web developers to collect complete timing information related to resources on a document.

Current aligned | Usage relative | Show all

IE	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
		🟩 31						
		🟩 33						
		🟩 35					🟥 4.1	
🟥 8	🟥 [1] 31	🟩 36	🟥 5.1				🟥 4.3	
🟥 9	🟥 [1] 32	🟩 37	🟥 7		🟥 7.1		🟩 4.4	
🟩 10	🟥 [1] 33	🟩 38	🟥 7.1		🟥 8		🟩 4.4.4	
🟩 11	🟥 [1] 34	🟩 39	🟥 8	🟩 26	🟥 8.1	🟥 8	🟩 37	🟩 39
🟩 TP	🟩 35	🟩 40		🟩 27				
	🟩 36	🟩 41		🟩 28				
	🟩 37	🟩 42						

Notes | Known issues (0) | Resources (6) | Feedback

[1] Can be enabled in Firefox using the dom.enable_resource_timing flag

TIPS

- For many sites, most of your content will not be same-origin, so ensure all of your CDNs and third-party libraries send `Timing-Allow-Origin`
- What isn't included in ResourceTiming:
  - The root HTML page (get this from `window.performance.timing`)
  - Transfer size or content size (privacy concerns)
  - HTTP code (privacy concerns)
  - Content that loaded with errors (eg 404s)

TIPS (PT 2)

- If you're going to be managing the ResourceTiming buffer, make sure no other scripts are managing it as well
- The `duration` attribute includes Blocking time (when a resource is behind other resources on the same socket)
- Each `IFRAME` will have its own ResourceTiming data, and those resources won't be included in the parent FRAME/document. So you'll need to traverse the document frames to get all resources. See github.com/nicjansma/resourcetiming-compression.js for an example
- `about:blank`, `javascript:` URLs will be seen in RT data

USERTIMING

www.w3.org/TR/user-timing

Goal: Standardized interface to note timestamps ("marks") and durations ("measures")

Current status: Recommendation

HOW IT WAS DONE BEFORE

```
var start = new Date().getTime();
// do stuff
var now = new Date().getTime();
var duration = now - start;
```

WHAT'S WRONG WITH THIS?

- *Nothing really, but...*
- `Date().getTime()` is not reliable
- We can do better!

USERTIMING

window.performance

```
partial interface Performance {
    void mark(DOMString markName);

    void clearMarks(optional  DOMString markName);

    void measure(DOMString measureName, optional DOMString startMark,
        optional DOMString endMark);

    void clearMeasures(optional DOMString measureName);
};
```

HOW TO USE - MARK

```
// mark
performance.mark("start");
performance.mark("end");

performance.mark("another");
performance.mark("another");
performance.mark("another");
```

HOW TO USE - MARK

```
// retrieve
performance.getEntriesByType("mark");

[
    {
        "duration":0,
        "startTime":150384.48100000096,
        "entryType":"mark",
        "name":"start"
    },
    {
        "duration":0,
        "startTime":150600.5250000013,
        "entryType":"mark",
        "name":"end"
    },
    ...
]
```

HOW TO USE - MEASURE

```
// measure
performance.mark("start");
// do work
performance.mark("start2");

// measure from "now" to the "start" mark
performance.measure("time to do stuff", "start");

// measure from "start2" to the "start" mark
performance.measure("time from start to start2", "start", "start2");
```

HOW TO USE - MEASURE

```
// retrieval - specific
performance.getEntriesByName("time from start to start2", "measure");

[
    {
        "duration":4809.890999997151,
        "startTime":145287.66500000347,
        "entryType":"measure",
        "name":"time from start to start2"
    }
]
```

