

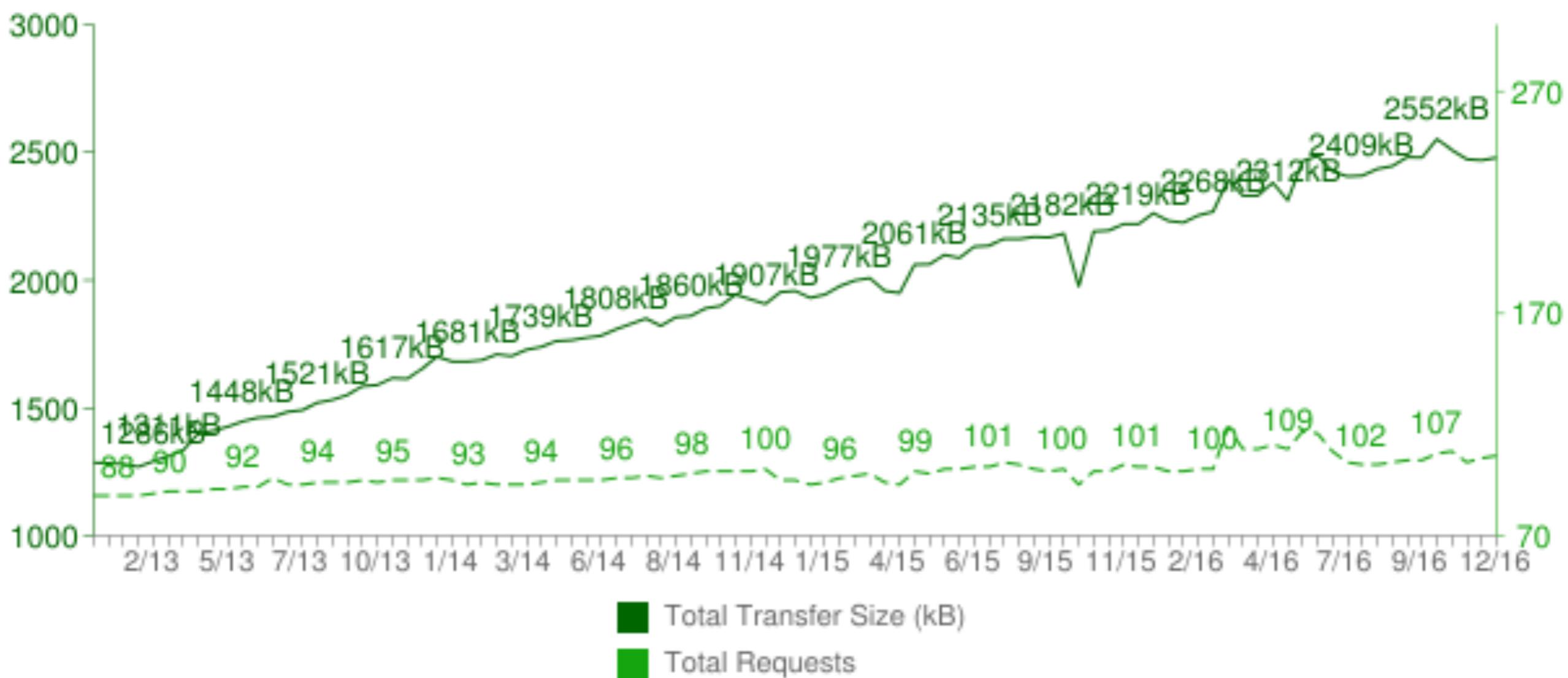
Critical Rendering Path

Declan Rek

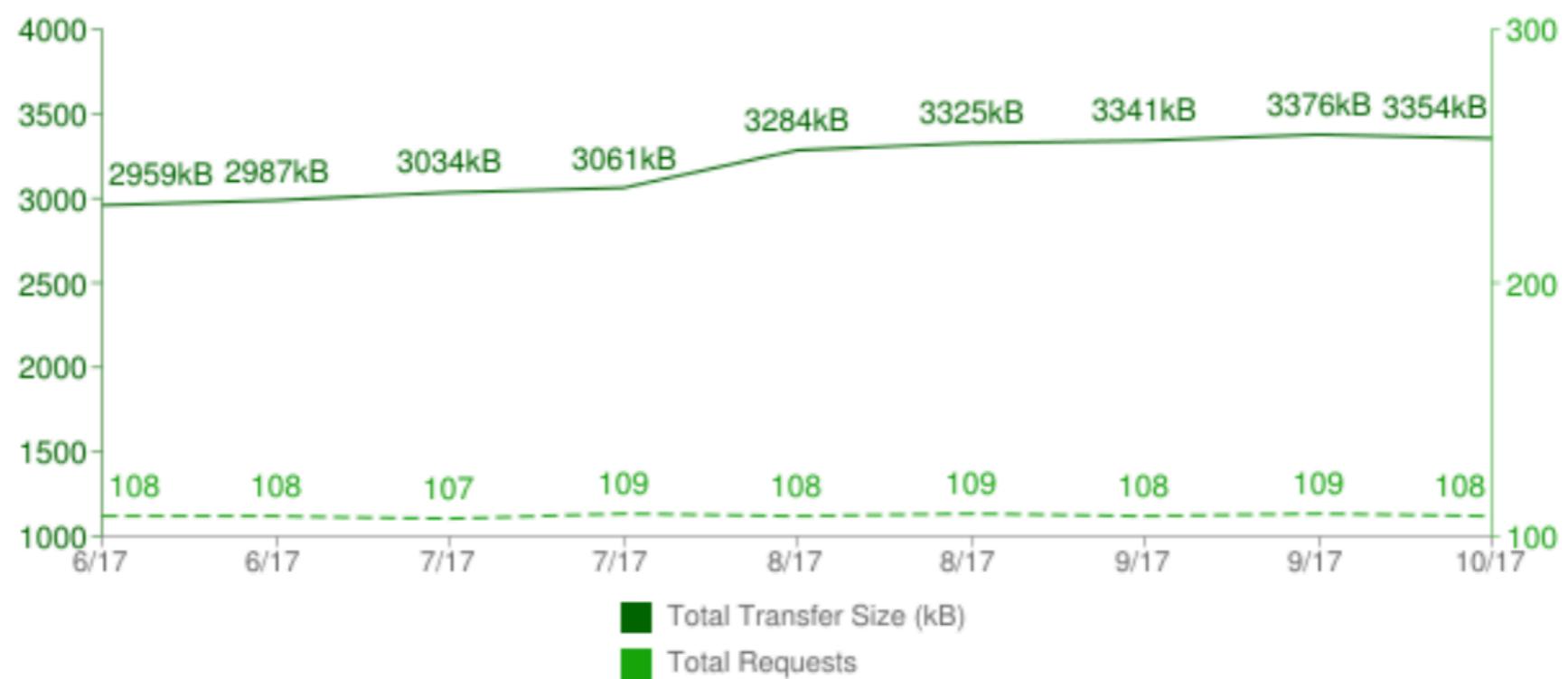
PERFORMANCE MATTERS

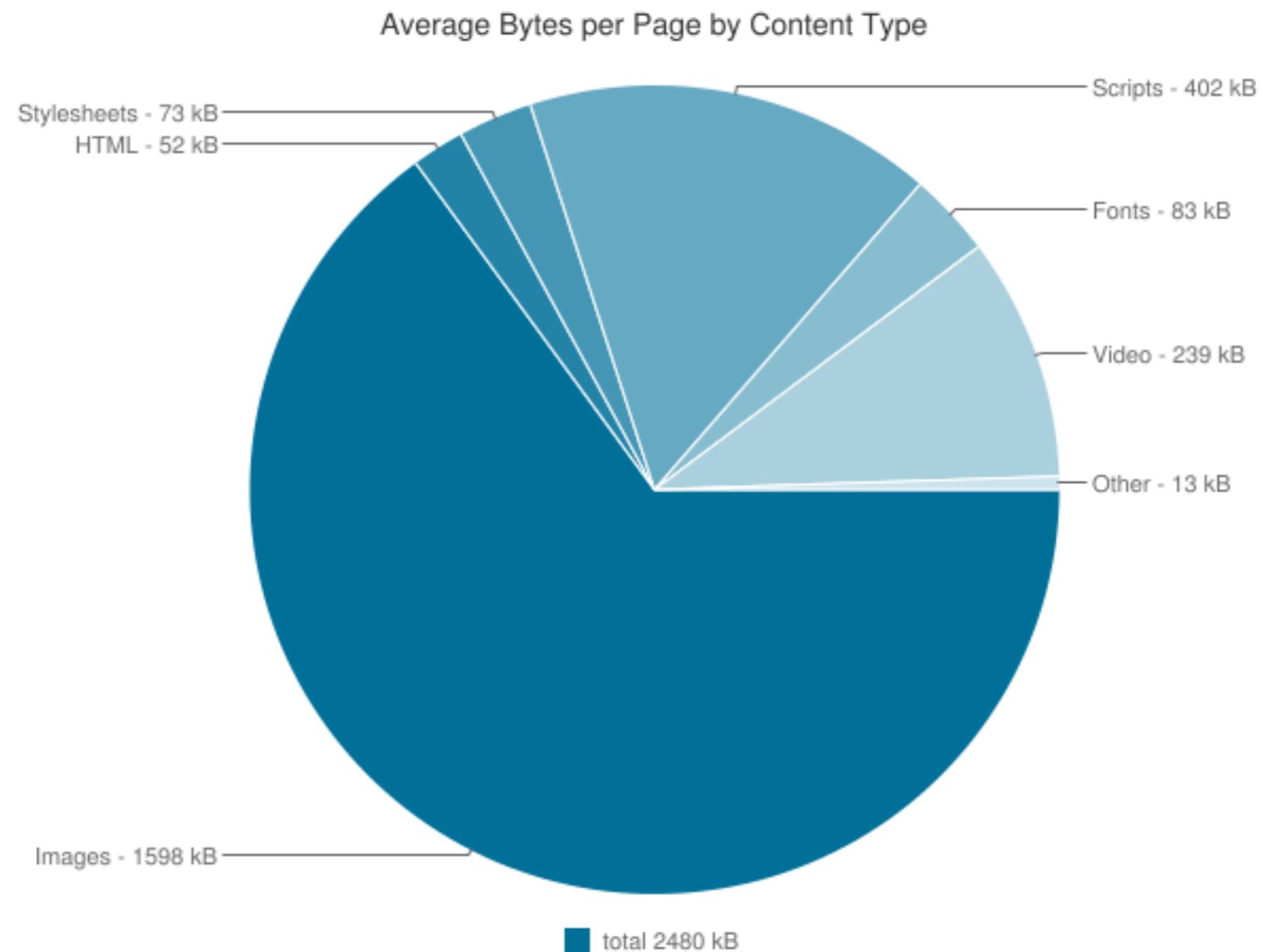


Total Transfer Size & Total Requests

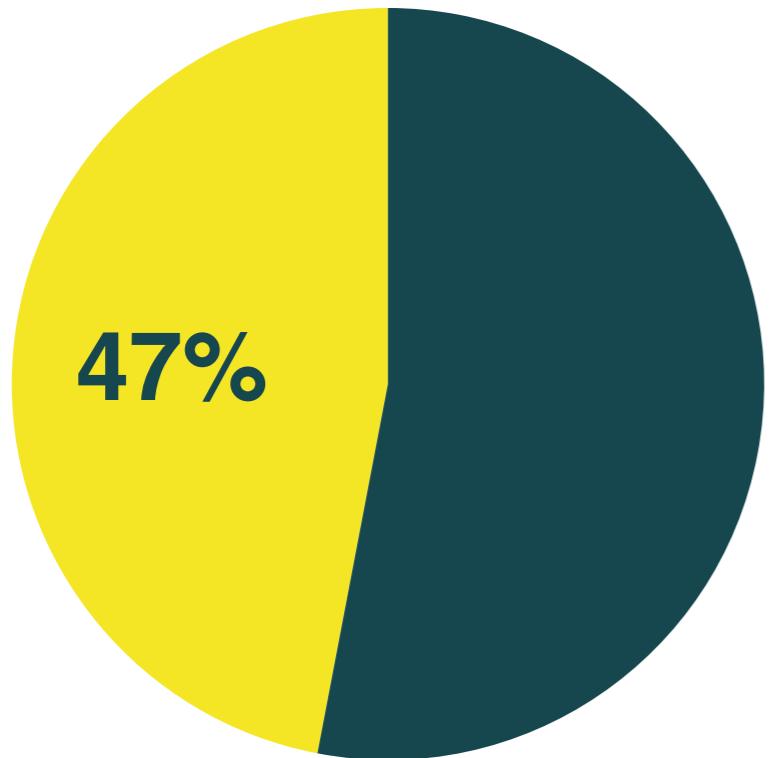


Total Transfer Size & Total Requests

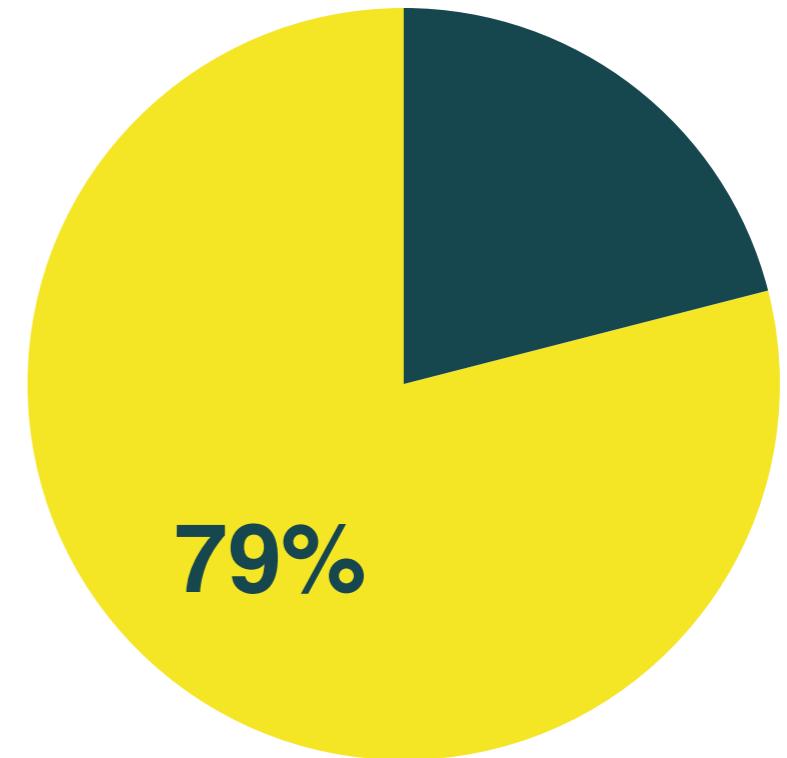




HIGH EXPECTATIONS



47% of consumers expect a webpage to load within 2 seconds



79% of online shoppers will not return to a website after a disappointing experience due to poor performance

Fast websites have more visitors, who visit more pages, for longer period of times, who come back more often, and are more likely to buy

PERCEIVED PERFORMANCE



Perceived performance refers
to how fast a user thinks your
website is



Photo by Daniel Hong

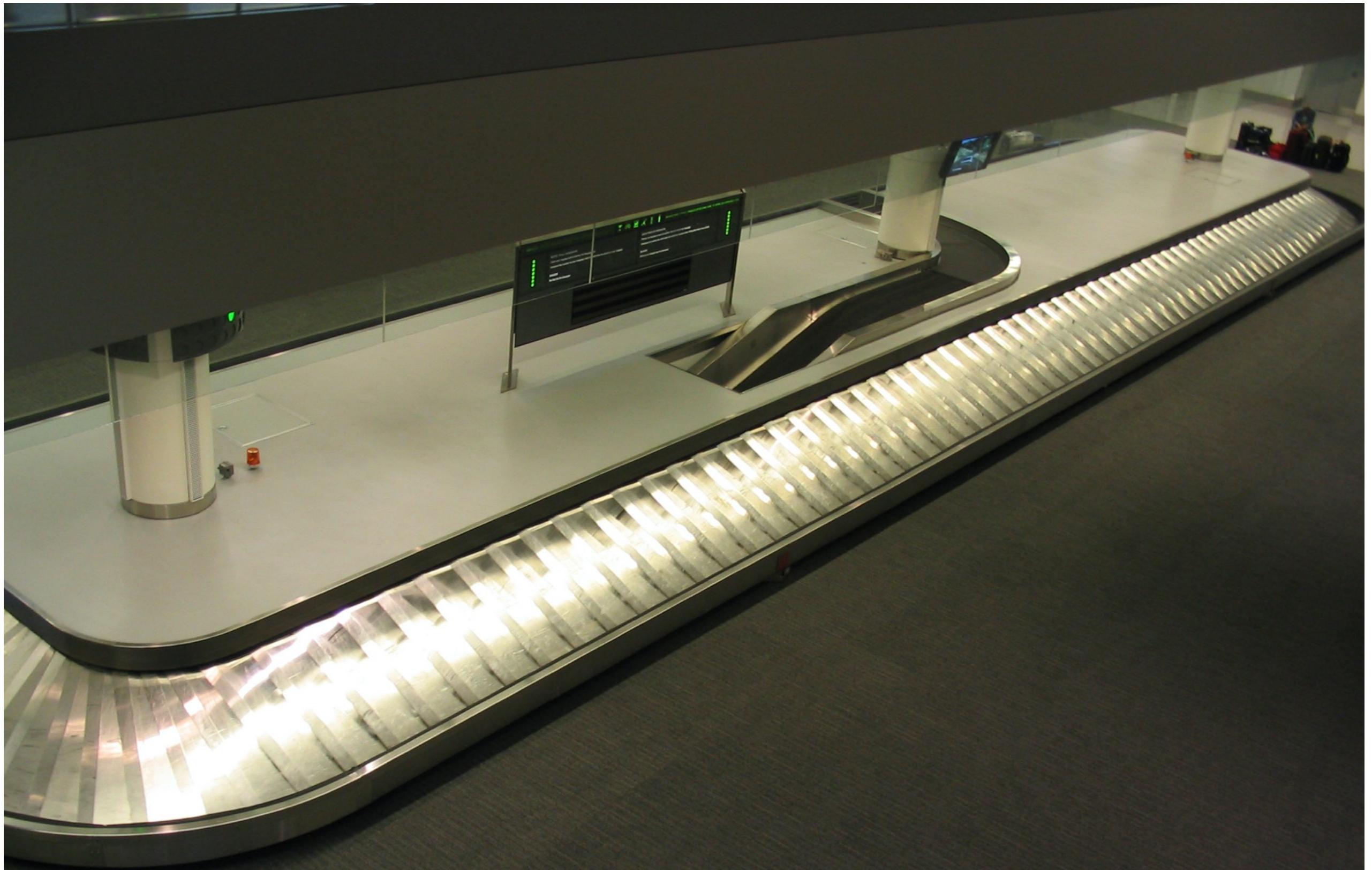


Photo by Gloom

We already know how people
behave and we already
solved these problems before

Start Loading

Initial Content

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut tristique egestas ornare. Cras et quam sed risus fringilla tempus. Aenean in mauris enim. Cras et tempus est. Donec eros dui, commodo vel volutpat non, tempus et mi. Nam ac rutrum mi. Vestibulum luctus mauris eu pulvinar dapibus.

Aliquam viverra nunc non augue cursus faucibus. Donec mollis, dui at imperdiet varius, tellus tellus ultricies leo, sed consectetur nisl eros non orci. Ut augue lacus, placerat eget posuere ut, tincidunt ut dui.

Aenean in mauris enim. Cras et tempus est. Donec eros dui.

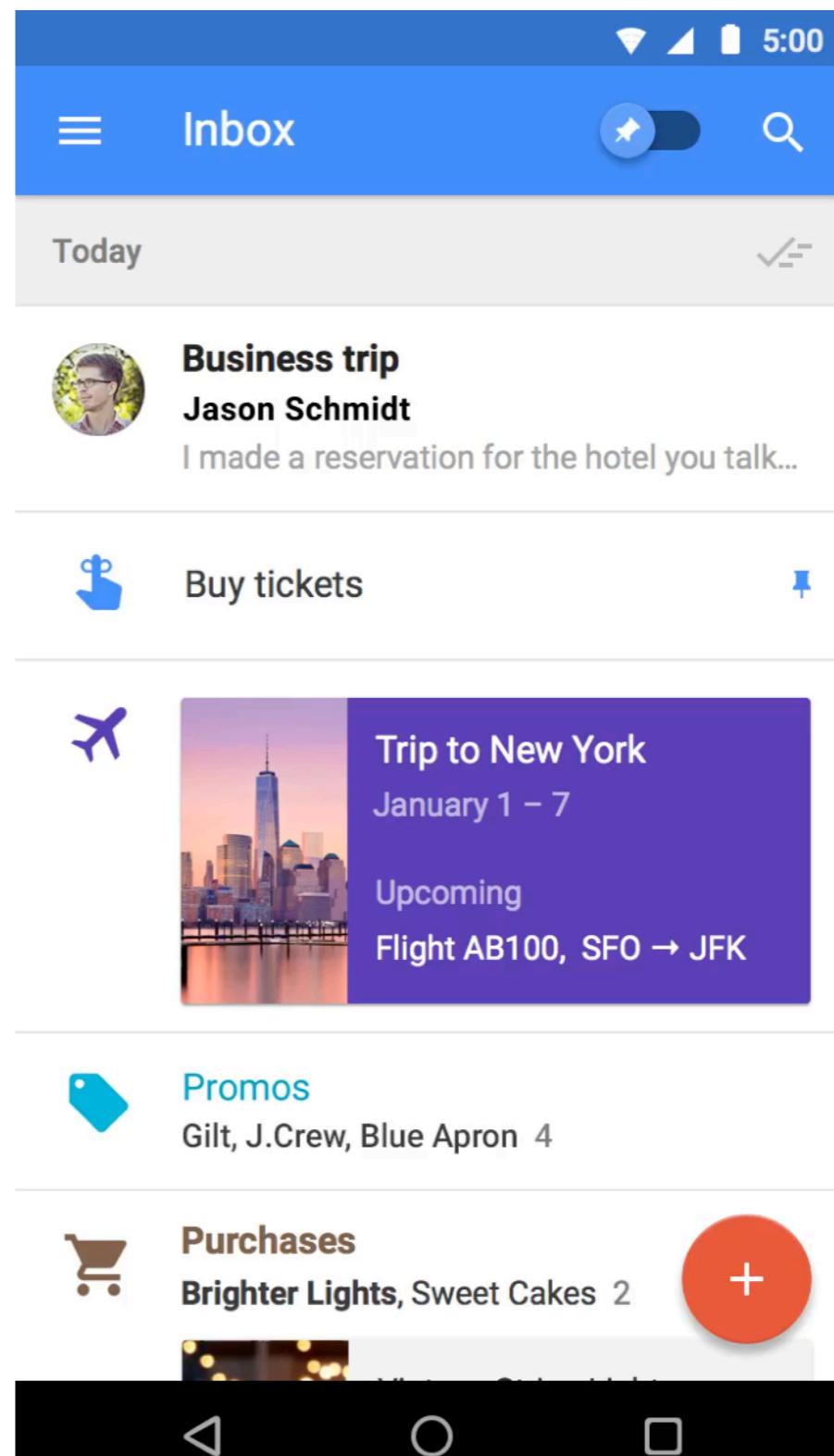
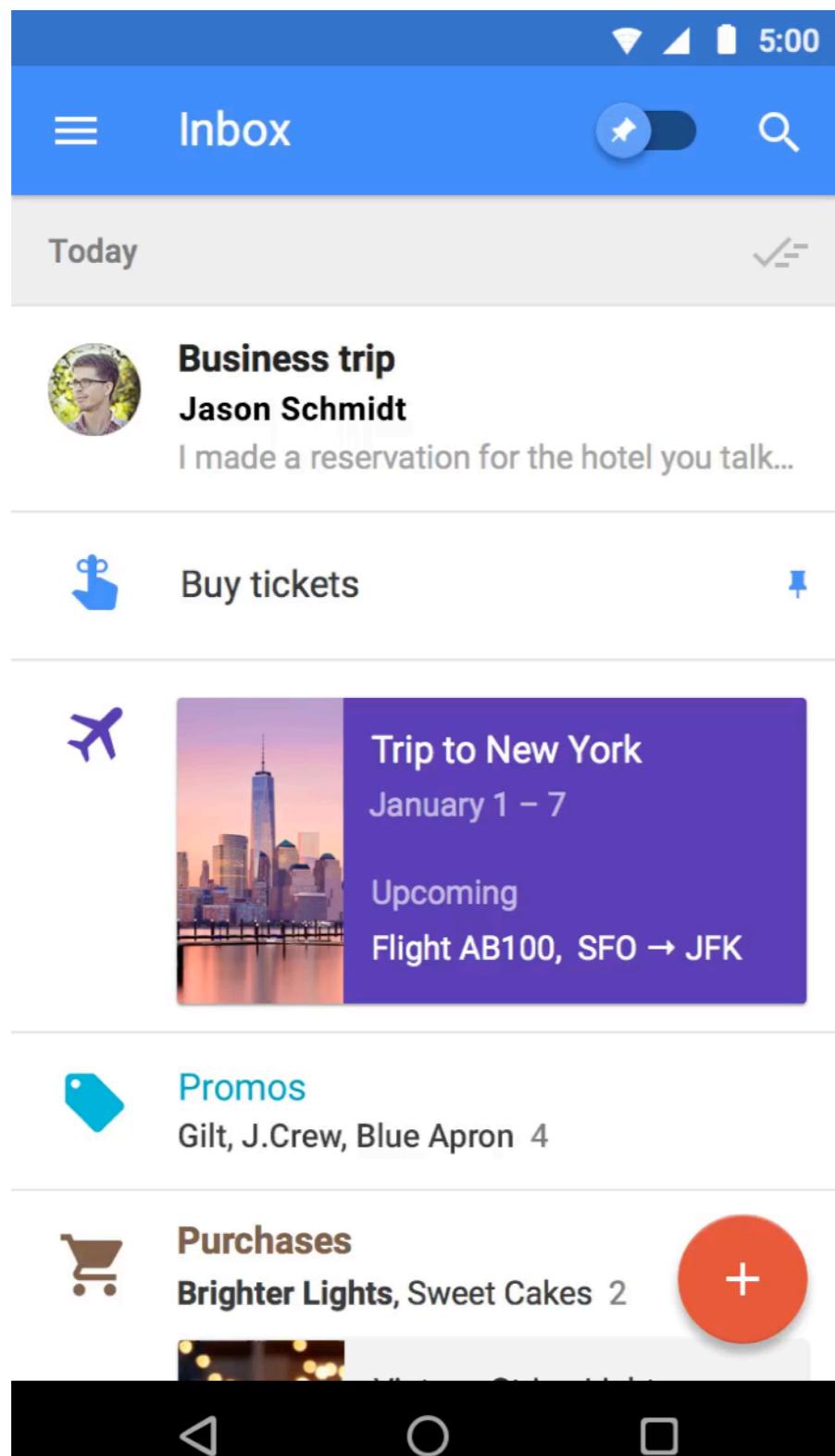
Initial Content

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut tristique egestas ornare. Cras et quam sed risus fringilla tempus. Aenean in mauris enim. Cras et tempus est. Donec eros dui, commodo vel volutpat non, tempus et mi. Nam ac rutrum mi. Vestibulum luctus mauris eu pulvinar dapibus.

