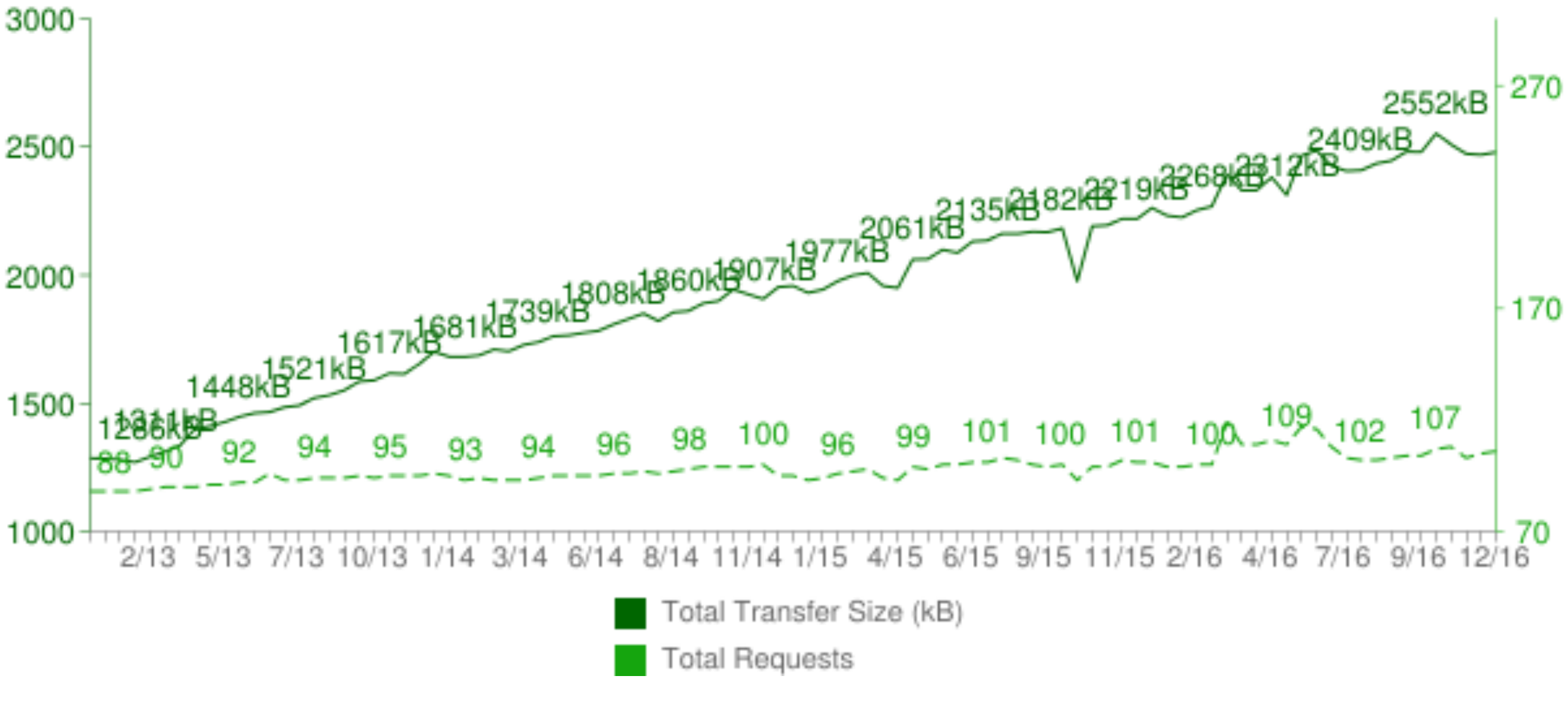
De Voorhoede

Critical Rendering Path

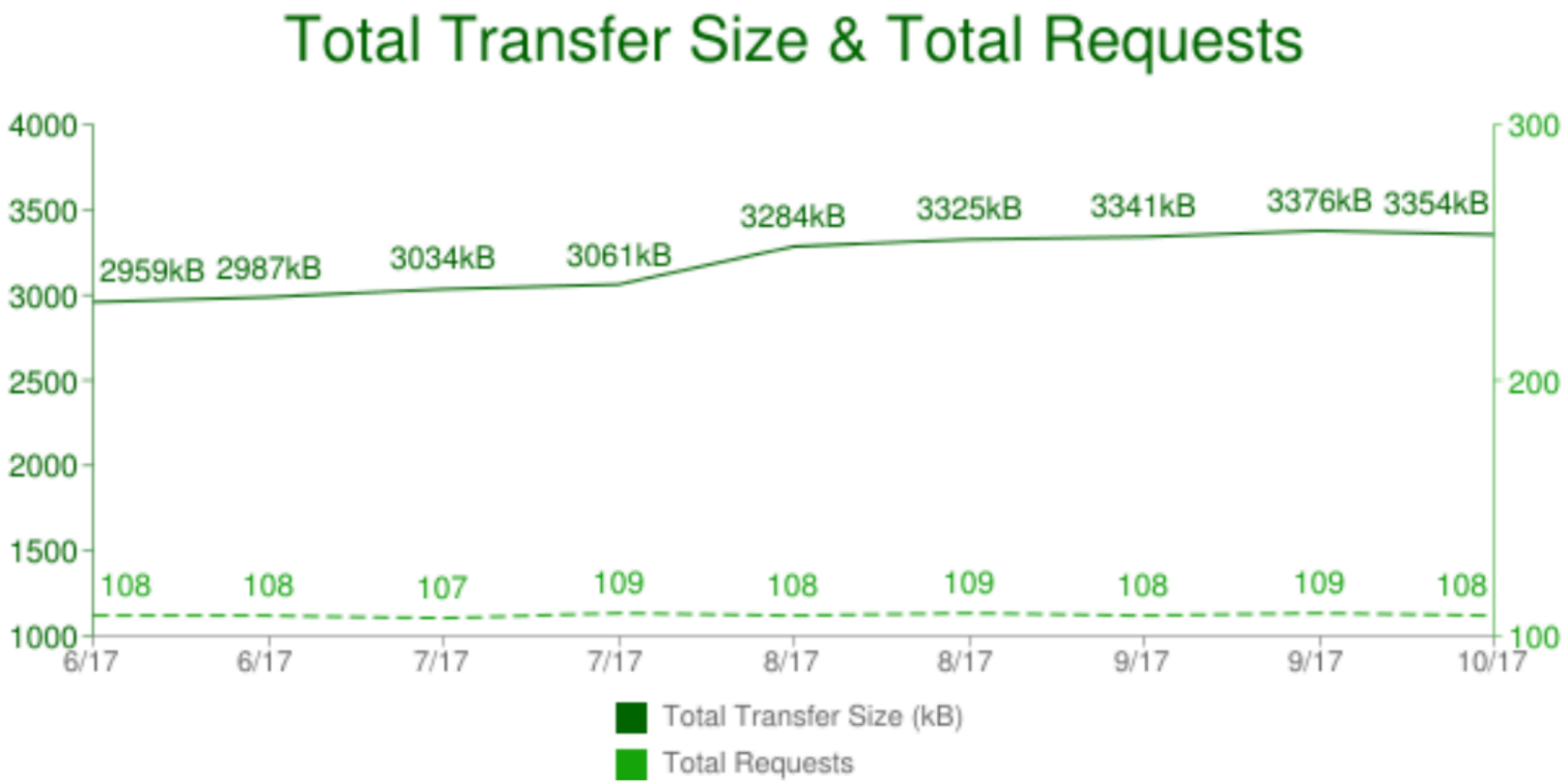
Declan Rek