BENEFITS

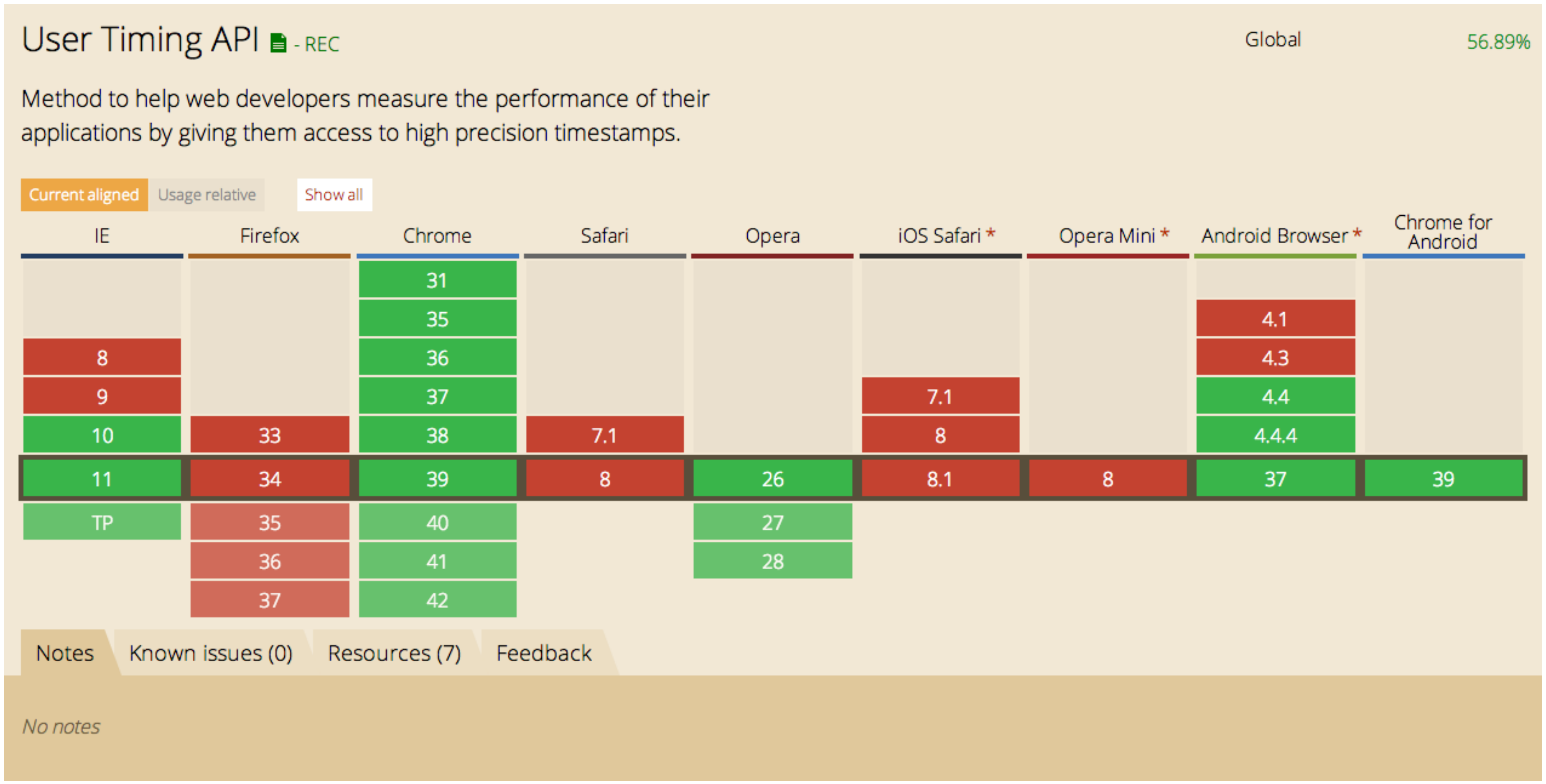
- Uses the `PerformanceTimeline`, so `marks` and `measures` are in the PerformanceTimeline along with other events
- Uses `DOMHighResTimestamp` instead of `Date` so sub-millisecond, monotonically non-decreasing, etc
- More efficient, as the native browser runtime can do math quicker and store things more performantly than your JavaScript runtime can

USE CASES

- Easy way to add profiling events to your application
- Note important scenario durations in your Performance Timeline
- Measure important durations for analytics
- Browser tools are starting to add support for showing these

USERTIMING

caniuse.com/#feat=user-timing



User Timing API - REC

Global 56.89%

Method to help web developers measure the performance of their applications by giving them access to high precision timestamps.

IE	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
		🟩 31						
		🟩 35					🟥 4.1	
🟥 8		🟩 36					🟥 4.3	
🟥 9		🟩 37			🟥 7.1		🟩 4.4	
🟩 10	🟥 33	🟩 38	🟥 7.1		🟥 8		🟩 4.4.4	
🟩 11	🟥 34	🟩 39	🟥 8	🟩 26	🟥 8.1	🟥 8	🟩 37	🟩 39
🟩 TP	🟥 35	🟩 40		🟩 27				
	🟥 36	🟩 41		🟩 28				
	🟥 37	🟩 42						

No notes

USERTIMING.JS

- Polyfill that adds UserTiming support to browsers that do not natively support it.
- UserTiming is accessed via the PerformanceTimeline, and requires window.performance.now() support, so UserTiming.js adds a limited version of these interfaces if the browser does not support them
- github.com/nicjansma/usertiming.js

DIY / OPEN SOURCE

- Compress + send this data to your backend for logging
- WebPageTest sends UserTiming to Google Analytics, Boomerang and SOASTA mPulse

COMMERCIAL

- SOASTA mPulse: soasta.com
- WebPageTest: webpagetest.org

TIPS

- Not the same as Google Analytic's "User Timings" API (`_trackTiming(...)`)

YOUR JOB

MAKE IT FAST!

LINKS

- Presentation: slideshare.net/nicjansma
- Code: github.com/nicjansma/talks

Thanks - Nic Jansma - nicj.net - @NicJ