Aliquam viverra nunc non augue cursus faucibus. Donec mollis, dui at imperdiet varius, tellus tellus ultricies leo, sed consectetur nisl eros non orci. Ut augue lacus, placerat eget posuere ut, tincidunt ut dui.

Aenean in mauris enim. Cras et tempus est. Donec eros dui.

An introduction to perceived performance



Motion material

HOW FAST IS FAST ENOUGH?

100ms = Instantaneous

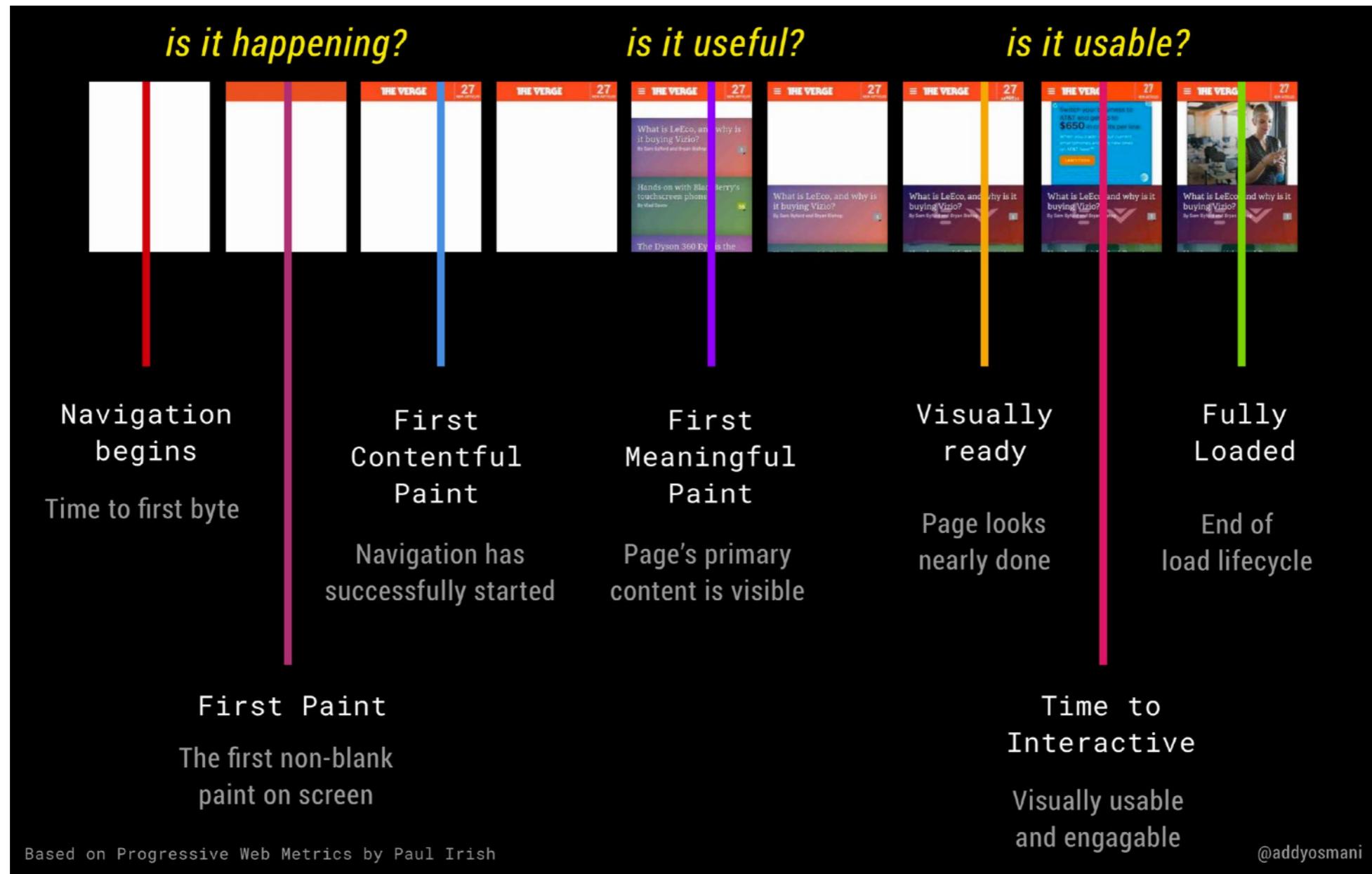
1 second = Uninterrupted flow

[Response Times: The 3 Important Limits -Jakob Nielsen](#)

CRITICAL RENDERING PATH

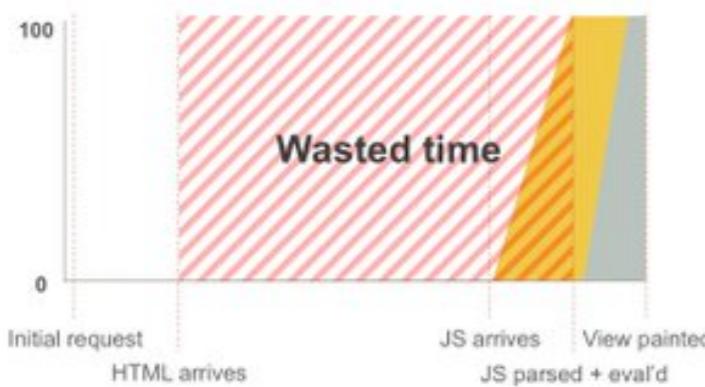


RENDERING MOMENTS



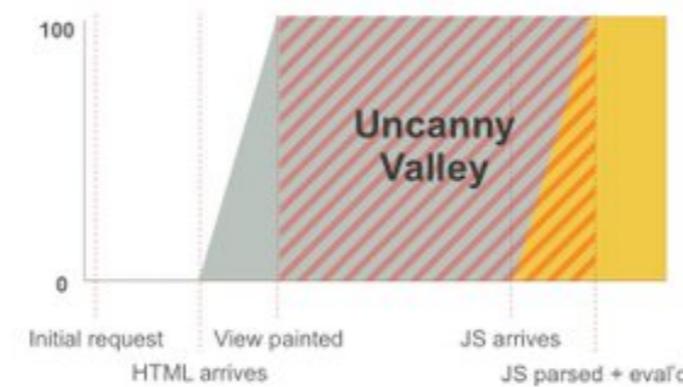
[@addyosmani, Aug 2, 2016](#)

PROGRESSIVE RENDERING



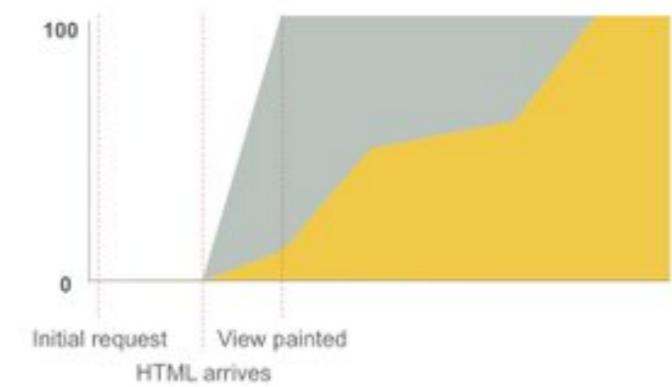
JS-based Render

In a JavaScript-based render you are reliant on all the script to be downloaded, parsed, and evaluated before you are able to render the page. This ends up with a lot of wasted time from when the HTML arrives to when you give the user something meaningful.



Server render + "Hydrate" / Fastboot

With server render + "hydration" or "fastboot", you send a view to the user, but you're still reliant on the JavaScript before functionality is available. This can result in an "uncanny valley" where the app looks interactive, but isn't.

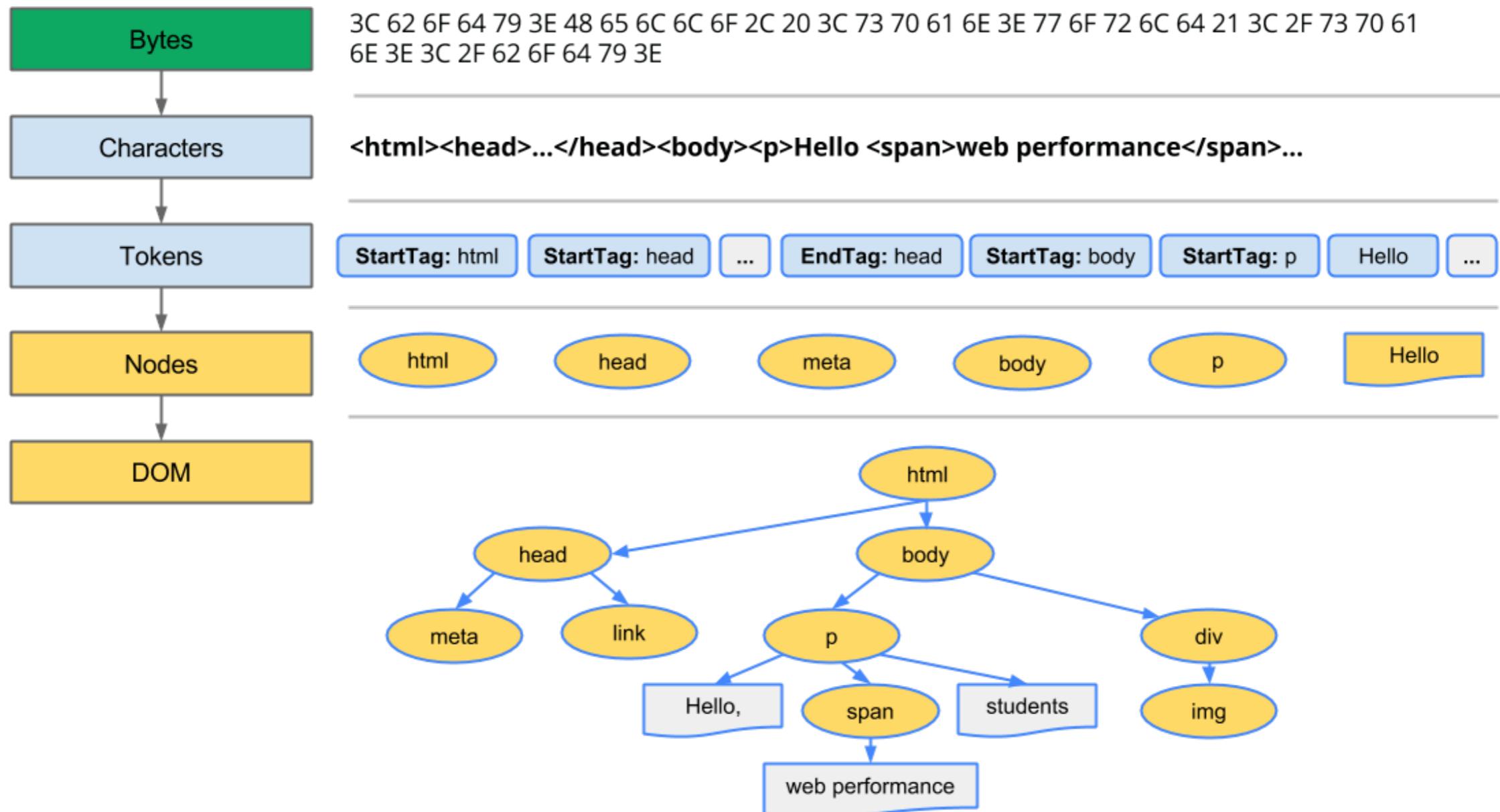


Progressive render + bootstrap

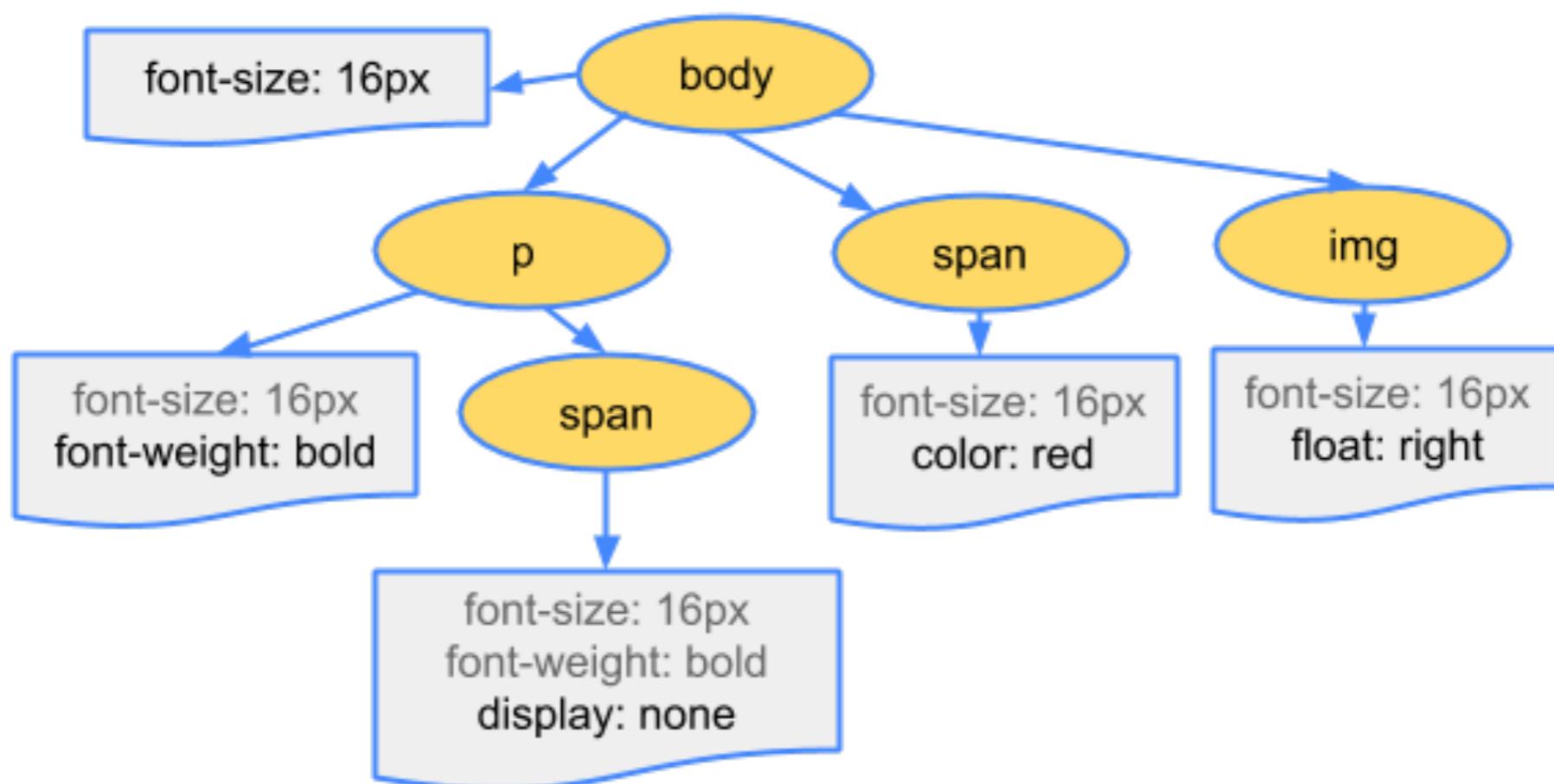
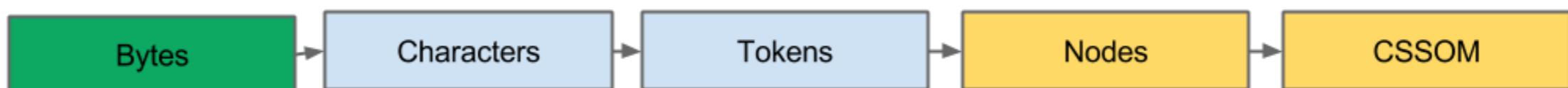
Progressive rendering and bootstrapping means you send a functionally viable (though minimal) view in the HTML, including JavaScript and CSS. As more resources arrive the app progressively "unlocks" features.

The sequence of steps the
browser goes through to
render a page

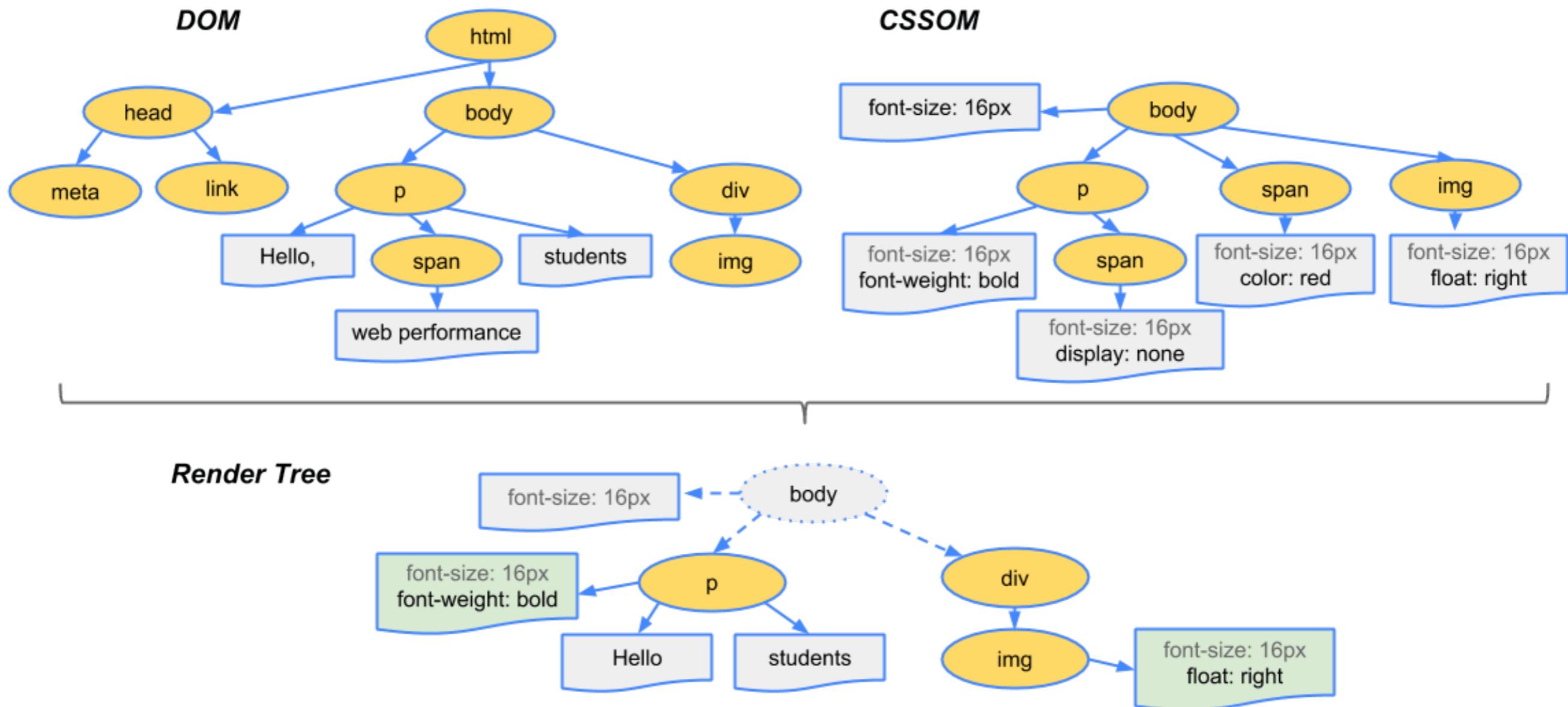
DOM



CSSOM



RENDER TREE CONSTRUCTION



RENDER TREE CONSTRUCTION

- Process HTML markup and build the DOM tree.
- Process CSS markup and build the CSSOM tree.
- Combine the DOM and CSSOM into a render tree.
- Run layout on the render tree to compute geometry of each node.
- Paint the individual nodes to the screen.

Optimizing the critical rendering path is
the process of minimizing the total
amount of time spent performing these
steps

We distinguish between first
view and repeat view

First view is about optimising
the first meaningful paint

Repeat view is about
squashing bytes and caching
strategies

Time to first byte(TTFB)

How fast does the server respond

First meaningful paint(FMP)

How fast is the primary content
available

Time to interactive(TTI)

How long does it take for a page to be
interactive

TOOLING



CHROME DEV TOOLS

Aww yeah, Bootstrap 4 is coming!

Bootstrap

B

Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web.