PERFORMANCE MATTERS

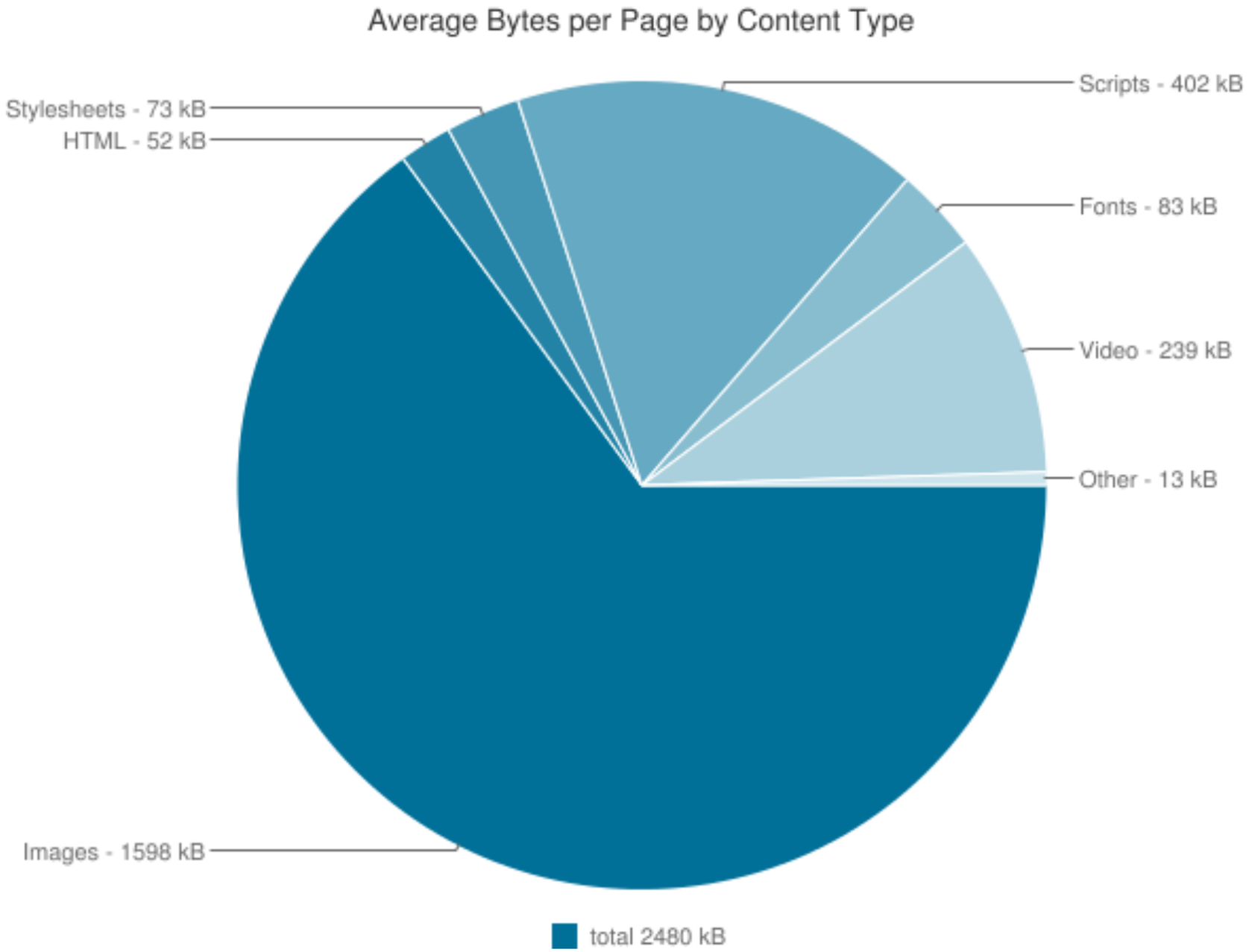
Total Transfer Size & Total Requests



HTTPArchive

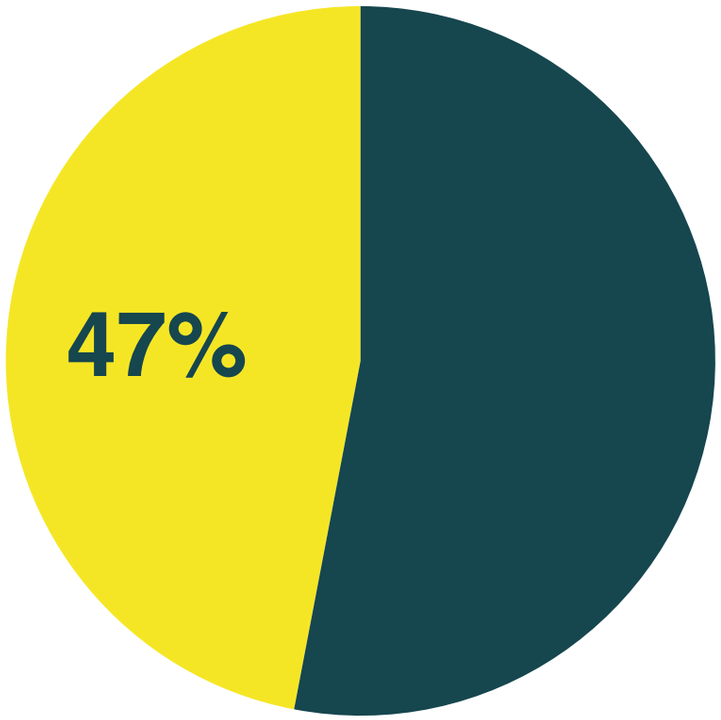