Download Bootstrap

Currently v3.3.7

Designed for everyone,

localhost:3003

Elements Console Sources Network Timeline Profiles Application Security Audits PageSpeed AdBlock

1.72s 1.85s 1.92s 2.13s 2.98s 3.00s 3.14s 3.22s 4.14s 4.19s 4.30s

Preserve log Disable cache Offline Good 3G (40ms, 1.5Mi)

Filter Regex Hide data URLs All XHR JS CSS Img Media Font Doc WS Manifest Other

Name	M...	Status	Protocol	Type	Initiator	Size	Time	Content...	Timeline – Start Time	3.00s	4.00s	5.00s
localhost	GET	200 OK	http/1.1	doc...	Other	9.4KB 9.1KB	91ms 45ms					
bootstrap.css /dist/css	GET	200 OK	http/1.1	style...	(inde... Parser	144KB 143KB	1.56s 43ms					
docs.css /assets/css/src	GET	200 OK	http/1.1	style...	(inde... Parser	30.6KB 30.3KB	410ms 51ms					
sass-less.png /assets/img	GET	200 OK	http/1.1	png	(inde... Parser	14.4KB 14.2KB	450ms 67ms					
devices.png /assets/img	GET	200 OK	http/1.1	png	(inde... Parser	6.7KB 6.5KB	244ms 75ms					
components.png /assets/img	GET	200 OK	http/1.1	png	(inde... Parser	3.1KB 2.9KB	167ms 82ms					
expo-lyft.jpg /assets/img	GET	200 OK	http/1.1	jpeg	(inde... Parser	156KB 156KB	4.06s 91ms					
expo-vogue.jpg /assets/img	GET	200 OK	http/1.1	jpeg	(inde... Parser	194KB 194KB	4.48s 84ms					
expo-riot.jpg /assets/img	GET	200 OK	http/1.1	jpeg	(inde... Parser	158KB 158KB	3.97s 84ms					
expo-newweek.jpg /assets/img	GET	200 OK	http/1.1	jpeg	(inde... Parser	197KB 197KB	3.52s 83ms					

18 requests | 1.2MB transferred | Finish: 6.62s | DOMContentLoaded: 3.13s | Load: 6.62s

CHROME AUDITS

The image shows a split-screen view. On the left is the official Bootstrap website homepage. At the top, a blue header bar displays the text "Aww yeah, Bootstrap 4 is coming!". Below this, the main content area has a purple background. It features a large white letter "B" inside a rounded square box. Below the letter, the text reads: "Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web." A prominent button labeled "Download Bootstrap" is centered below the text. At the bottom of the page, it says "Currently v3.3.7". On the right side of the image is a screenshot of the Chrome DevTools Audits tab. The tab bar at the top includes "Elements", "Console", "Sources", "Network", "Performance", "Memory", "Application", "Security", "Audits" (which is selected), and "Page". Below the tab bar, the URL "localhost:6521 19/10/2017, 16:45:46" is shown. The "Audits" section is divided into four categories: "Progressive Web App" (score 45), "Performance" (score 70), "Accessibility" (score 100), and "Best Practices" (score 77). The "Progressive Web App" section contains a summary of 6 failed audits, each with a red "X" icon. The "Performance" section includes a "Metrics" chart showing various performance metrics over time, with a red line highlighting the "First meaningful paint" at 4,710 ms. The "Best Practices" section lists opportunities for optimization, with a green "100" score.

Aww yeah, Bootstrap 4 is coming!

Bootstrap

B

Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web.

Download Bootstrap

Currently v3.3.7

Designed for everyone, everywhere.

Bootstrap makes front-end web development faster and easier. It's made for folks of all skill levels, devices of all shapes, and projects of all sizes.

Sass {less}

45

70

100

77

Progressive Web App

These audits validate the aspects of a Progressive Web App, as specified by the baseline [PWA Checklist](#).

45

6 failed audits

- Does not register a Service Worker
- Does not respond with a 200 when offline
- Does not redirect HTTP traffic to HTTPS
- User will not be prompted to Install the Web App
 - Failures: No manifest was fetched, Site does not register a Service Worker, Manifest start_url is not cached by a Service Worker.
- Is not configured for a custom splash screen
 - Failures: No manifest was fetched.
- Address bar does not match brand colors
 - Failures: No manifest was fetched, No <meta name="theme-color"> tag found.

5 Passed Audits

Manual checks to verify

Performance

These encapsulate your app's performance.

70

Metrics

These metrics encapsulate your app's performance across a number of dimensions.

471 ms 942 ms 1.4 s 1.9 s 2.4 s 2.8 s 3.3 s 3.8 s 4.2 s 4.7 s

First meaningful paint 4,710 ms

First Interactive (beta) 4,710 ms

Consistently Interactive (beta) 4,710 ms

Perceptual Speed Index: 3,277 (target: < 1,250)

Estimated Input Latency: 16 ms (target: < 50 ms)

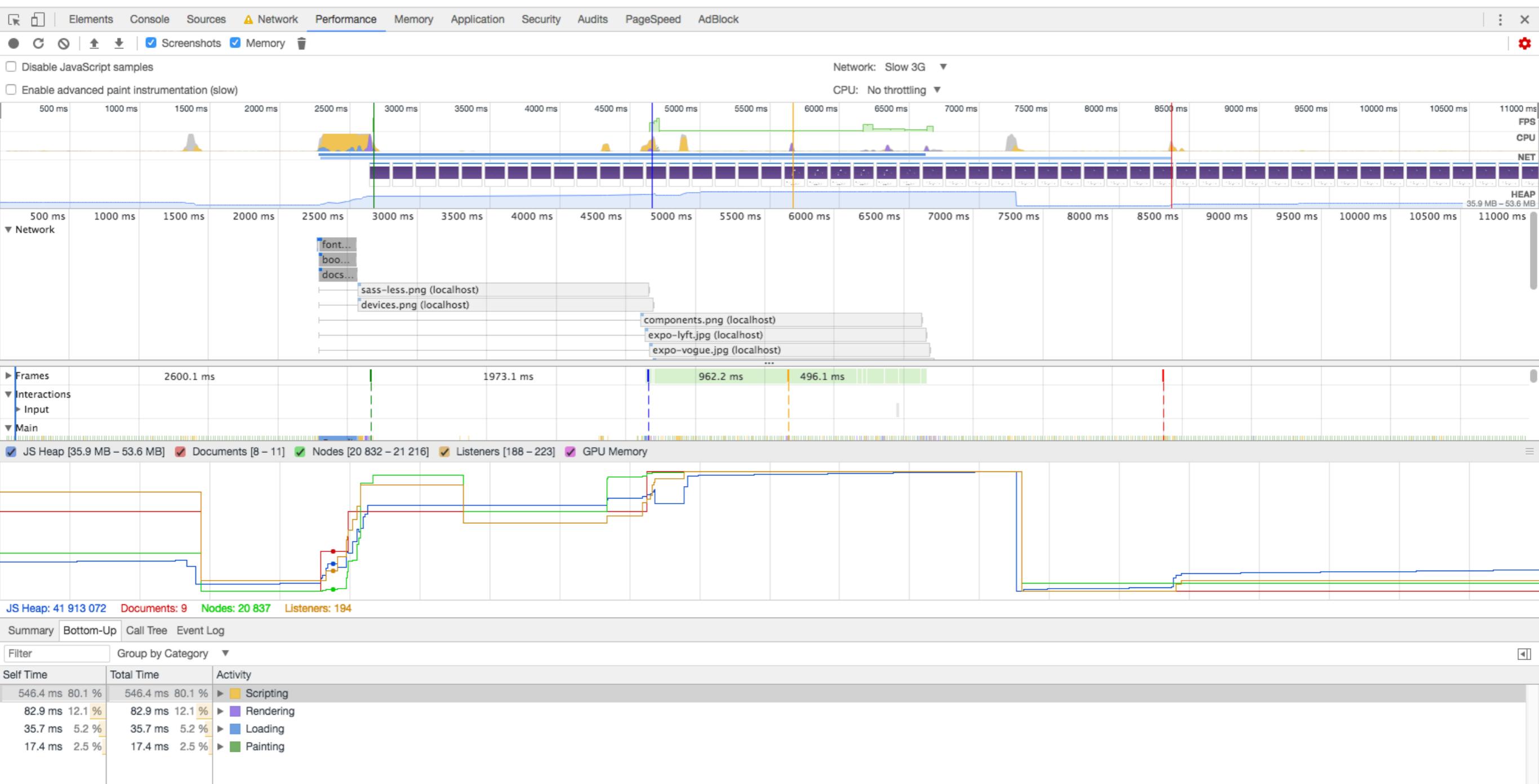
77

100

Opportunities

These are opportunities to speed up your application by optimizing the following resources.

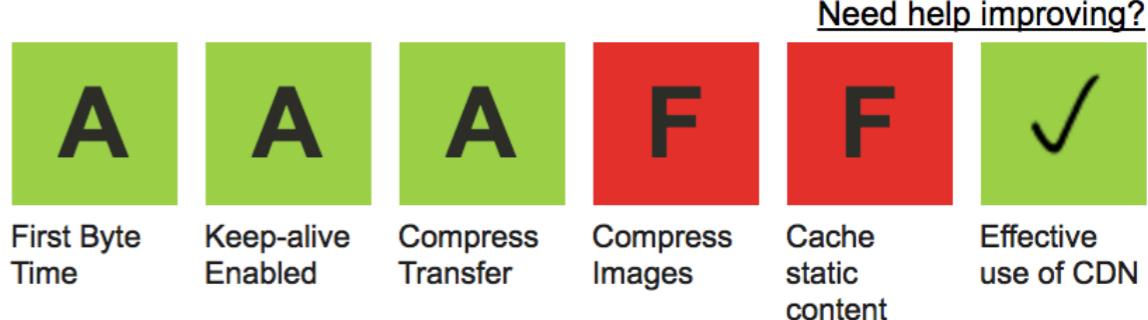
CHROME PERFORMANCE



WEBPAGETEST

Web Page Performance Test for getbootstrap.com

From: Amsterdam, NL - IISpeed - Chrome - Cable
26/01/2017, 21:22:40

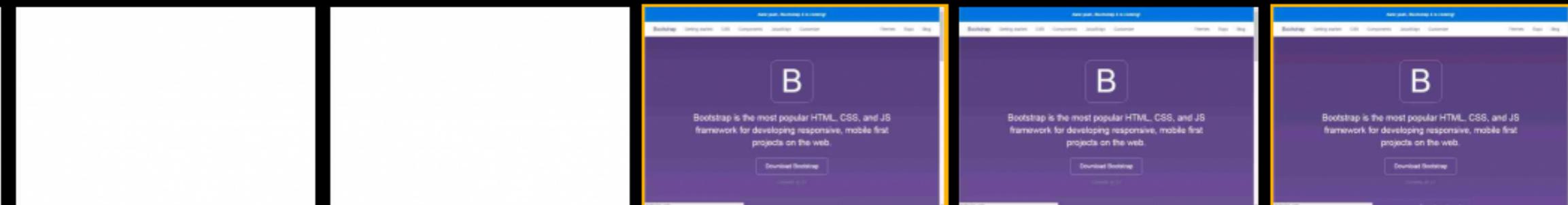


Summary	Details	Performance Review	Content Breakdown	Domains	Processing Breakdown	Screen Shot	
Tester: VPS16230-93.191.133.233							Raw page data - Raw object data
Test runs: 5							Export HTTP Archive (.har)
Re-run the test							View Test Log

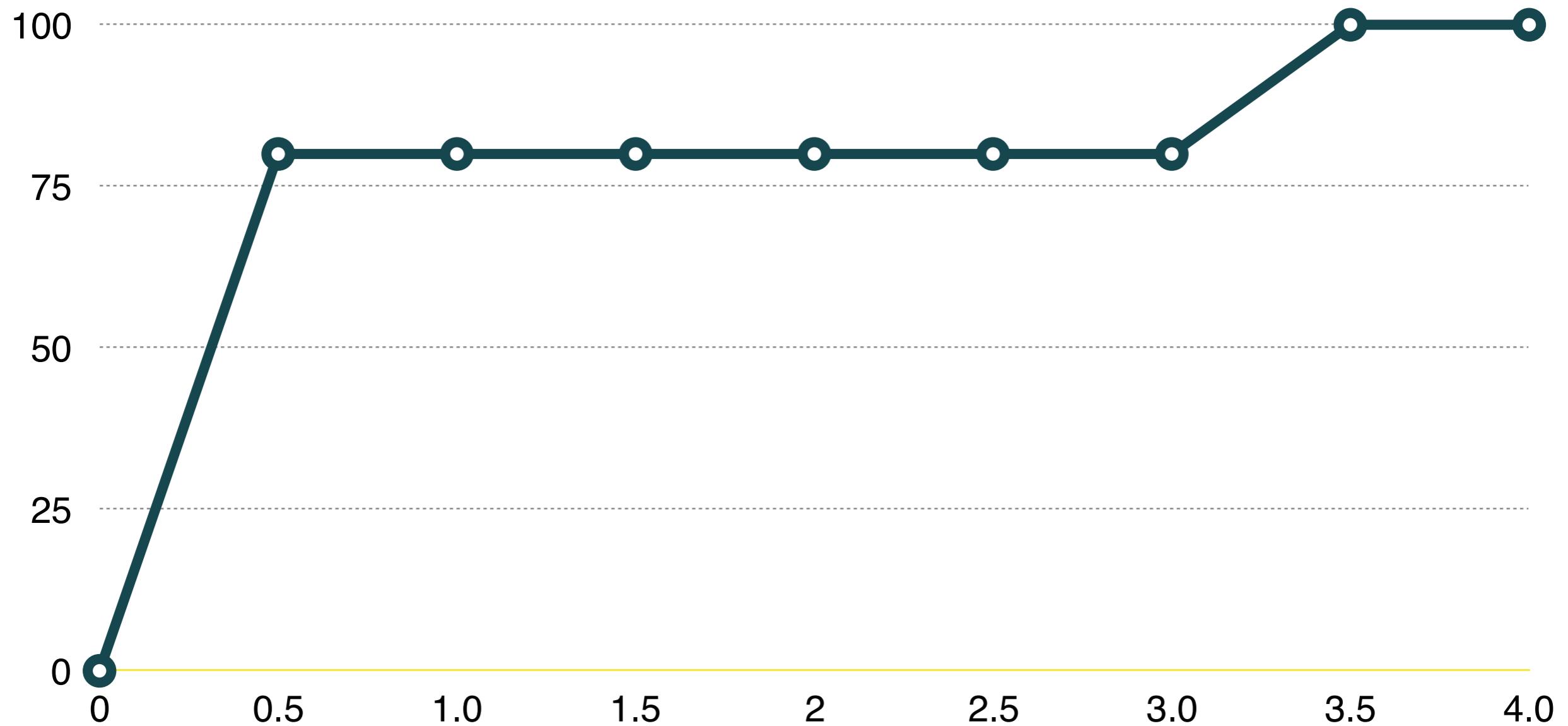
Performance Results (Median Run)

	Load Time	First Byte	Start Render	Speed Index	DOM Elements	Document Complete			Fully Loaded				
						Time	Requests	Bytes In	Time	Requests	Bytes In	Certificates	Cost
First View (Run 3)	2.950s	0.452s	0.812s	866	155	2.950s	25	884 KB	3.013s	26	886 KB	16 KB	\$---
Repeat View (Run 4)	0.837s	0.572s	-	0	153	0.837s	4	10 KB	0.837s	4	10 KB	3 KB	

2.0s 2.5s 3.0s 3.5s 4.0s 4.5s



SPEED INDEX



Latest SRP

PAINT
1.6 Sec.

USABLE PAGE
2.2 Seconds



Min + gzip

PAINT
1.6 Sec.

USABLE PAGE
2.0 Seconds



CriticalCSS + min + gzip

PAINT
0.6 Sec.

USABLE PAGE
1.3 Seconds



Critical + FOIT + min + gzip

PAINT and USABLE PAGE
0.7 Sec.

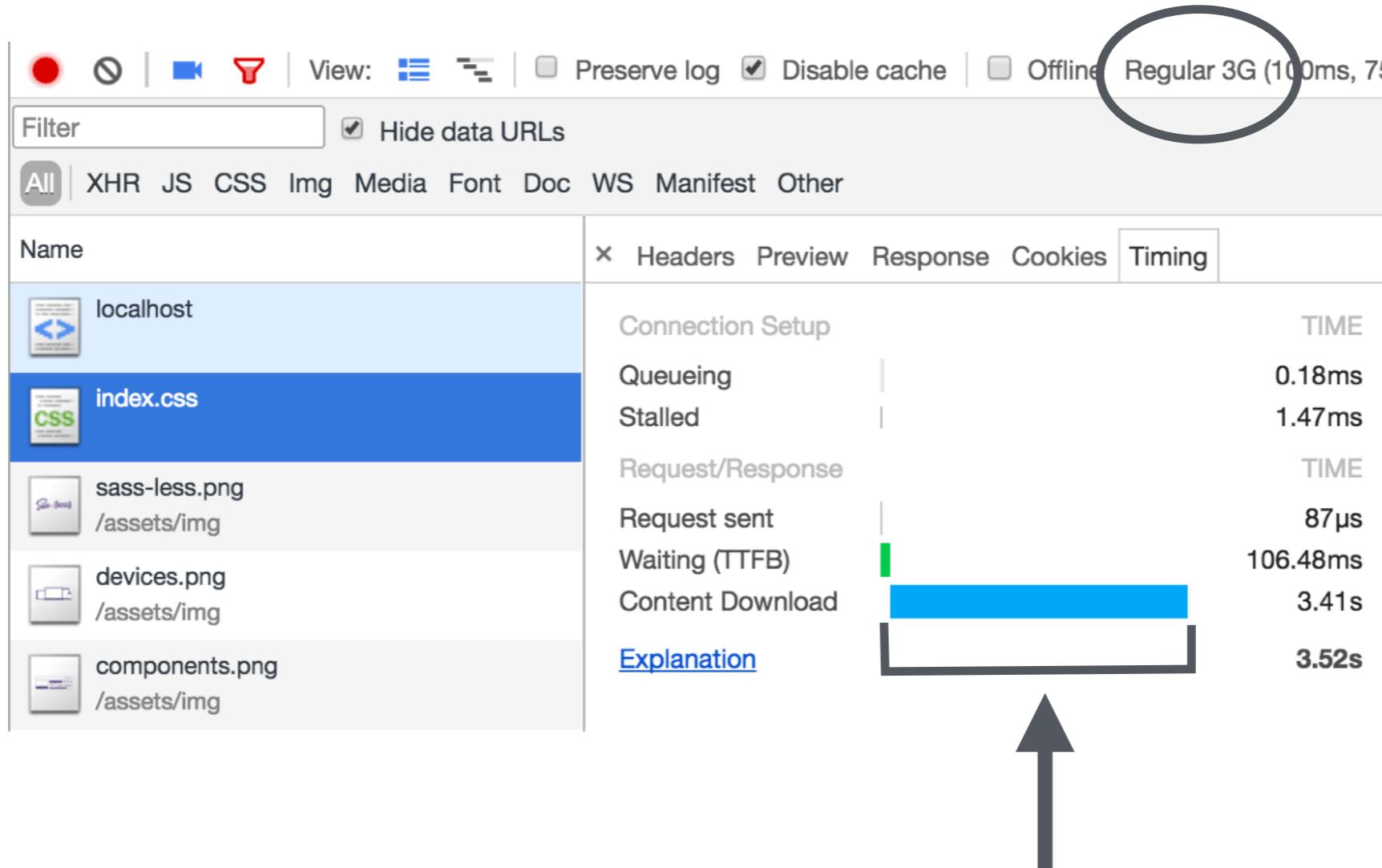




MINIFY HTML, CSS, JS



PROBLEM



download time = file size / download speed

Minification

removing all unnecessary characters from
source code
without changing its functionality.

HTML BEFORE

```
<html>
<head>
  <style>
    #myContent { font-family: Arial; }
    #myContent { font-size: 90%; }
  </style>
</head>

<body>
<!-- start myContent -->
<div id="myContent">
  <p>Hello world!</p>
</div>
<!-- end myContent -->
</body>
</html>
```

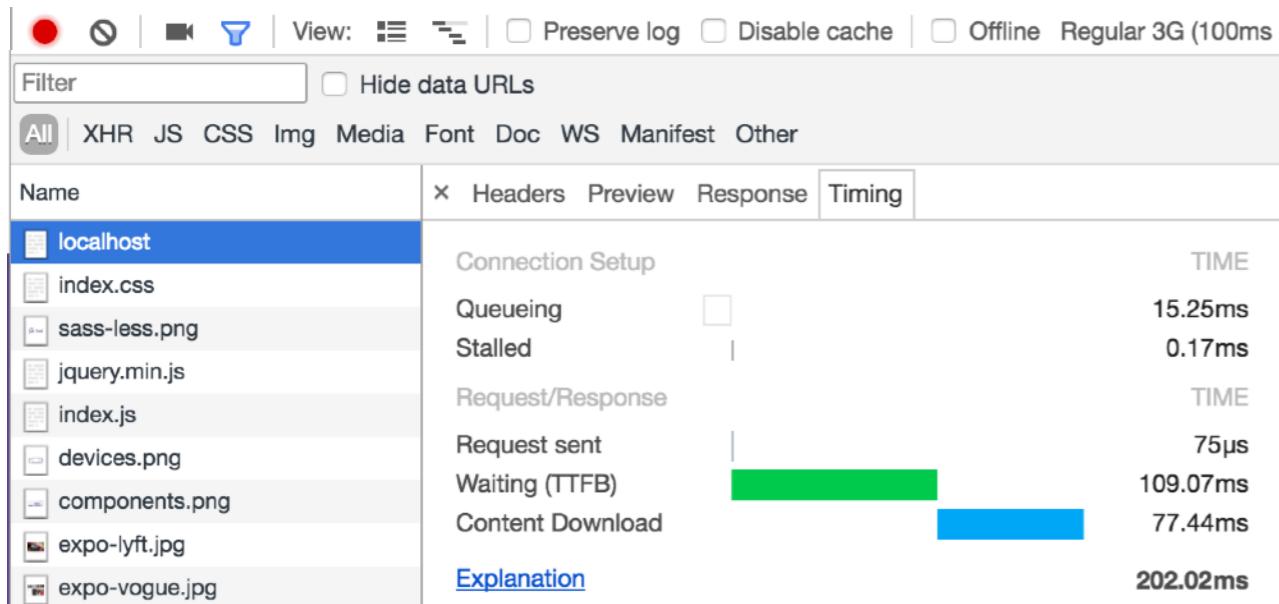
www.maxcdn.com/one/visual-glossary/minification/

HTML AFTER

```
<style>#myContent{font-family:Arial;font-size:90%}</style><div id=myContent><p>Hello world!</p></div>
```

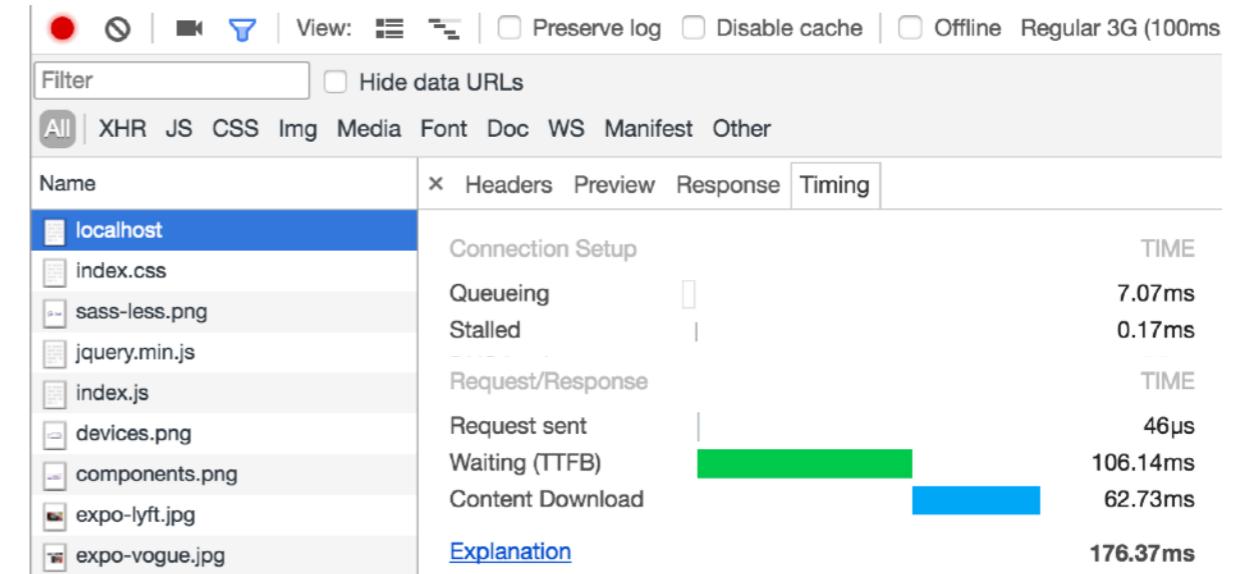
COMPARE: HTML

BEFORE



8.3KB ~0.20 sec

AFTER



6.5KB ~0.17 sec

JS BEFORE

```
var x = 40 + 2;

function foo(left, right) {
    return left + right;
}
```

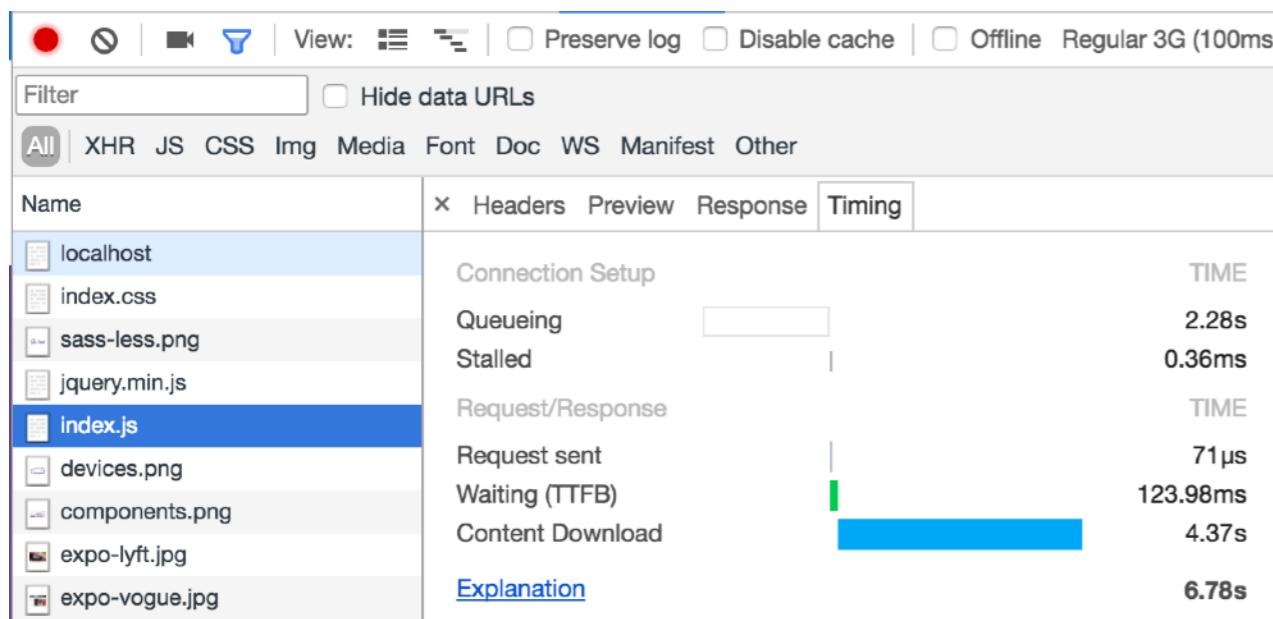
JS AFTER

```
function foo(n,o){return n+o}var x=42;
```

demos.forbeslindesay.co.uk/uglify-js
alistapart.com/article/better-javascript-minification

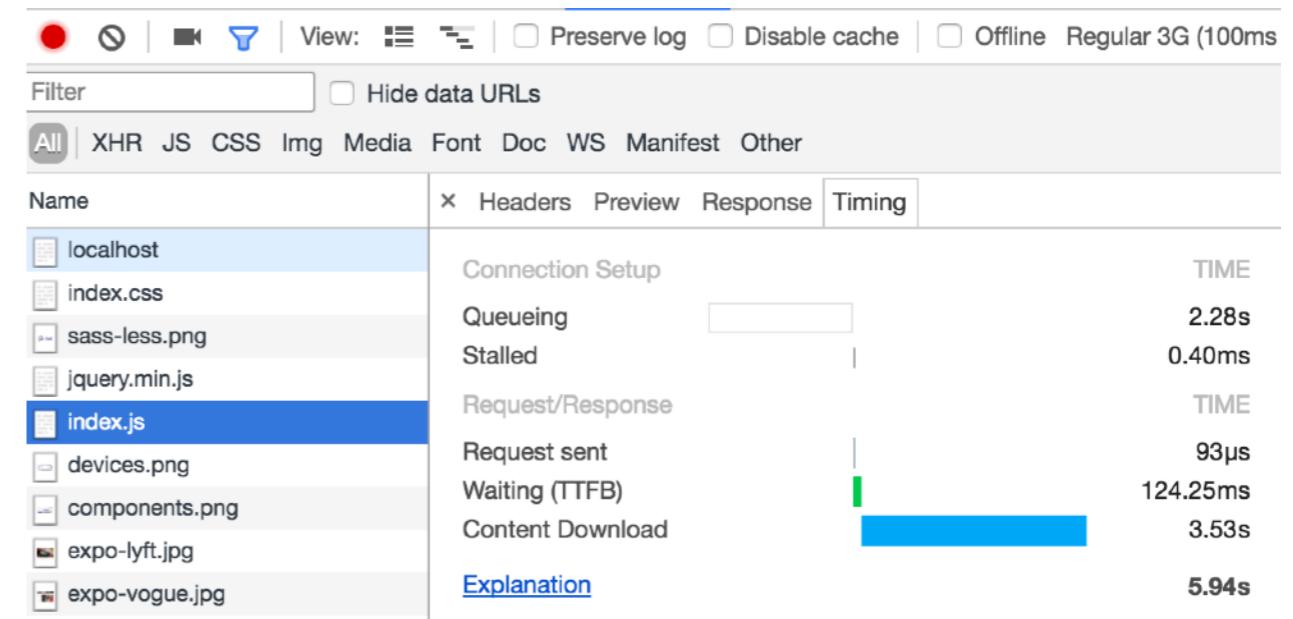
COMPARE: JS

BEFORE



115KB ~6.8 sec

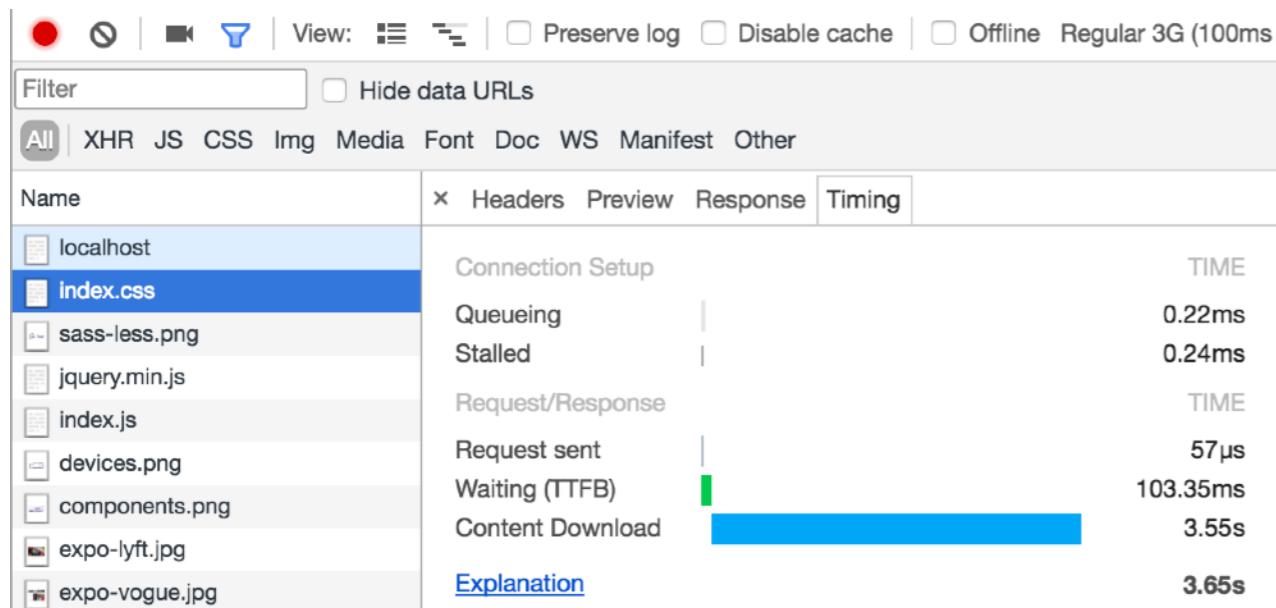
AFTER



81.8KB ~5.9 sec

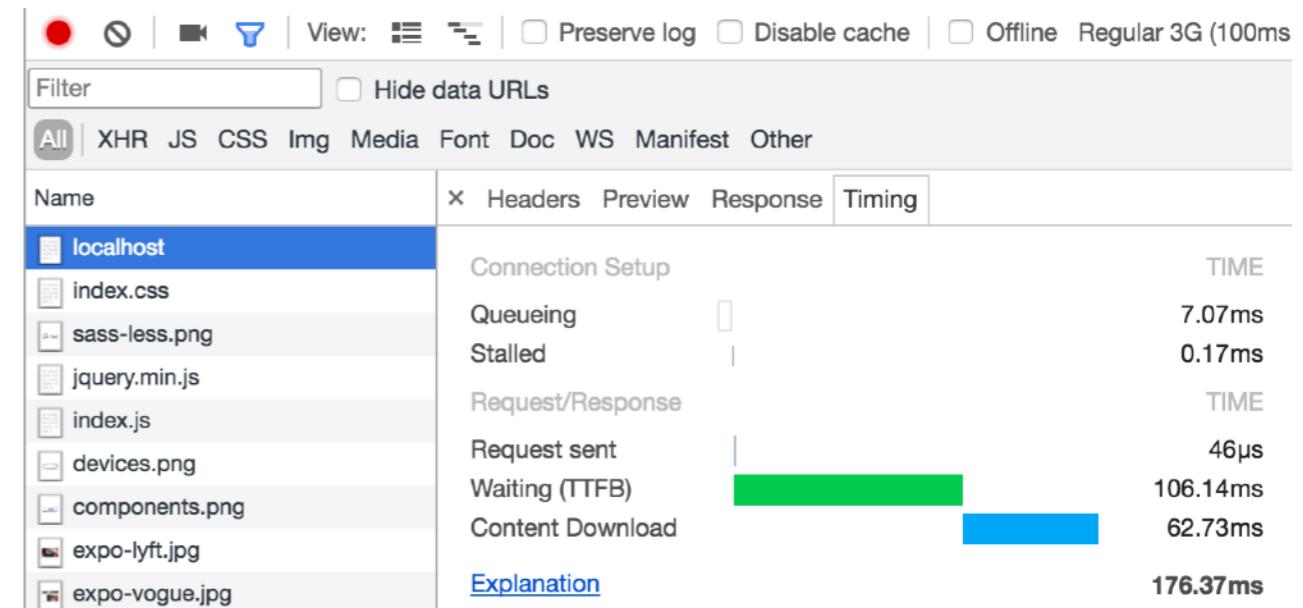
COMPARE: CSS

BEFORE



166KB ~3.6 sec

AFTER

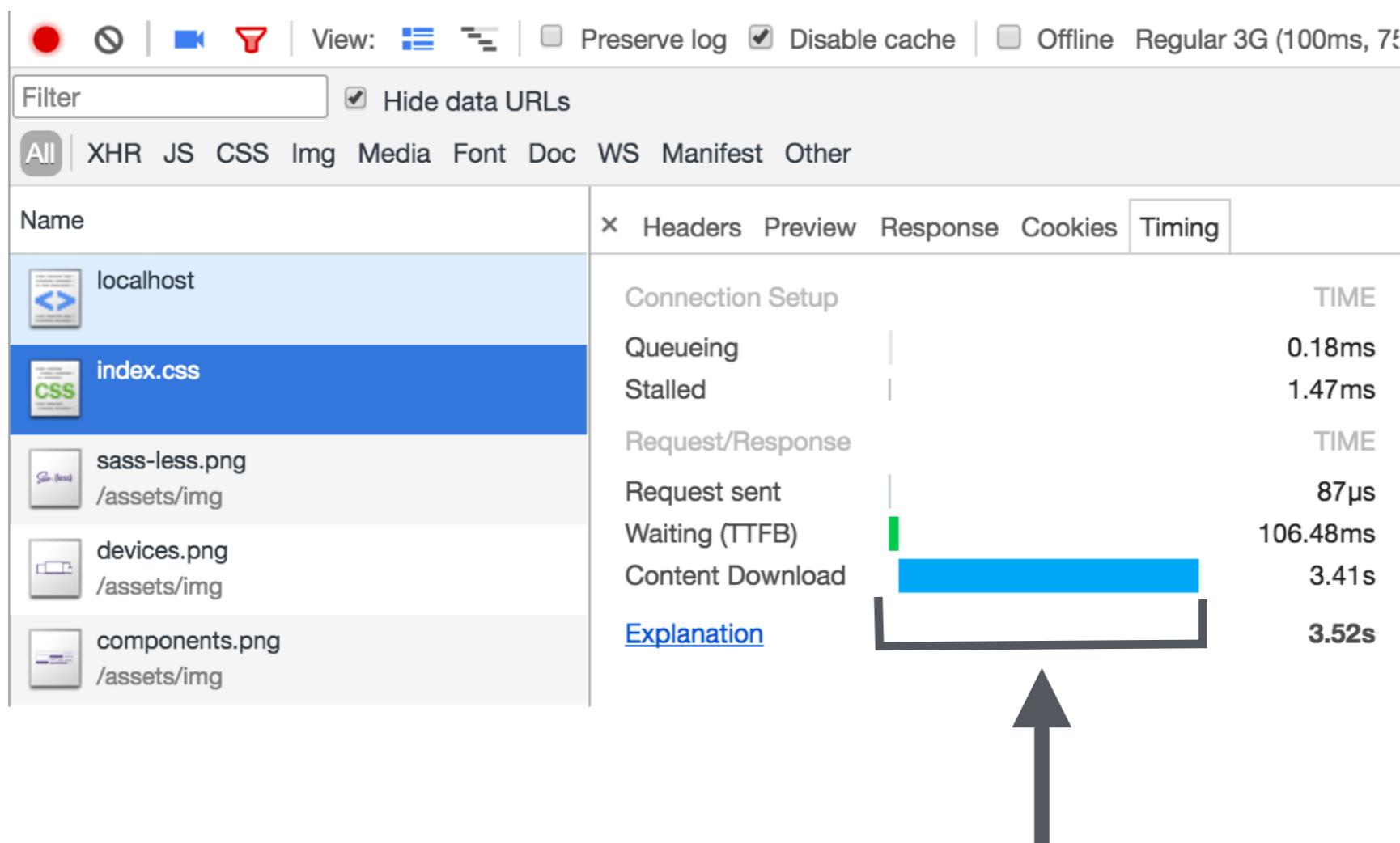


132KB ~2.9 sec

HTTP CACHING



PROBLEM: NO CACHE



connect, wait and download

Page

Hey, I need "/script-v1.js", "/styles-v1.css" and "/cats-v1.jpg" 10:24

Server

Sure, here you go. 10:25

Page

Cheers! 10:25

adapted from
[jakearchibald.com/2016/
caching-best-practices/](http://jakearchibald.com/2016/caching-best-practices/)

Page

Hey, I need "/script-v1.js", "/styles-v1.css" and "/cats-v1.jpg" 10:24

Server

Sure, here you go. 10:25

Page

Cheers! 10:25

THE NEXT DAY

Page

Hey, I need "/script-v1.js", "/styles-v1.css" and "/cats-v1.jpg" 10:24

Server

Sure, here you go.

Page

Cheers! 10:25

adapted from
[jakearchibald.com/2016/
caching-best-practices/](http://jakearchibald.com/2016/caching-best-practices/)

Page

Hey, I need "/script-v1.js", "/styles-v1.css" and "/cats-v1.jpg" 10:24

Cache

I got nuthin', how about you Server? 10:24

Server

Sure, here you go. Btw Cache: these are fine to use for like, a year. 10:25

Cache

Thanks! 10:25

Page

Cheers! 10:25

[jakearchibald.com/2016/
caching-best-practices/](http://jakearchibald.com/2016/caching-best-practices/)

Page

Hey, I need "/script-v1.js", "/styles-v1.css" and "/cats-v1.jpg" 10:24

Cache

I got nuthin', how about you Server? 10:24

Server

Sure, here you go. Btw Cache: these are fine to use for like, a year. 10:25

Cache

Thanks! 10:25

Page

Cheers! 10:25

THE NEXT DAY

Page

Hey, I need "/script-v2.js", "/styles-v2.css" and "/cats-v1.jpg" 08:14

Cache

I've got the cats one, here you go. I don't have the others. Server? 08:14

Server

Sure, here's the new CSS & JS. Btw Cache: these are also fine to use for a year.

08:15

Cache

Brilliant! 08:15

Page

Thanks! 08:15

LATER

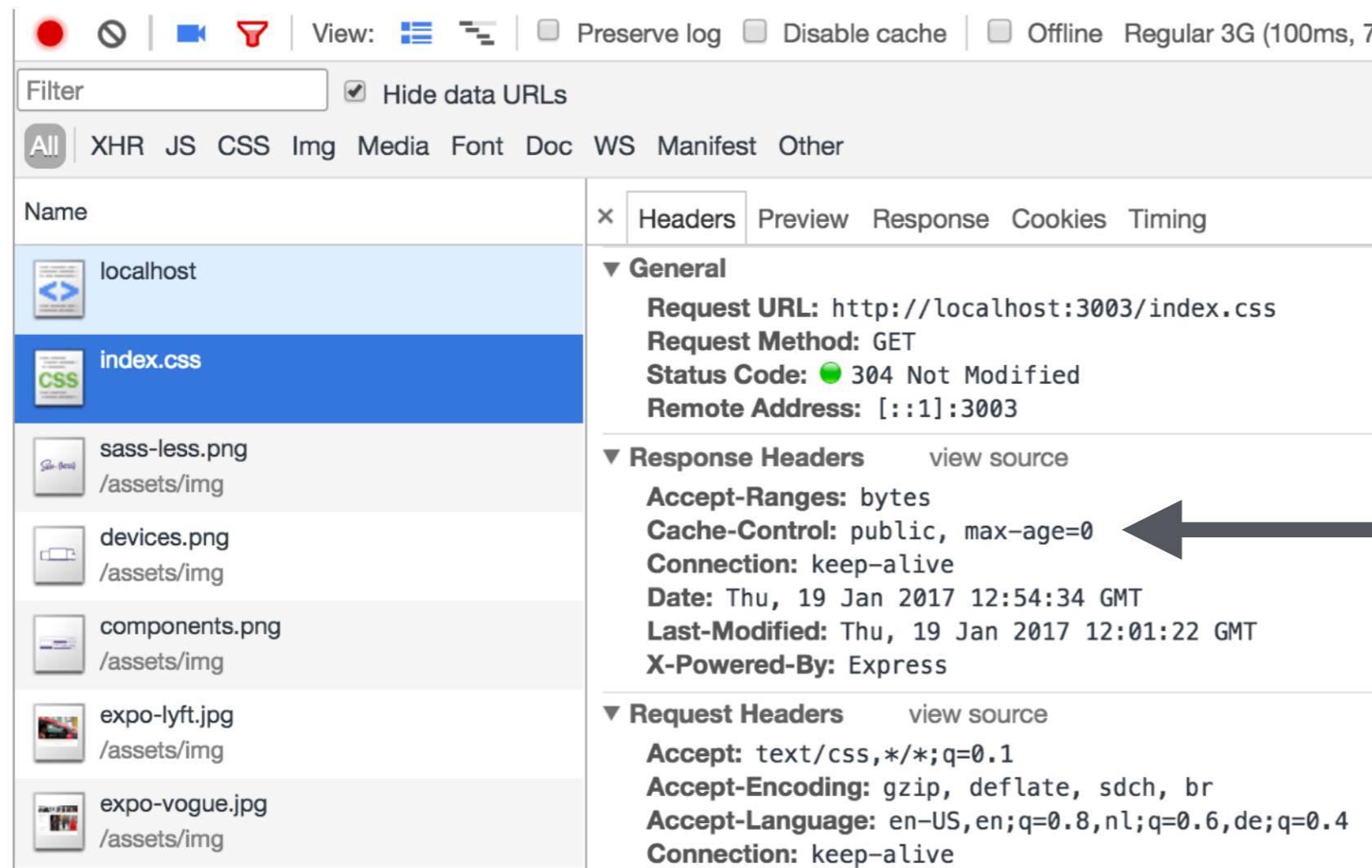
Cache

Hm, I haven't used "/script-v1.js" & "/styles-v1.css" for a while. I'll just delete them.

12:32

[jakearchibald.com/2016/
caching-best-practices/](http://jakearchibald.com/2016/caching-best-practices/)

CACHE CONTROL



Filter Hide data URLs

All XHR JS CSS Img Media Font Doc WS Manifest Other

Name
localhost
index.css
sass-less.png /assets/img
devices.png /assets/img
components.png /assets/img
expo-lyft.jpg /assets/img
expo-vogue.jpg /assets/img

x Headers Preview Response Cookies Timing

▼ General

Request URL: http://localhost:3003/index.css
Request Method: GET
Status Code: 304 Not Modified
Remote Address: [::1]:3003

▼ Response Headers [view source](#)

Accept-Ranges: bytes
Cache-Control: public, max-age=0 ← no caching
Connection: keep-alive
Date: Thu, 19 Jan 2017 12:54:34 GMT
Last-Modified: Thu, 19 Jan 2017 12:01:22 GMT
X-Powered-By: Express

▼ Request Headers [view source](#)

Accept: text/css,*/*;q=0.1
Accept-Encoding: gzip, deflate, sdch, br
Accept-Language: en-US,en;q=0.8,nl;q=0.6,de;q=0.4
Connection: keep-alive

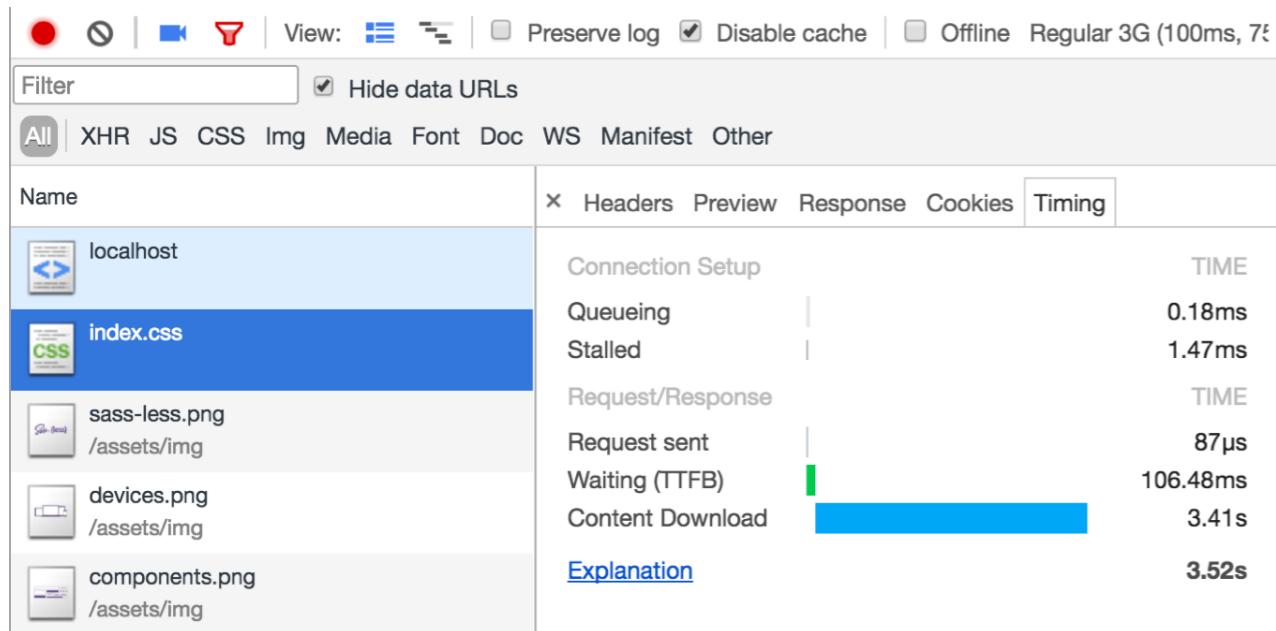
SET CACHE CONTROL

```
app.use((req, res, next) => {
  // todo: set cache header to 1 year
  res.setHeader('Cache-Control', 'max-age=' + 365 *
24 * 60 * 60);
  next();
});
```

Name		Headers	Preview	Response	Cookies	Timing
localhost						
index.css		<p>▼ General</p> <p>Request URL: http://localhost:3003/index.css Request Method: GET Status Code: 200 OK (from memory cache) Remote Address: [::1]:3003</p>				
sass-less.png /assets/img		<p>▼ Response Headers view source</p> <p>Accept-Ranges: bytes Cache-Control: max-age=31536000 Connection: keep-alive</p>				
devices.png /assets/img						