HTTPArchive

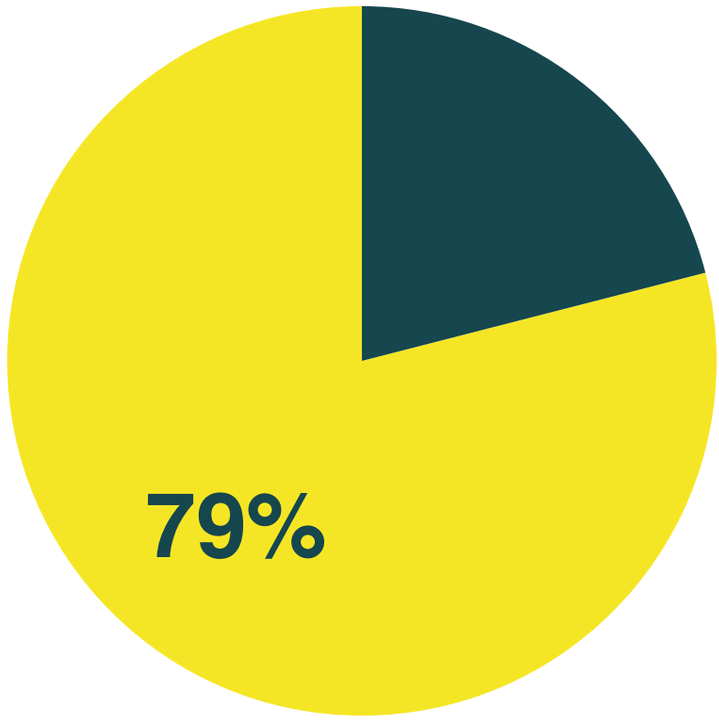


HTTPArchive

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

KissMetrics

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

wpostats.com

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.