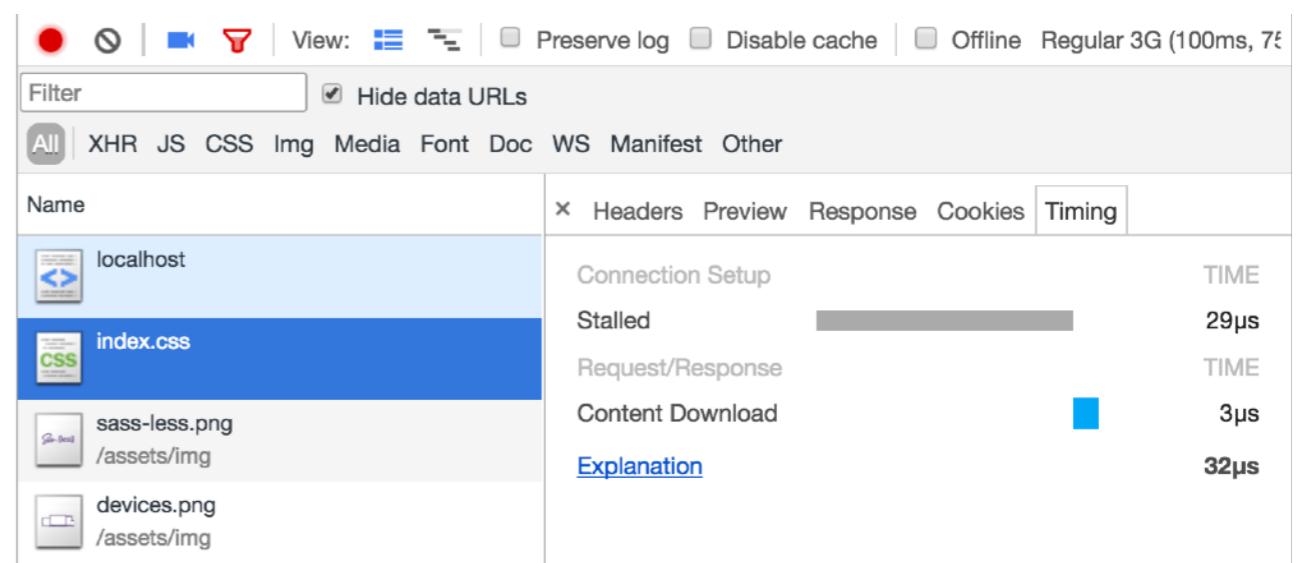
COMPARE

BEFORE



~3.5 sec

AFTER



~ 0.0 sec

REVISIONING



PROBLEM: CACHE INVALIDATION

 index.css	GET	200 OK	stylesheet	(from memory cache)	0ms 0ms
---	-----	--------	------------	---------------------	------------

when a file is
cached by the browser,
file updates are no longer
fetched from our server

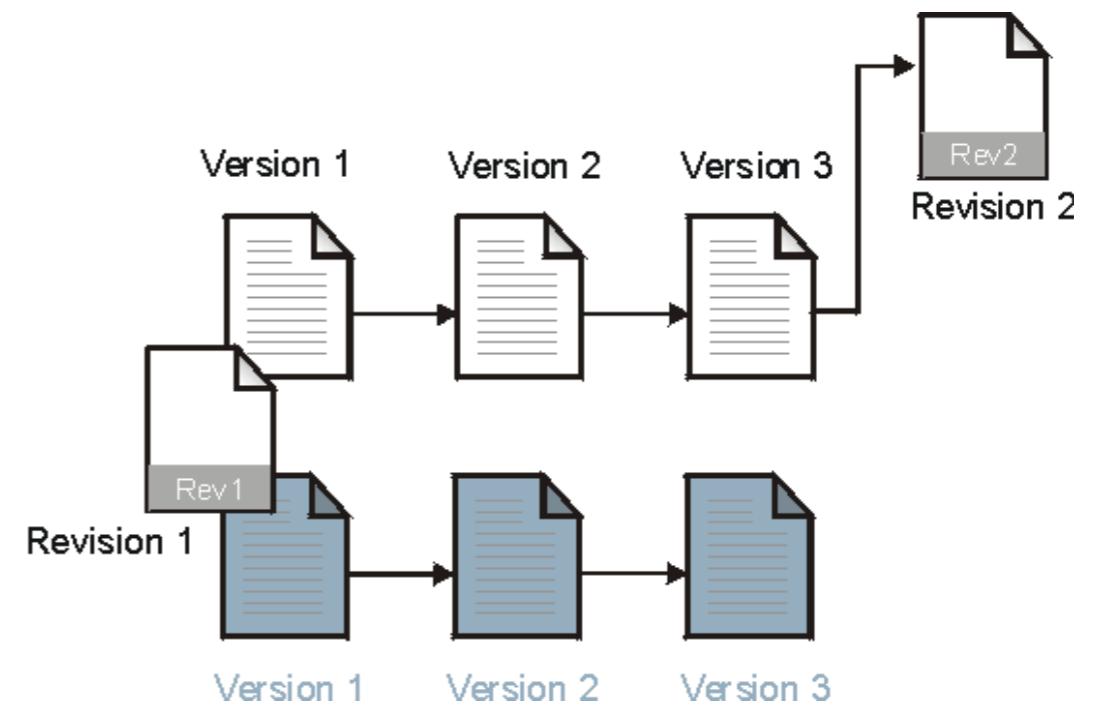
FILE REVISIONING

index.css

index.css?v=2

index-2.css

...



REVISIONED FILES

- cache/
 - 📄 index.html
 - 📄 index.css
 - 📄 index-67760ac4ed.css
 - 📄 index.js
 - 📄 index-40b67b7222.js
 - 📄 ...
 - 📄 rev-manifest.json

USING ETAG

first visit:

- status: 200
- size: 4 688 bytes

Name	Headers	Preview	Response	Timing
a11y-viewer.netlify.com	General Request URL: https://a11y-viewer.netlify.com/ Request Method: GET Status Code: 200 Remote Address: 54.93.54.1:443			
www.voorhoede.nl				
?fmt=1&num=1&cv=8&frm=2				

repeat visit:

- status: 304
- size: 47 bytes

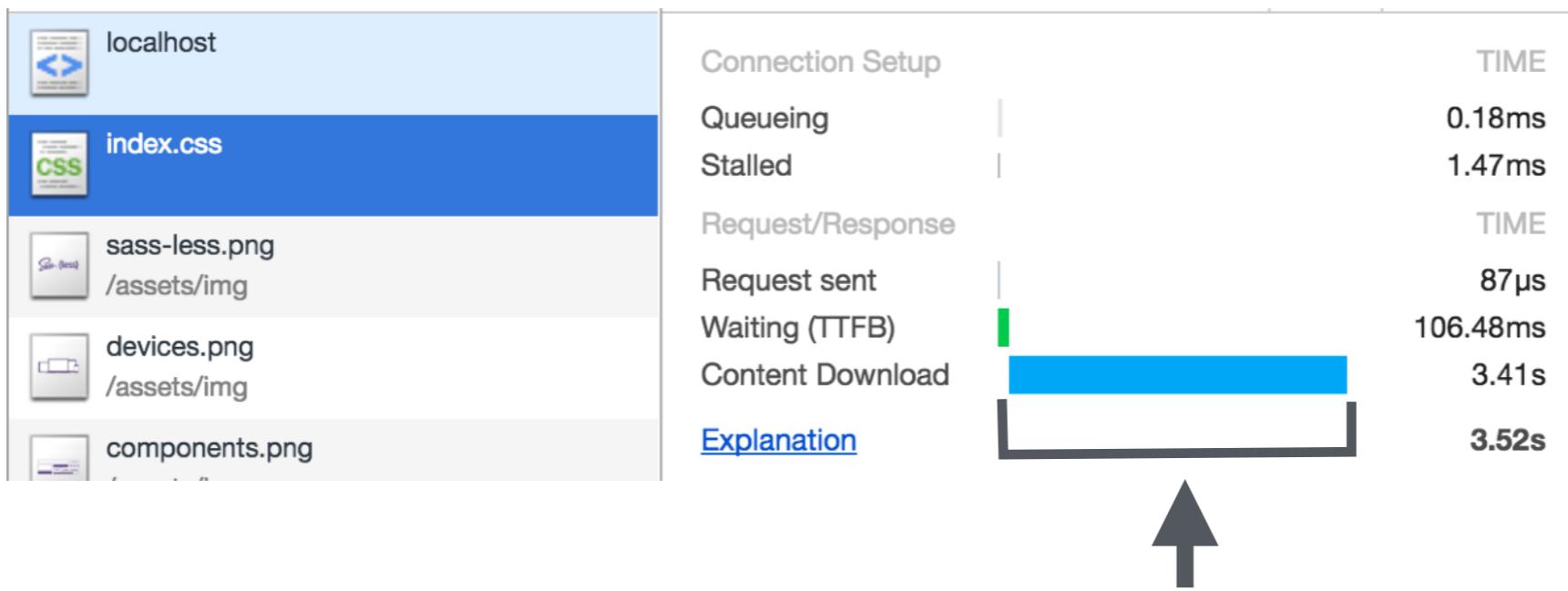
Name	Headers	Preview	Response	Timing
a11y-viewer.netlify.com	General Request URL: https://a11y-viewer.netlify.com/ Request Method: GET Status Code: 304 Remote Address: 54.93.54.1:443			
www.voorhoede.nl				
?fmt=1&num=1&cv=8&frm=2				

[example on netlify.com](#)
[see also caching best practices on jakearchibald.com](#)

COMPRESSION



PROBLEM



download time = file size / download speed

Name	Method	Status	Type	Size	Time
index.css	GET	200 OK	stylesheet	166KB 166KB	4.86s 112ms

uncompressed: download size === file size

Compression

encoding information using
fewer bits than
the original representation.

en.wikipedia.org/wiki/Data_compression

THEORY

Uncompressed

Zopfli / Gzip

Brotli

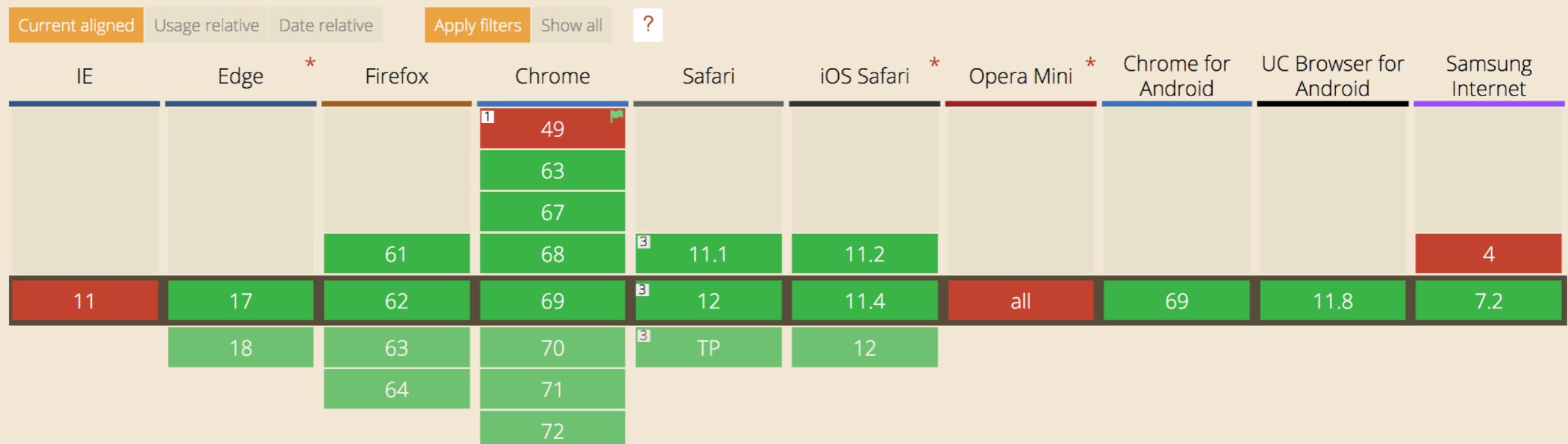
varvy.com/performance/brotli.html

BROTLI SUPPORT

Brotli Accept-Encoding/Content-Encoding - OTHER

Usage % of all users
Global 85%

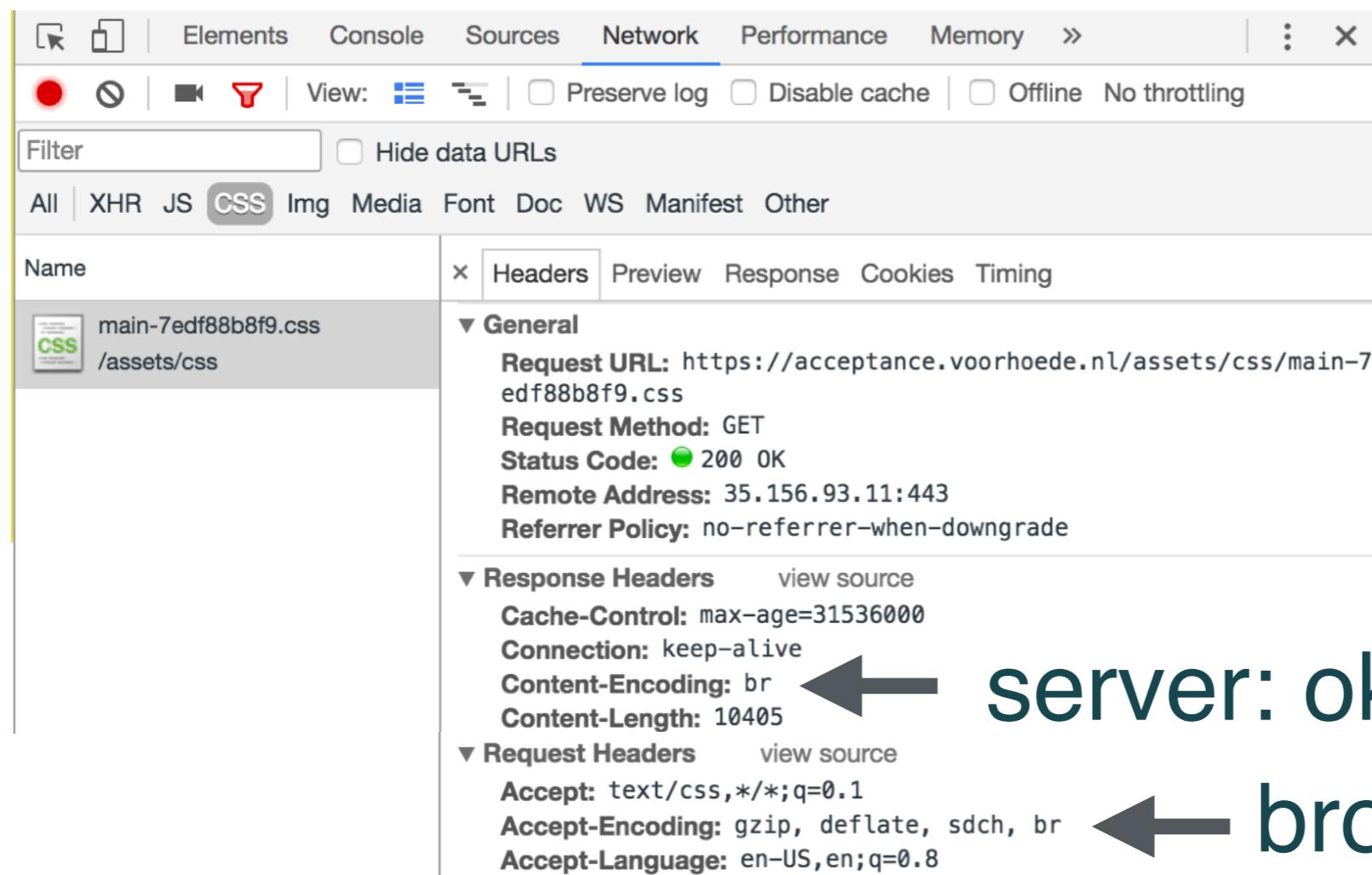
More effective lossless compression algorithm than gzip and deflate.



*requires HTTPS

caniuse.com/#feat=brotli

ENCODING HEADERS



The screenshot shows the Chrome DevTools Network tab. A CSS file named "main-7edf88b8f9.css" from the URL "https://acceptance.voorhoede.nl/assets/css/main-7edf88b8f9.css" is selected. The "Headers" tab is active. The "Response Headers" section shows:

- Cache-Control: max-age=31536000
- Connection: keep-alive
- Content-Encoding: br ← server: okay, I'm using this
- Content-Length: 10405

The "Request Headers" section shows:

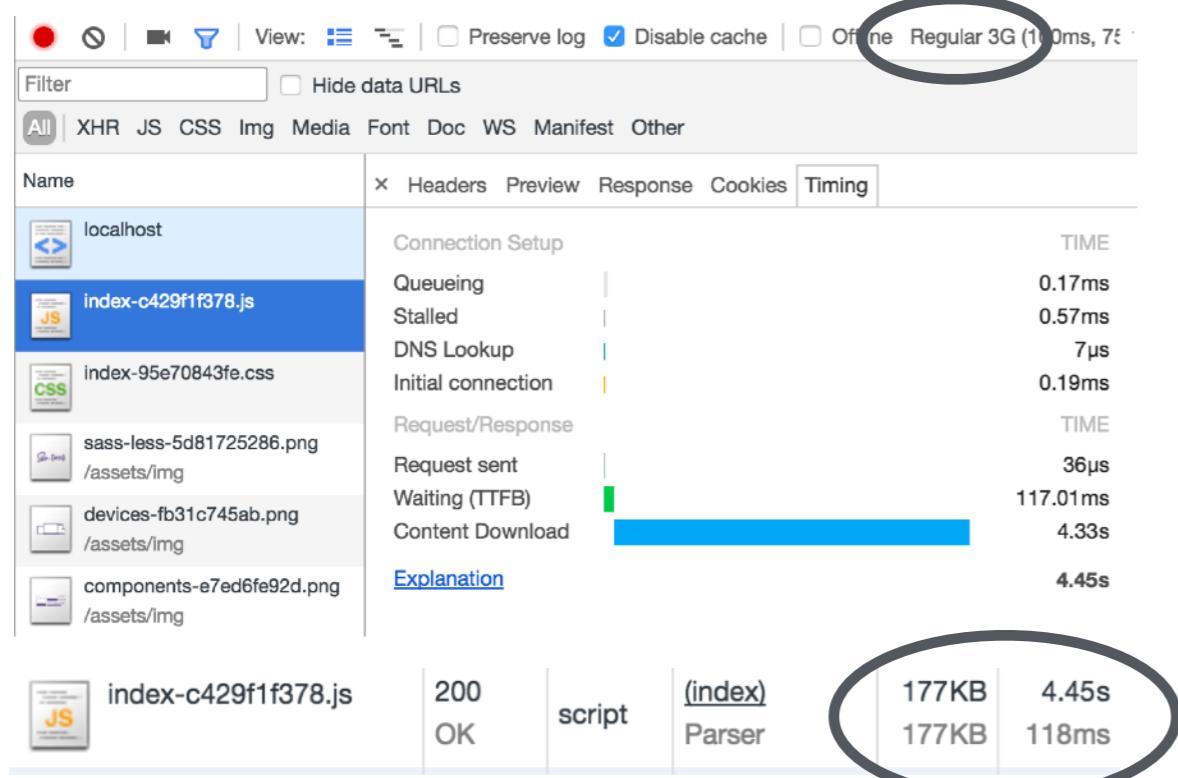
- Accept: text/css,*/*;q=0.1 ← browser: I know how
- Accept-Encoding: gzip, deflate, sdch, br
- Accept-Language: en-US,en;q=0.8

server: okay, I'm using this
← browser: I know how
to decompress these

acceptance.voorhoede.nl/en/blog/static-site-explosion-with-brotli-and-gzip/

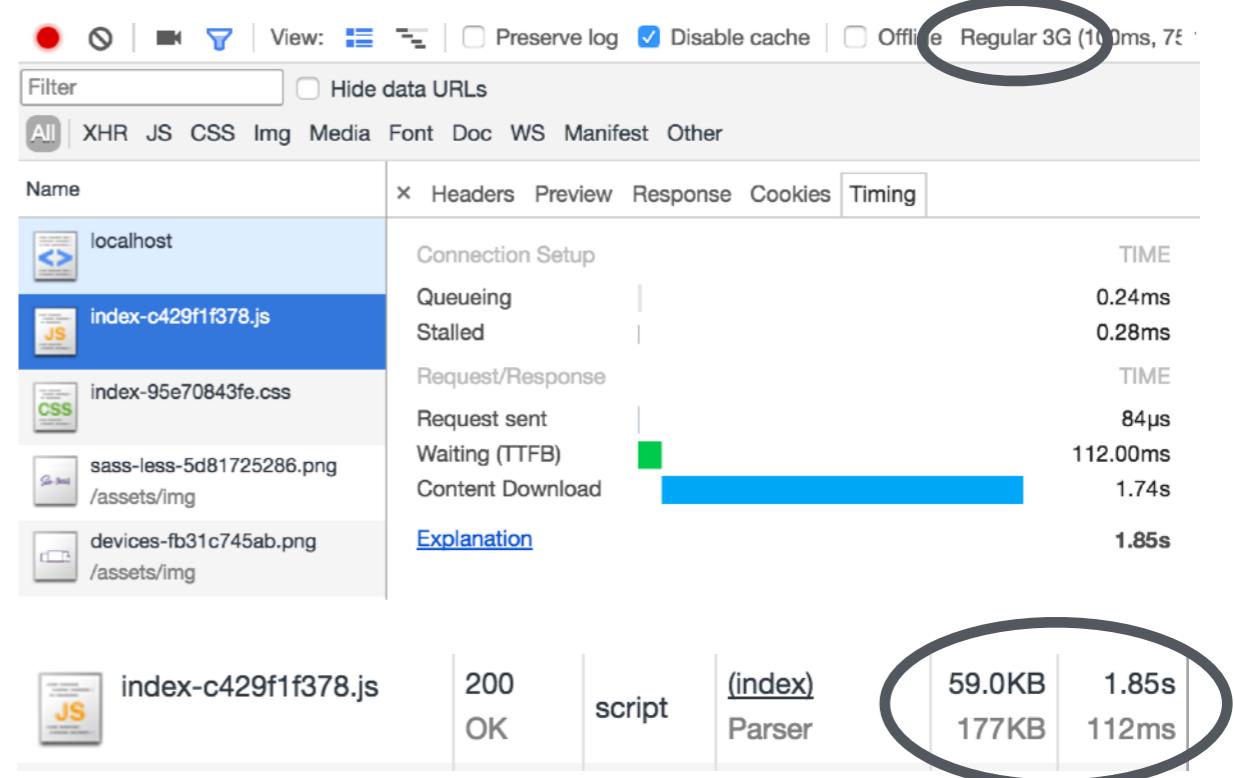
COMPARE

BEFORE



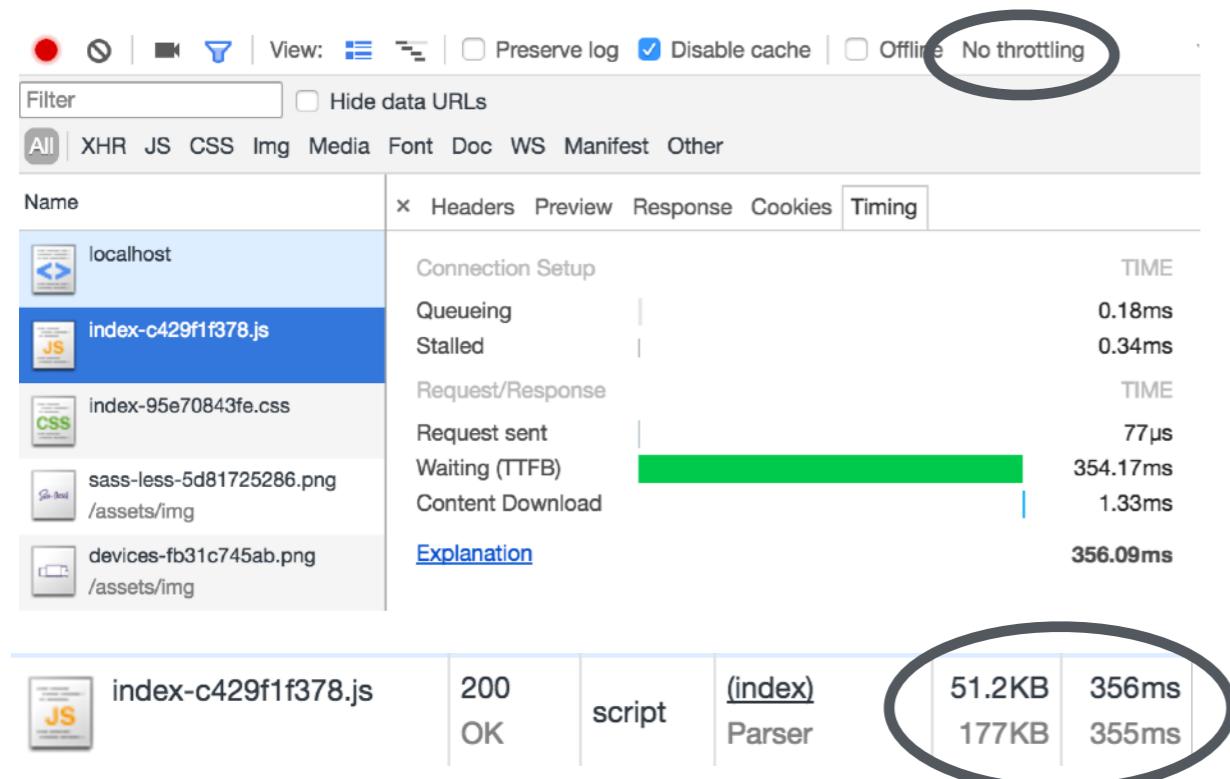
no compression
117 ms wait - 4.45 sec total

AFTER



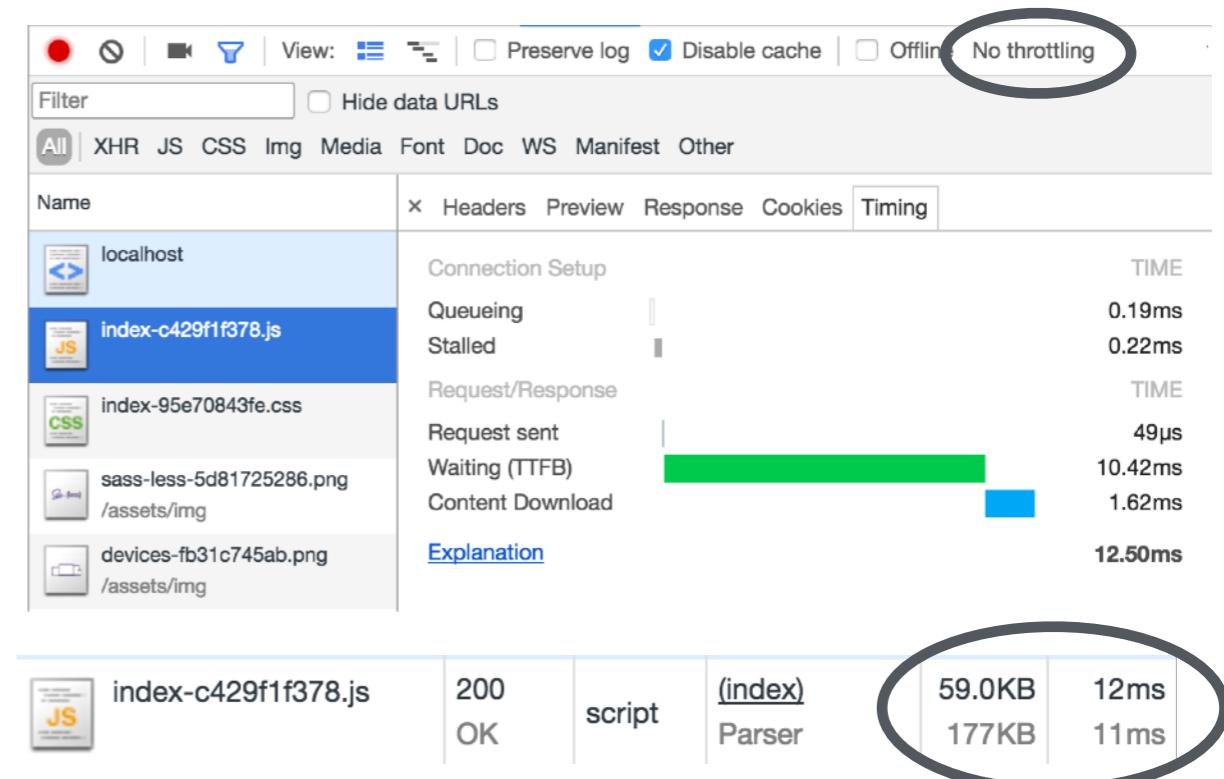
brotli quality 4, no caching
112 ms wait - 1.85 ms total

COMPARE MAX COMPRESS



brotli quality 11, no caching
354 ms wait - 356 ms total

DEFAULT + CACHE

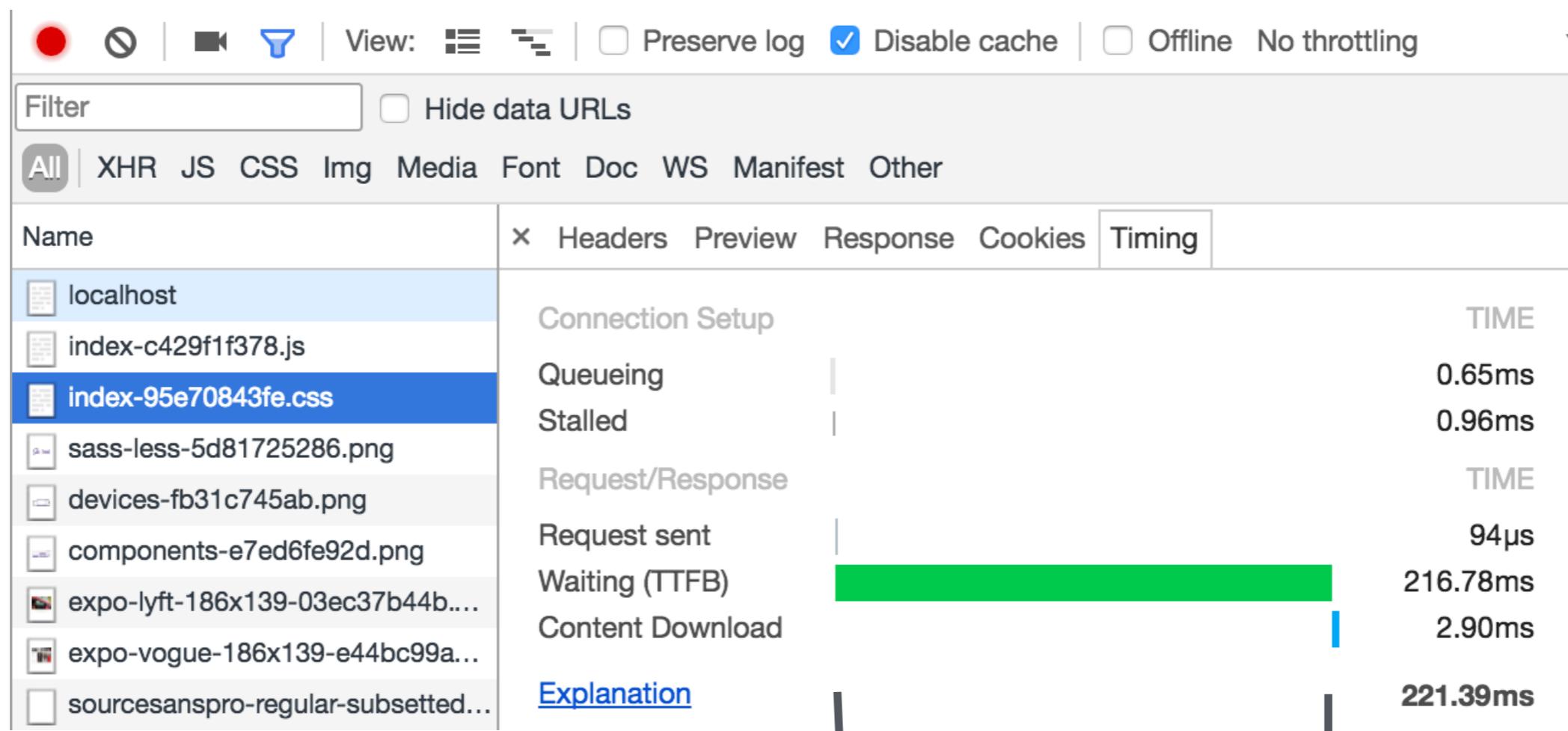


brotli quality 4, caching
10 ms wait - 12 ms total

PRECOMPRESSION

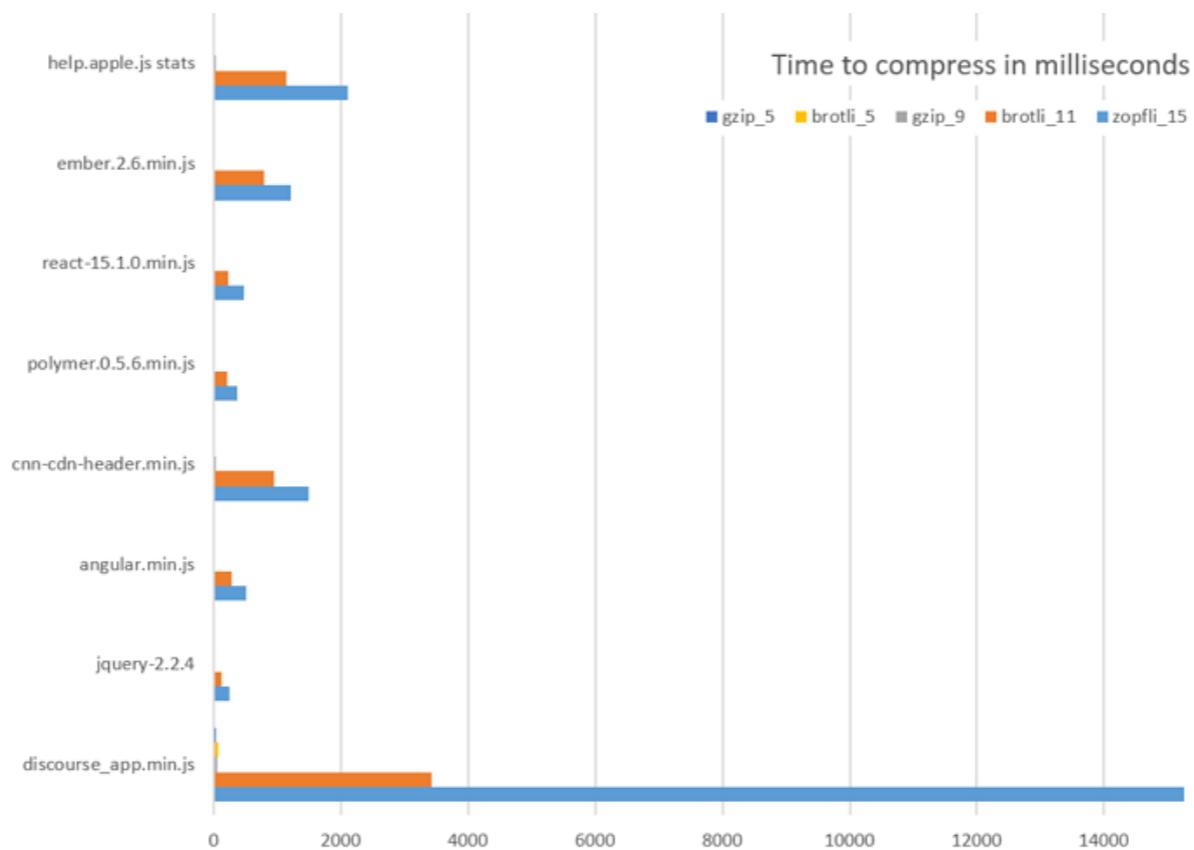


PROBLEM



long wait caused by on-the-fly compression

COMPRESSION TIME



	zopfli_15	brotli_11	gzip_9	brotli_5	gzip_5
discourse_app.min.js	15261	3426	60	70	34
jquery-2.2.4	254	132	5	4	4
angular.min.js	518	282	9	7	5
cnn-cdn-header.min.js	1489	948	30	17	12
polymer.0.5.6.min.js	375	207	6	5	4
react-15.1.0.min.js	467	237	6	6	4
ember.2.6.min.js	1217	784	19	14	11
help.apple.js stats	2101	1150	37	24	16

samsaffron.com/archive/2016/06/15/the-current-state-of-brotli-compression

Static site implosion with Brotli and Gzip



by Thadee

November 28, 2016

Table of contents

[On Brotli compression](#)

[Brotlify all the things with maximum compression!](#)

[Static compression](#)

URL	NO COMPRESSION	GZIP	BROTLI
index.html (initial view)	34.25 KB	12.68 KB (-63%)	11.34 KB (-67%)
main.css	53.10 KB	12.89 KB (-76%)	11.46 KB (-78%)
index.js	4.90 KB	2.75 KB (-44%)	2.52 KB (-49%)
animation.svg	69.38 KB	5.37 KB (-92%)	4.47 KB (-94%)

This is the request/response timing from the Chrome DevTools network tab:



voorhoede.nl/en/blog/static-site-implosion-with-brotli-and-gzip/

PRECOMPRESSED FILES

cache/



...



index.css



index-67760ac4ed.css



index-67760ac4ed.css.br



index-67760ac4ed.css.gz



index.js



index-40b67b7222.js



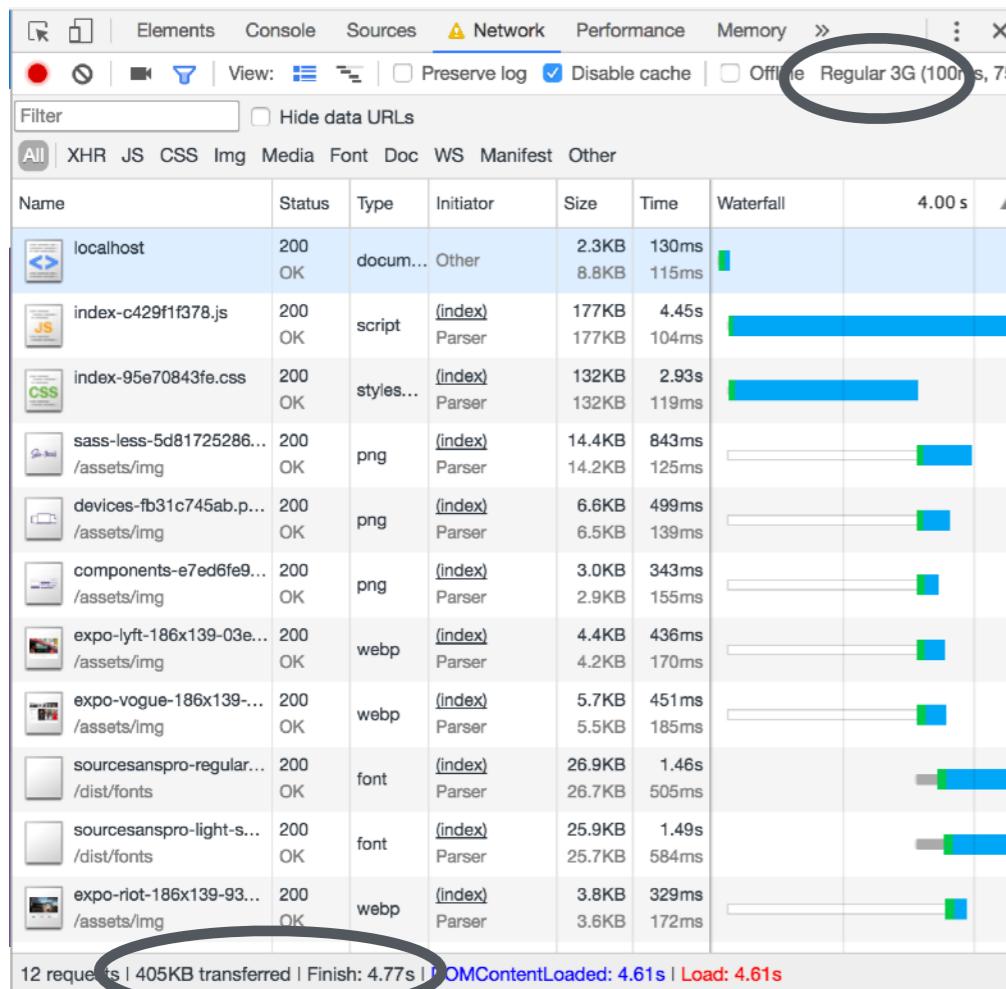
index-40b67b7222.js.br

index-40b67b7222.js.gz

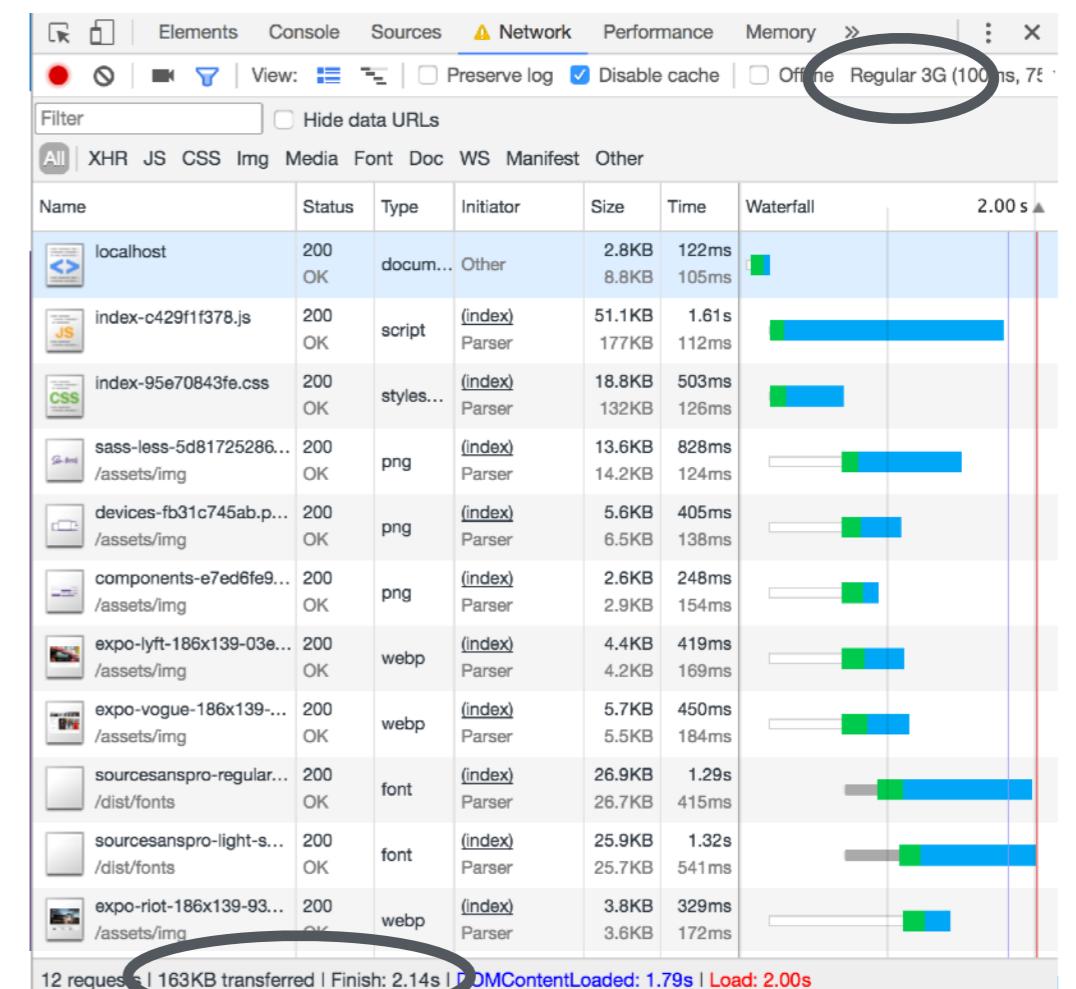
...

COMPARE

DYNAMIC PRECOMPRESSED



shrink-ray default settings
total 405KB, finish: 4.77 sec



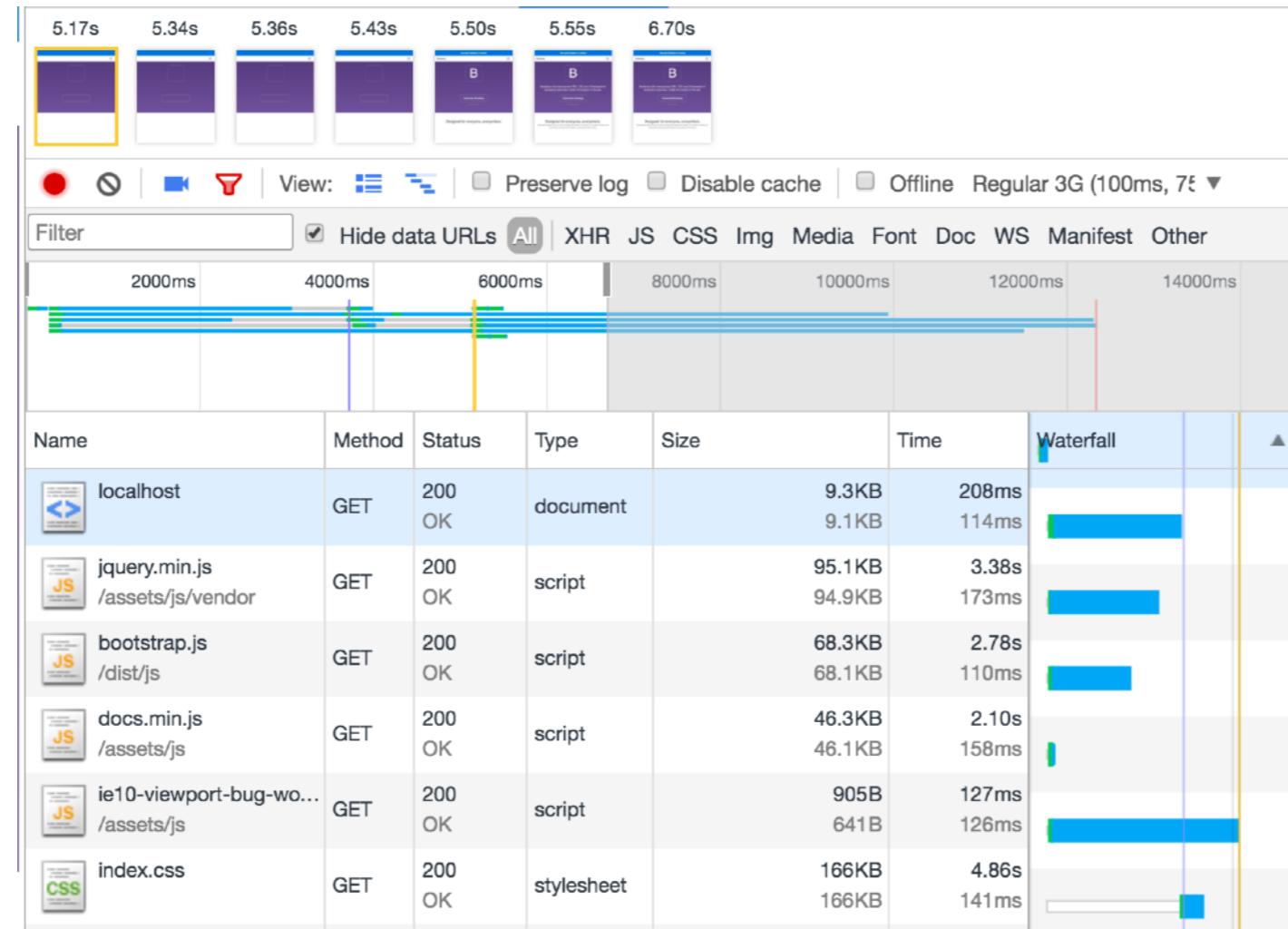
max precompression
total 163KB, finish: 2.14 sec

JS



PROBLEM

scripts [



render blocking ↑

PLAIN SCRIPT ELEMENTS

```
<script src="//other-domain.com/1.js"></script>  
<script src="2.js"></script>
```

Spec says: Download together, execute in order after any pending CSS, block rendering until complete.

Browsers say: Yes sir!

DEFER SCRIPTS

```
<script defer src="//other-domain.com/1.js"></script>
<script defer src="2.js"></script>
```

Spec says: Download together, execute in order just before DOMContentLoaded. Ignore “defer” on scripts without “src”.

IE < 10: might execute 2.js halfway through execution of 1.js.

Unsupported browsers (in red): I have no idea what this “defer” thing is, I’m going to ~~load the scripts as if it weren’t~~ there.