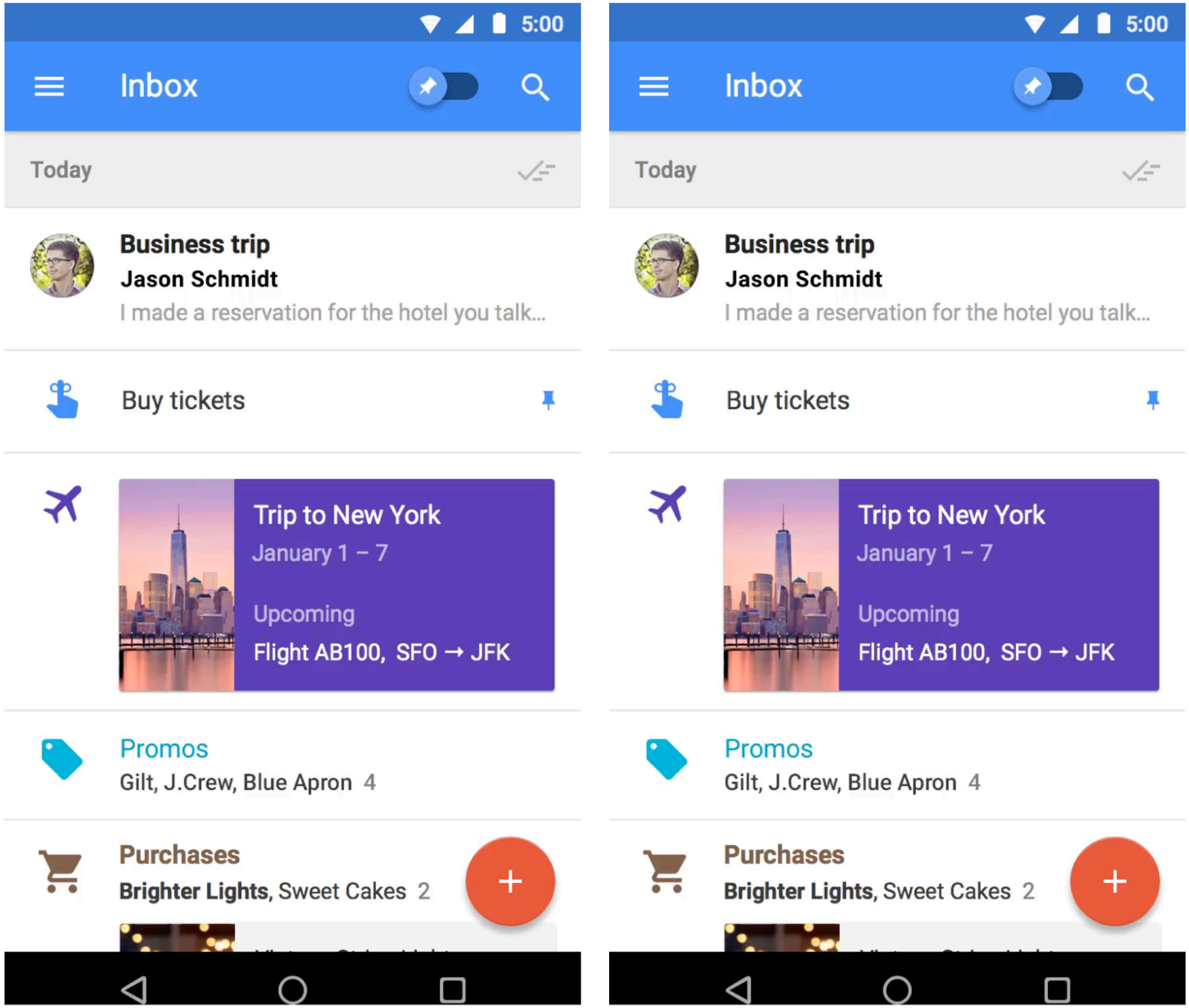
Initial Content

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?