ASYNC SCRIPTS

```
<script async src="//other-domain.com/1.js"></script>
<script async src="2.js"></script>
```

Spec says: Download together, execute in whatever order they download in.

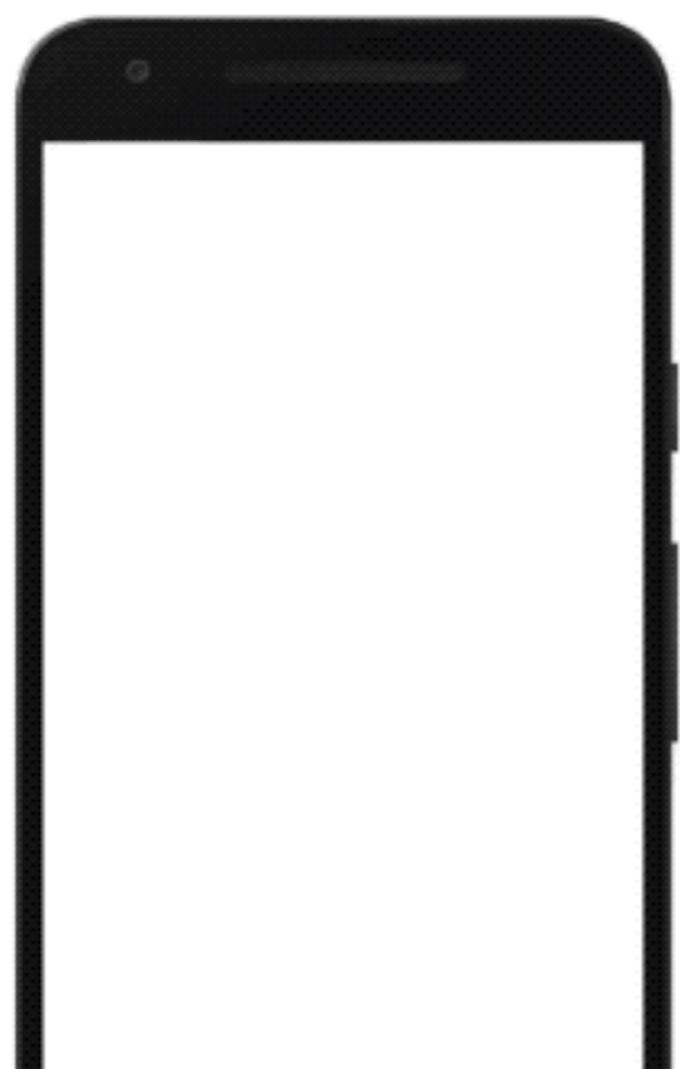
Unsupported browsers (in red): What's 'async'? I'm going to load the scripts as if it weren't there.

Other browsers say: Yeah, ok.

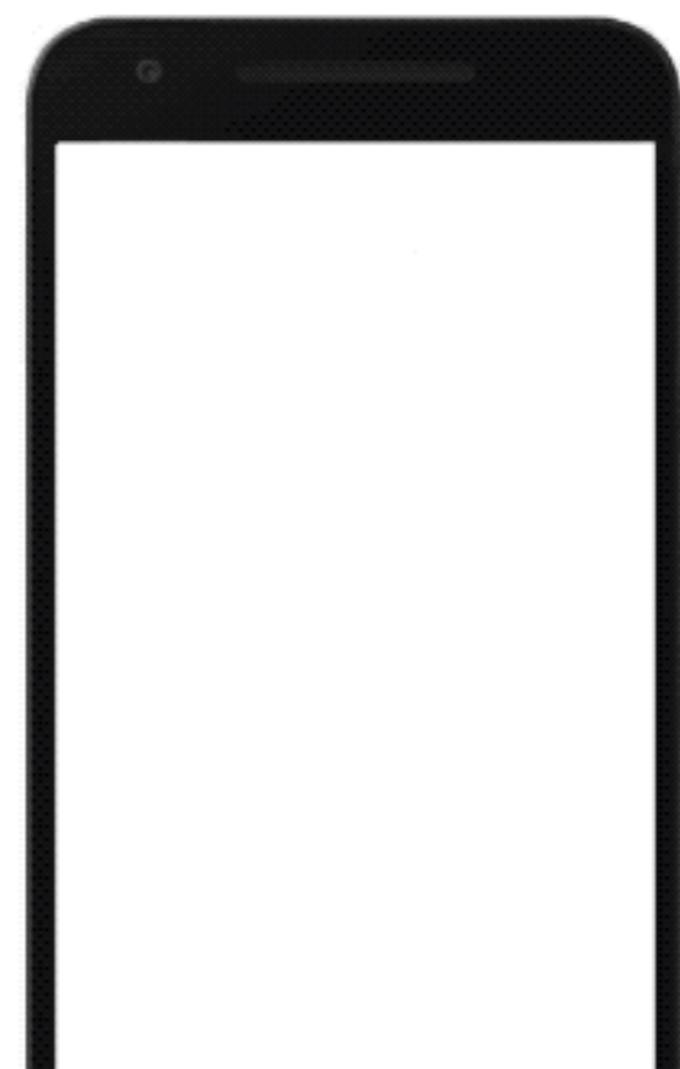
html5rocks.com/en/tutorials/speed/script-loading/

TIME TO INTERACTIVE

0s 00

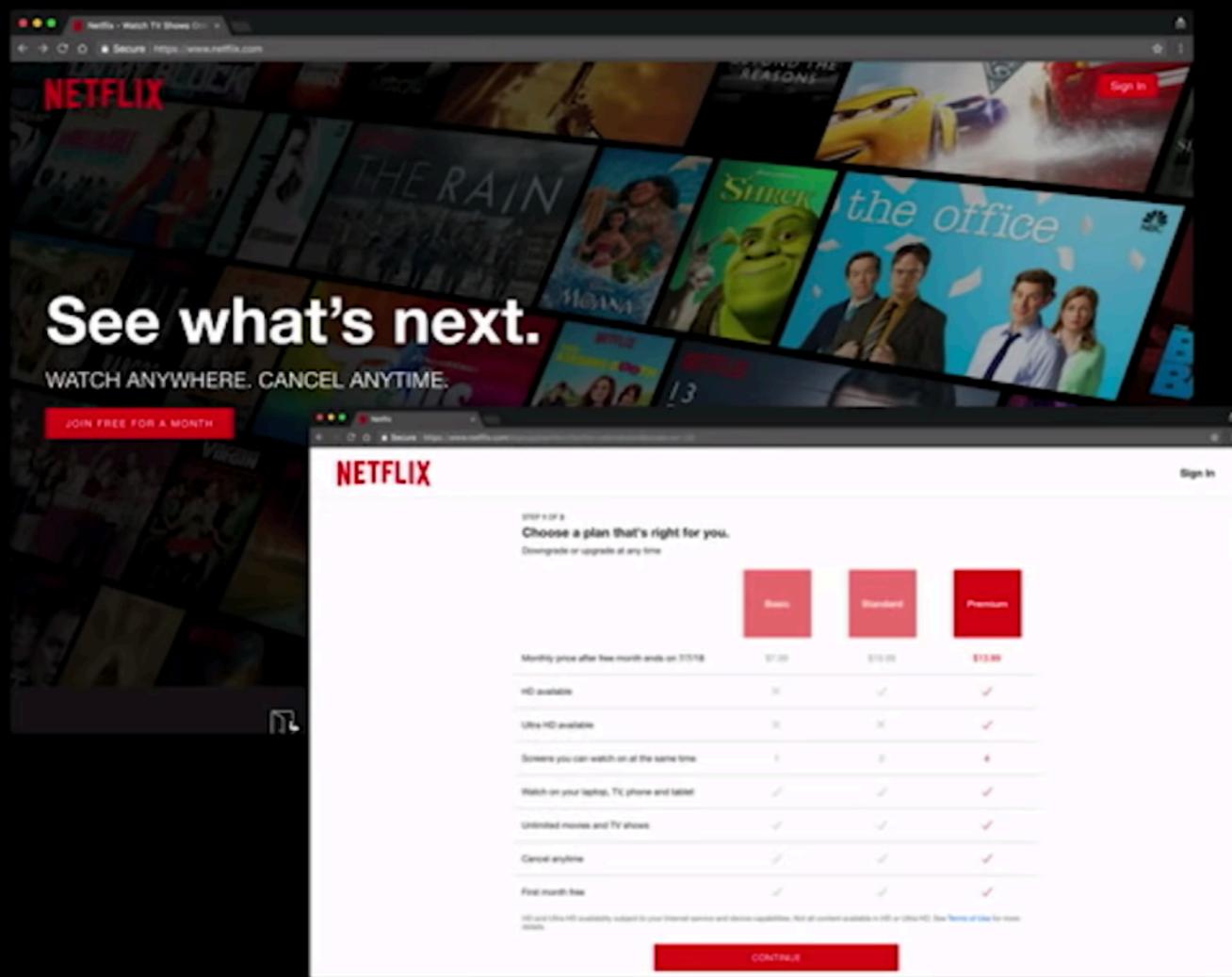


0s 00



[The cost of javascript](#)

NETFLIX



NON-MEMBER LANDING PAGE

**REMOVING
CLIENT-SIDE REACT,
PRE-FETCHING IT
FOR FUTURE PAGES
IMPROVED TIME-
TO-INTERACTIVE BY
50%**

REACT WAS STILL USED SERVER-SIDE

The cost of javascript

JavaScript bytes != JPEG bytes

~170KB

Network Transmission

~170KB

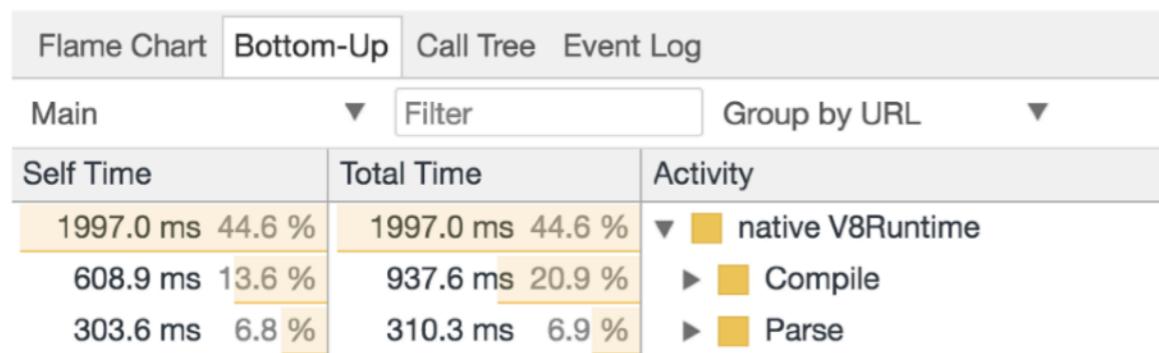
main-javascript-bundle.js

3.4 s 170KB

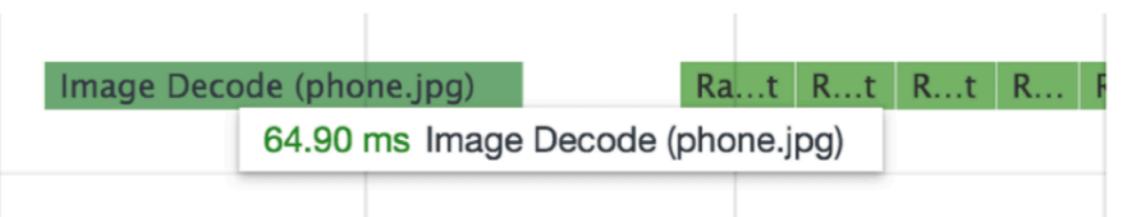
photo.jpg

3.4 s 170KB

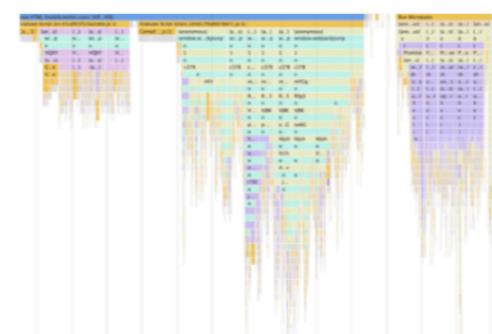
Resource Processing



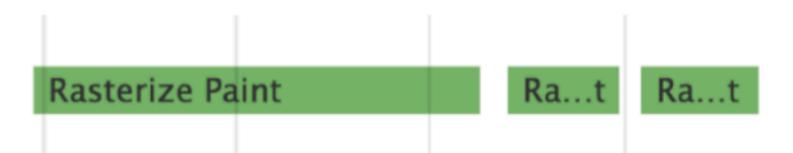
~2s in Parse/Compile



0.064s in Image Decode



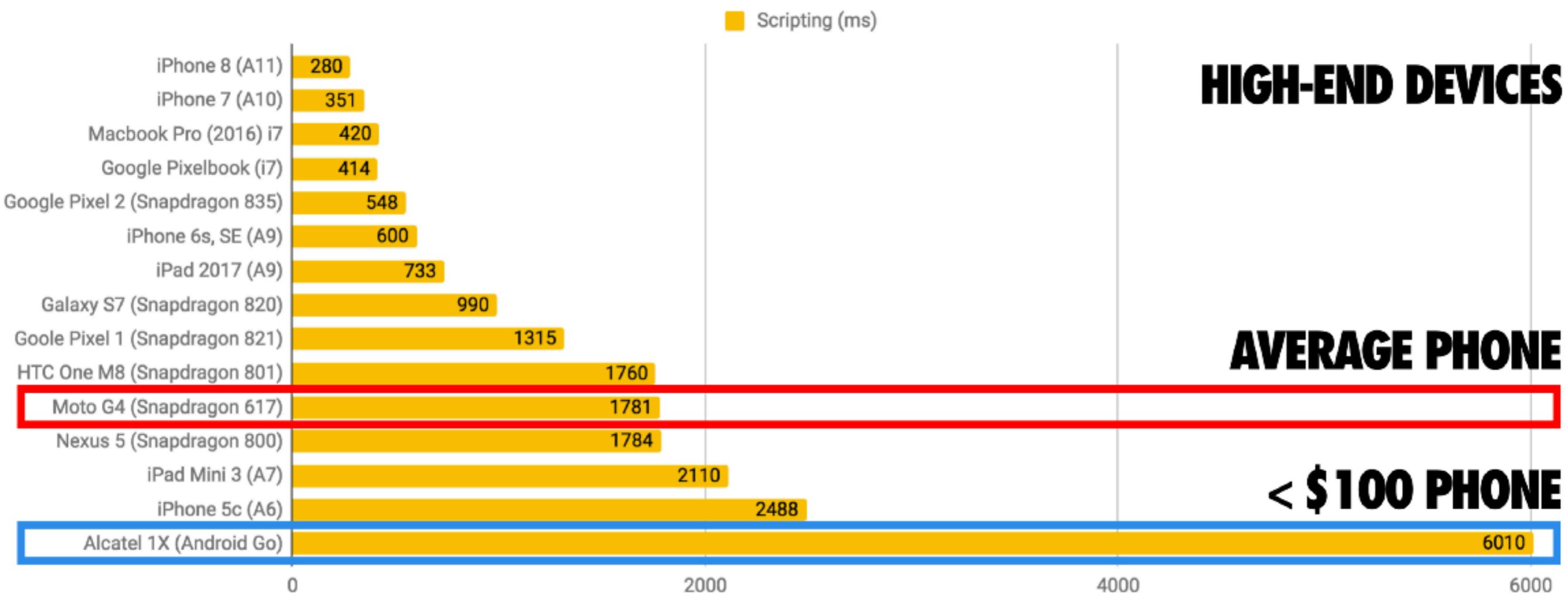
~1.5s in Execution



0.028s in Rasterize Paint

The cost of javascript

2018 JAVASCRIPT PROCESSING TIMES



Tests run during July, 2018 on hardware running the latest versions of Android and iOS available

1MB JS UNCOMPRESSED (200KB min/compressed)

The cost of javascript

JS PROCESSING FOR CNN.COM

HIGH-END PHONE

iPhone 8 (A11) 3967 2693 1148

iPhone 7+ (A10) 5669 2002 2582

iPhone SE (A9) 5476 4464 1713

iPhone 6s (A9) 6074 4483 1737

iPhone 6 (A8) 12038 5614 3069

Moto G4 (Snapdragon 617) 13355 4107 1002

Samsung S7 (Exynos 8890) 14354 3250 1048

AVERAGE PHONE - 9s SLOWER

< \$100 PHONE - 32s SLOWER

Alcatel 1X 36284 5047 1023

Thinkpad T430 (Core i5 3320M) 7179 4955 2851

Desktop (Core i7-5930K) 2061 818 265

Thinkpad Yoga (Core i7 6600U) 2891 1243 422

0 10000 20000 30000 40000 50000

The cost of javascript

Code splitting is the process of
splitting pieces of your code into
async chunks (at build time)

CHROME DEV TOOLS

The screenshot shows the Chrome Dev Tools interface. The top navigation bar includes tabs for Elements, Console (which is selected), Sources, Network, Performance, and Memory. Below the navigation is a toolbar with icons for play, stop, and refresh, followed by a dropdown menu set to 'top'. A 'Filter' input field and a 'Default levels' dropdown are also present. The main content area displays a configuration section with several checkboxes: 'Hide network', 'Preserve log', 'Selected context only', 'Group similar' (which is checked), 'Log XMLHttpRequests', 'Eager evaluation' (which is checked), and 'Autocomplete from history'. Below this is a 'Coverage' tab, which is active. The coverage report lists various URLs and their usage statistics:

URL	Type	Total Bytes	Unused Bytes	% Unused
/vendor.498e9bedca67bd	JS	296 569	155 581	52.5 %
https://www.voorhoed... /	CSS+JS	69 172	34 328	49.6 %
https://ww... /analytics.js	JS	44 130	24 724	56.0 %
/app.458d4abd79367dc1	JS	66 942	24 481	36.6 %
/index.e04ac01db73529e	JS	68 951	10 602	15.4 %
/default.e0dfed14ea278d	JS	47 386	9 167	19.3 %
/manifest.10d89cee2a4b6	JS	2 147	470	21.9 %

A note at the bottom states: "253 KB of 581 KB bytes are not used. (44%)". On the left side of the slide, there is a yellow sidebar with the text "Wij zijn Front-end developers" and "Hoe kunnen we helpen?", accompanied by three cartoon characters (a superhero, a woman, and a man with a sword). At the bottom left, there is a vertical scroll bar icon and the text "We werken lean en agile", "Beginnen klein en itereren snel", and "om je van een idee,".

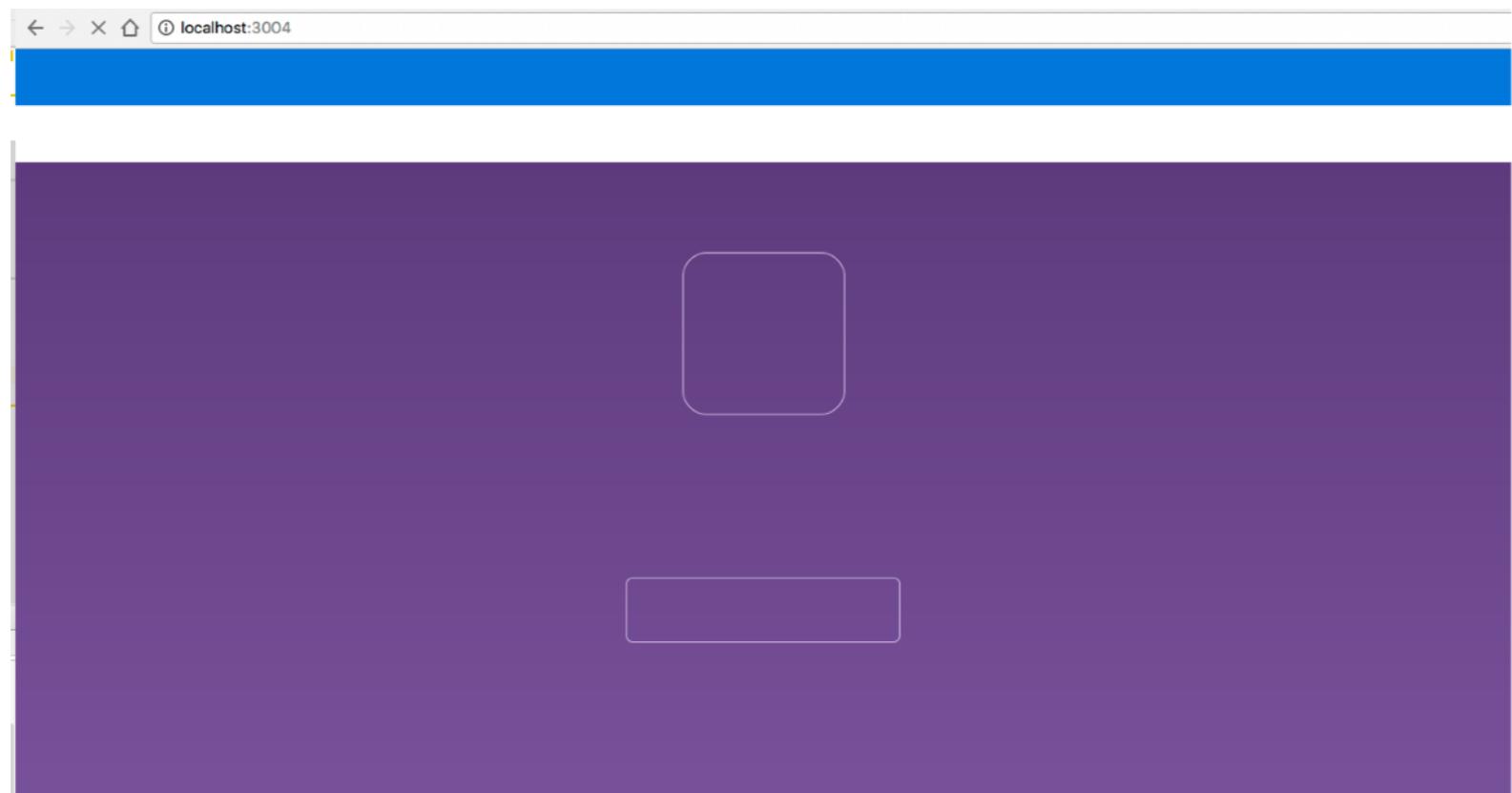
FONT LOADING



Save ~65% in custom web
font file size using
font subsetting

[font squirrel font subsetting](#)

FOIT PROBLEM



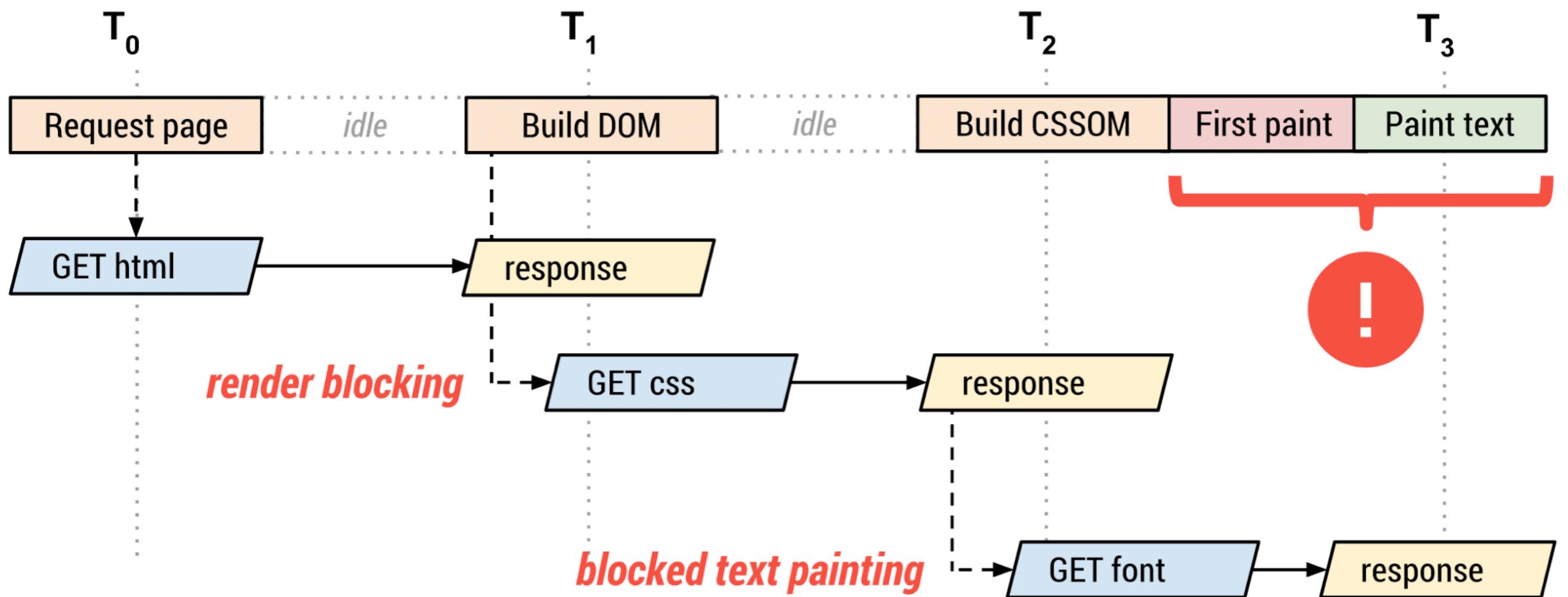
Sass {less}



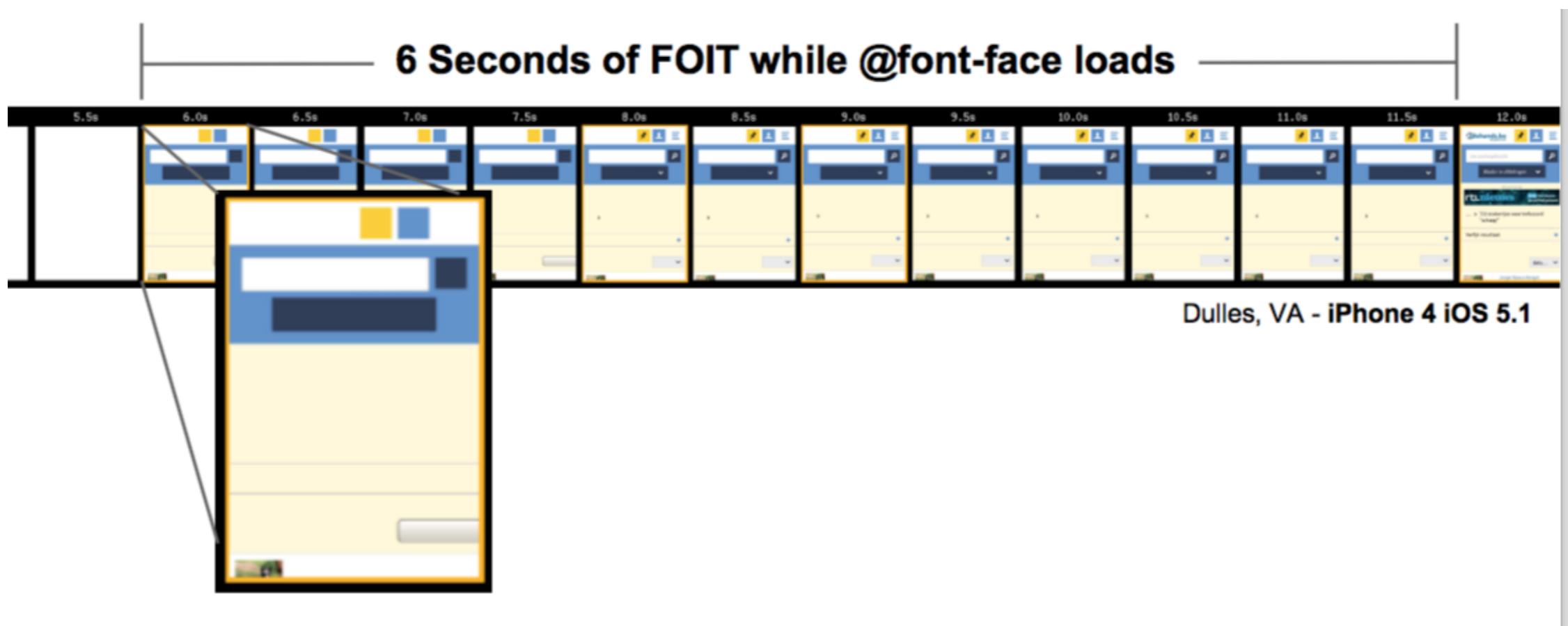
BLOCKING FONT LOADING

```
@font-face {  
    font-family: 'source_sans_pro';  
    src: url('sourcesanspro.woff2') format('woff2'),  
        url('sourcesanspro.woff') format('woff');  
}  
  
html {  
    font-family: 'source_sans_pro', sans-serif;  
}
```

BLOCKING FONT LOADING



Web Font Optimization



From FilamentGroup 2dehands review

FONT LOADING TIMEOUT

Browser	Timeout	Fallback	Swap
Chrome 35+	3 seconds	Yes	Yes
Opera	3 seconds	Yes	Yes
Firefox	3 seconds	Yes	Yes
Internet Explorer	0 seconds	Yes	Yes
Safari	No timeout	N/A	N/A

[Controlling Font Performance with font-display](#) [Controlling Font Performance with font-display](#)

FONT RENDERING CONTROLS



FONT DISPLAY

```
@font-face {  
    font-family: 'Noto';  
    src: url('/path/to/noto.woff2')  
format('woff2');  
    font-display: swap;  
}
```

Possible values are auto, block, swap, fallback and optional

[Font display property](#)

FOUT WITH FONT DISPLAY



DE VOORHOEDE
front-end developers

WORK BLOG TEAM CONTACT NL | EN

.NET can has wow front-end too

Building a modular front-end in .NET Core



by Joao
September 16, 2016

Recently we successfully integrated our front-end approach into a .NET setup. We created a demo viewer, defined a suitable project structure, and shared our code for it on Github:

[.NET Core front-end development guide →](#)

In this blog post, we explain why and how we set it up.

Why we need a .NET front-end stack

Node.js on the back-end is still a fairly small affair, moreso in corporate environments where .NET and Java still reign supreme. However, as front-enders we prefer to work with Node.js — or, to be honest, we *need* to, as many of the tools we use are written with Node.

We often try to **combine the front-end and back-end into a single project environment**. If for some reason we can't, we deliver a static, modular front-end using Nunjucks templates and stub data. Back-end developers then build the actual application out of prerendered templates and precompiled assets.

This conversion however is time-consuming and prone to errors. So, to smoothen this process, we built a **front-end development stack for .NET**.

Running .NET on Mac OS

ASP.NET is available for Windows only, whereas all of De Voorhoede (and many



DE VOORHOEDE
front-end developers

WORK BLOG TEAM CONTACT NL | EN

.NET can has wow front-end too

Building a modular front-end in .NET Core



by Joao
September 16, 2016

Recently we successfully integrated our front-end approach into a .NET setup. We created a demo viewer, defined a suitable project structure, and shared our code for it on Github:

[.NET Core front-end development guide →](#)

In this blog post, we explain why and how we set it up.

Why we need a .NET front-end stack

Node.js on the back-end is still a fairly small affair, moreso in corporate environments where .NET and Java still reign supreme. However, as front-enders we prefer to work with Node.js — or, to be honest, we *need* to, as many of the tools we use are written with Node.

We often try to **combine the front-end and back-end into a single project environment**. If for some reason we can't, we deliver a static, modular front-end using Nunjucks templates and stub data. Back-end developers then build the actual application out of prerendered templates and precompiled assets.

This conversion however is time-consuming and prone to errors. So, to smoothen this process, we built a **front-end development stack for .NET**.

Running .NET on Mac OS

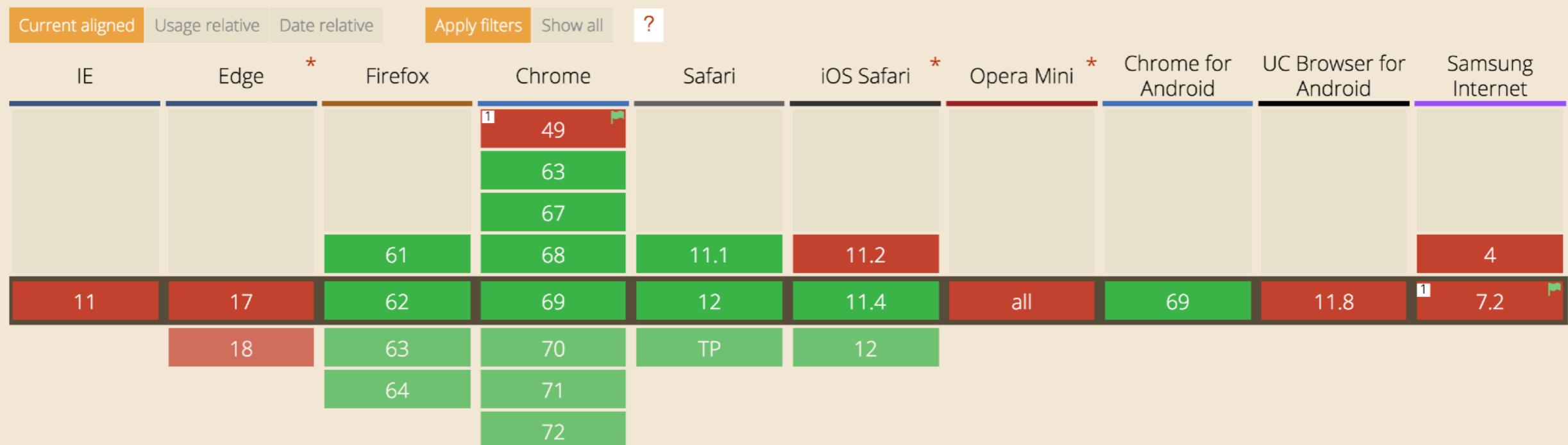
ASP.NET is available for Windows only, whereas all of De Voorhoede (and

FONT RENDERING CONTROLS

CSS font-rendering controls [- WD](#)

Usage
Global % of all users 74.78%

@font-face descriptor (currently defined as `font-display`) that allows control over how a downloadable font renders before it is fully loaded.



COMPARE FOIT



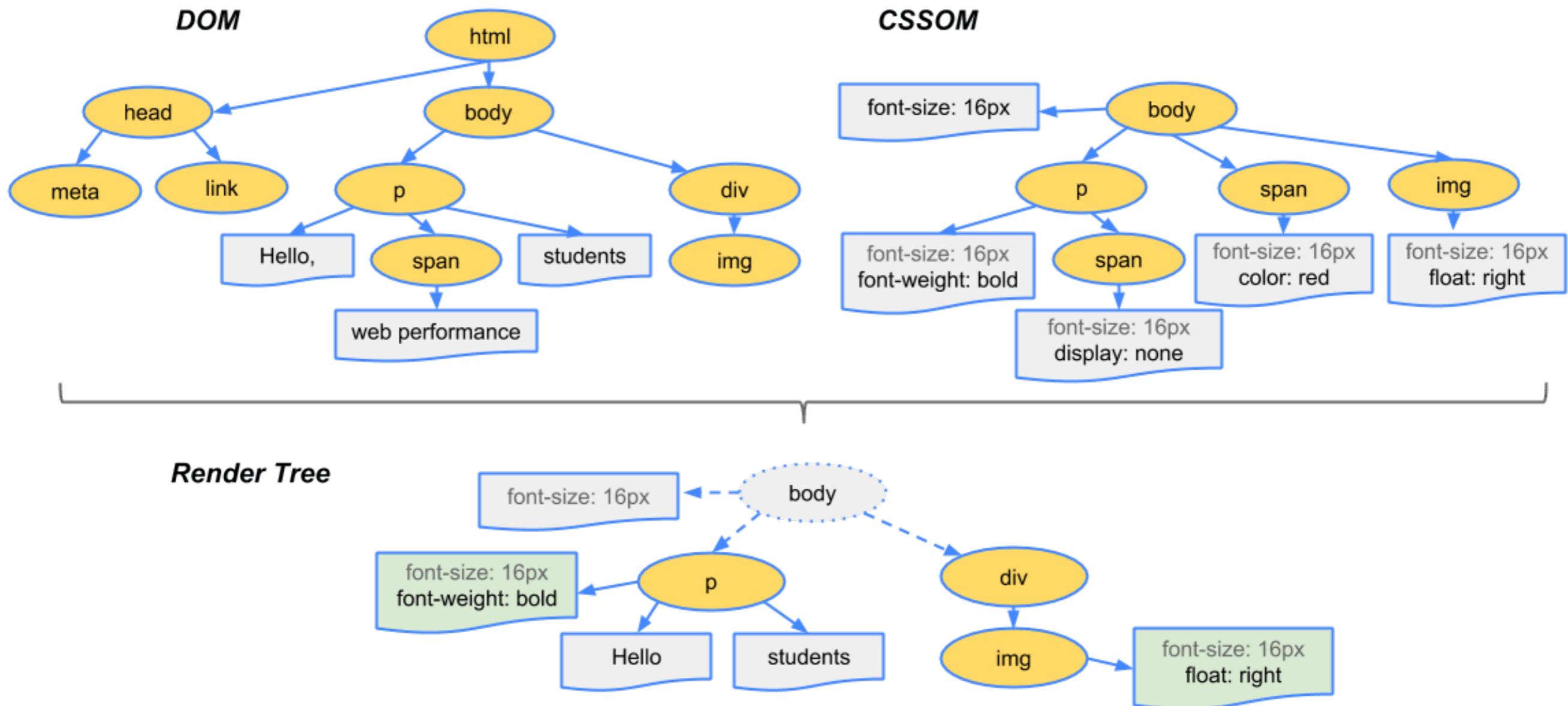
COMPARE FOUT



LOAD CSS



PROBLEM



localhost:3003

Aww yeah, Bootstrap 4 is coming!

Bootstrap

B

Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web.

Download Bootstrap

Currently v3.3.7

Designed for everyone, everywhere.

Bootstrap makes front-end web development

Elements Console Sources Network Performance Memory Application Security Audits PageSpeed

3.61s 3.93s 4.09s 4.31s 4.46s 4.47s 6.43s 6.62s 6.71s 8.50s 8.54s

View: Preset log Disable cache Offline Regular 3G (100ms, 75ms) ▾

Filter Hide data URLs All XHR JS CSS Img Media Font Doc WS Manifest Other

1000ms 2000ms 3000ms 4000ms 5000ms 6000ms 7000ms 8000ms 9000ms 10000ms 1100ms

Name	Method	Status	Type	Initiator	Size	Time	Waterfall	Duration	Total Time	Count
localhost	GET	200	document	Other	9.4...	212...		4.00 s	6.00 s	8
bootstrap.css	GET	200	stylesheet	(index)	144...	3.31s				
docs.css	GET	200	stylesheet	(index)	30.6...	967...				
expo-newsweek.jpg	GET	200	image	(index)	197...	7.25s				
jquery.min.js	GET	200	script	(index)	95.1...	2.36s				
bootstrap.min.js	GET	200	script	(index)	36.4...	827...				
docs.min.js	GET	200	script	(index)	46.3...	2.83s				
ie10-viewport-bug-workar...	GET	200	script	(index)	905B	128...				
sass-less.png	GET	200	image	(index)	14.4...	904...				
devices.png	GET	200	image	(index)	6.7...	514...				
components.png	GET	200	image	(index)	3.1...	357...				
sourcesanspro-regular.woff2	GET	200	font	(index)	75.4...	4.92s				
sourcesanspro-light.woff2	GET	200	font	(index)	74.6...	4.88s				
expo-lyft.jpg	GET	200	image	(index)	156...	8.31s				
expo-vogue.jpg	GET	200	image	(index)	194...	9.13s				
expo-riot.jpg	GET	200	image	(index)	158...	8.09s				
ZeroClipboard.swf?noCach...	GET	200	x-shockwave-flash	docs.min...	2.4...	2.27s				

17 requests | 1.2MB transferred | Finish: 13.63s | DOMContentLoaded: 6.41s | Load: 13.64s

BLOCKING STYLE LOADING

```
<head>
  <!-- Loading early means blocking page: -&gt;
  &lt;link rel="stylesheet" href="index-
f3c2b4.css"&gt;
&lt;/head&gt;</pre>
```

```
<body>
  <!-- Loading late means flash of unstyled
content: -&gt;
  &lt;link rel="stylesheet" href="index-
f3c2b4.css"&gt;
&lt;/body&gt;</pre>
```

LOAD STYLES ASYNC

```
<script>
    // (include loadCSS here)
    var stylesheet = loadCSS('index-
f3c2b4.css');
</script>

<noscript>
    <link rel="stylesheet" href="index-
f3c2b4.css">
</noscript>
```

RESULT

```
<noscript><link href="/index.css" rel="stylesheet"></noscript>
<script data-load-css>
  {% include "assets/js/vendor/loadcss.min.js" %}
  window.loadCSS('/index.css');
</script>
```

CRITICAL CSS



RESULT

set cookie when CSS is loaded

```
<noscript>
  <link href="{{ revUrl('/index.css') }}" rel="stylesheet">
</noscript>

<script data-load-css>
  {% include "assets/js/vendor/loadcss.min.js" %}
  {% include "assets/js/vendor/onloadcss.min.js" %}

  (function(window) {
    var cssUrl = '{{ revUrl("/index.css") }}';
    var stylesheet = window.loadCSS(cssUrl);
    window.onloadCSS(stylesheet, function() {
      console.log('Stylesheet has loaded.');
      cookie('cssLoaded', cssUrl, 365);
    });
  })(window);
</script>
```

RESULT

Check if latest CSS is loaded on server

```
const data = {
  fontsLoaded: req.cookies.fontsLoaded,
  cssLoaded: (req.cookies.cssLoaded === revUrl('/index.css')) ,
};

res.render(`./${filename}`, data, (err, html) => ...);
```

RESULT

conditionally load CSS async based on template variable

```
<!DOCTYPE html>
{%
  if not fontsLoaded %
<html lang="en">
{%
  else %
<html lang="en" class="fonts-loaded">
{%
  endif %
<head>
  {%
    include './app-meta.html' %

  {%
    if not cssLoaded or not fontsLoaded %
<script data-cookie-helper>
  {%
    include "assets/js/vendor/cookie.min.js" %
</script>
{%
  endif %

{%
  if cssLoaded %
<link href="{{ revUrl('/index.css') }}" rel="stylesheet">
{%
  else %
    {%
      include './load-css.html' %
{%
  endif %}
```

CRITICAL CSS



PROBLEM

[Skip to main content](#)

[Aww yeah, Bootstrap 4 is coming!](#)

[Toggle navigation](#) [Bootstrap](#)

- [Getting started](#)
- [CSS](#)
- [Components](#)
- [JavaScript](#)
- [Customize](#)
- [Themes](#)
- [Expo](#)
- [Blog](#)

B

Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web.

[Download Bootstrap](#)

Currently v3.3.7

Designed for everyone, everywhere.

Bootstrap makes front-end web development faster and easier. It's made for folks of all skill levels, devices of all shapes, and projects of all sizes.



Critical CSS is the minimum set of above the fold blocking CSS that we need to make the page recognisable to the user

CRITICAL CSS

```
<html>
<head>
  <style> /* inlined critical CSS */ </style>
  <script> loadCSS('non-critical.css'); </script>
  <noscript>
    <link rel="stylesheet" href="non-critical.css">
  </noscript>
</head>
<body>
...
</body>
</html>
```

APPLY CRITICAL CSS

The image shows two side-by-side screenshots of the De Voorhoeve website. Both screenshots feature a yellow header bar at the top with the company logo and navigation links (WERK, BLOG, TEAM, CONTACT, NL | EN). Below the header, there is a large white space containing text and icons.

Left Screenshot:

- Text:** "Wij zijn **DE VOORHOEDE**
We bouwen front-end oplossingen
waarmee je jaren vooruit kunt"
- Icon:** A smartphone icon.
- Section:** "Dit kunnen we voor je doen"
 - Icon:** A computer monitor and a smartphone icon.
 - Text:** "Snelle en responsive websites"
 - Icon:** A complex network diagram with nodes and arrows.
 - Text:** "Complexe webapplicaties"

Right Screenshot:

- Text:** "Wij zijn **DE VOORHOEDE**
We bouwen front-end oplossingen
waarmee je jaren vooruit kunt"
- Icon:** A smartphone icon.
- Section:** "Dit kunnen we voor je doen"
 - Text:** "We geloven in universele oplossingen die door iedereen te gebruiken zijn. Dit bereiken we door webtoepassingen te maken die altijd en overal werken op elk apparaat."

[Critical CSS on voorhoede.nl](http://voorhoede.nl)

CONDITIONAL STYLE LOADER

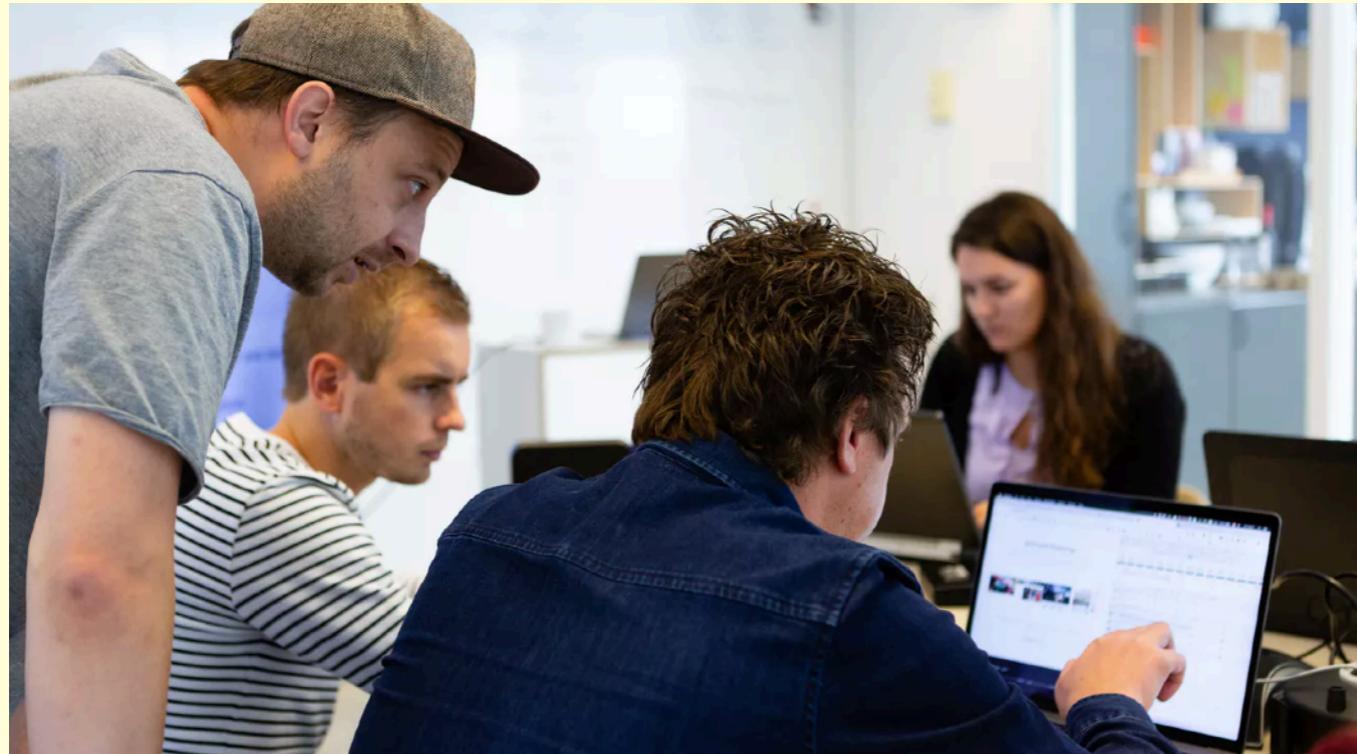
```
<html><head><style>/*critical CSS*/</style><script>
// (include loadCSS & onloadCSS here)
var stylesheet = loadCSS('index-f3c2b4.css');
onloadCSS(stylesheet, function() {
    setCookie('full-css-loaded', 'f3c2b4');
});
```

On server, if 'full-css-loaded' cookie and version is equal:

```
<link rel="stylesheet" href="index-f3c2b4.css">
```

Console						Sources	Network	Timeline	Profiles	Resources	Security	Audits	PageSpeed
Name	Value	Domain	Path	Expires / Max-Age									
fonts-loaded	true	www.voorhoede.nl	/	2016-10-21T13:01:32.000Z									
full-css-loaded	/assets/css/main-0d82fe89fd.css	www.voorhoede.nl	/	2016-10-21T13:01:32.000Z									

[cookies on voorhoede.nl](#)



thanks

Declan Rek

// @DeclanRek

// voorhoede.nl