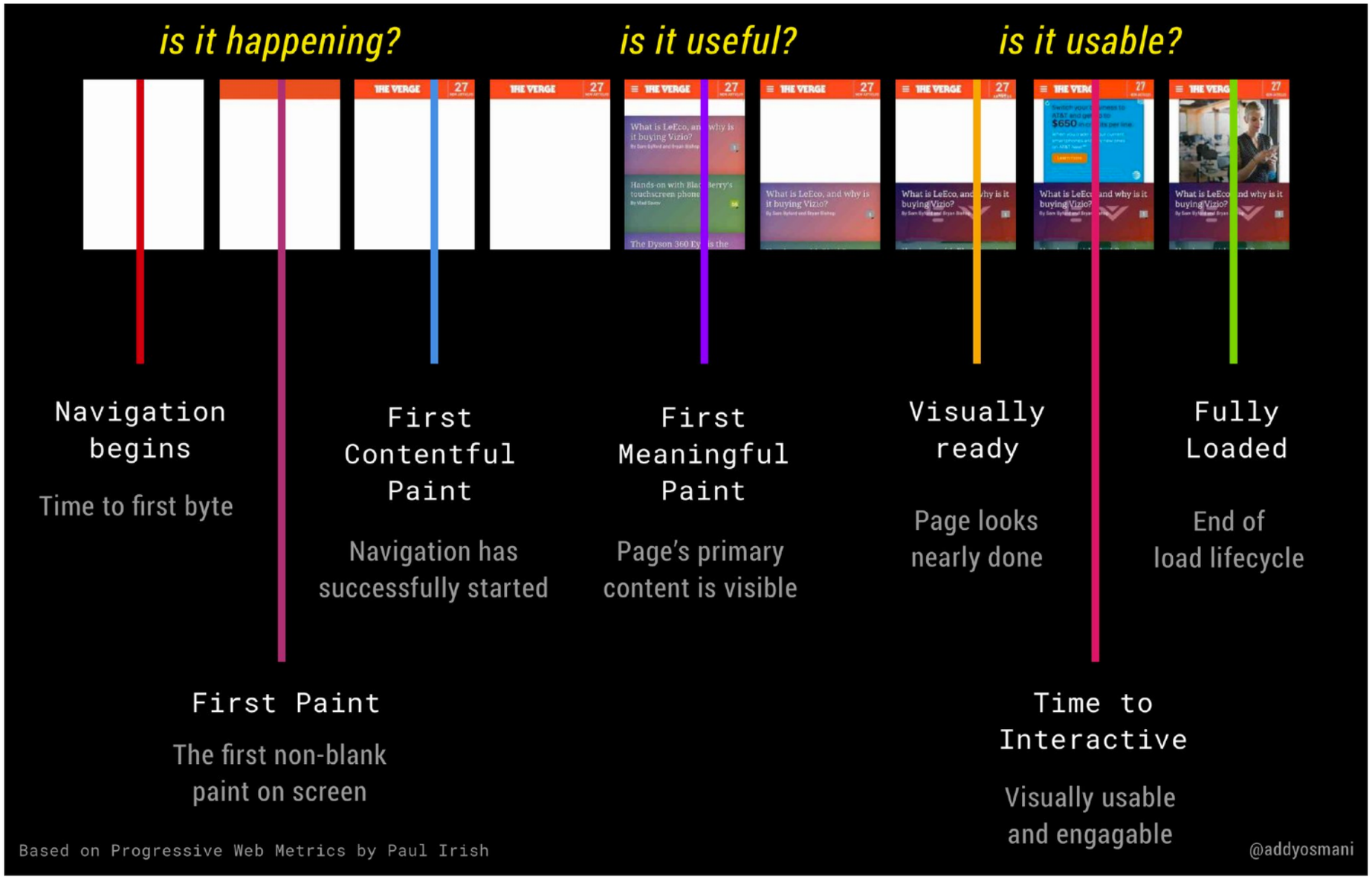
100ms = Instantaneous

1 second = Uninterrupted flow

Response Times: The 3 Important Limits -Jakob Nielsen

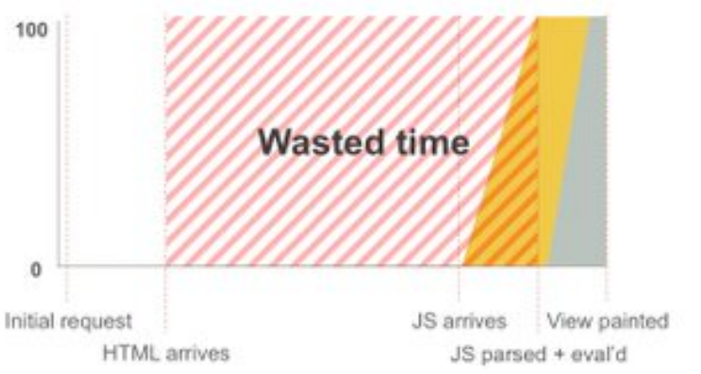
CRITICAL RENDERING PATH

RENDERING MOMENTS



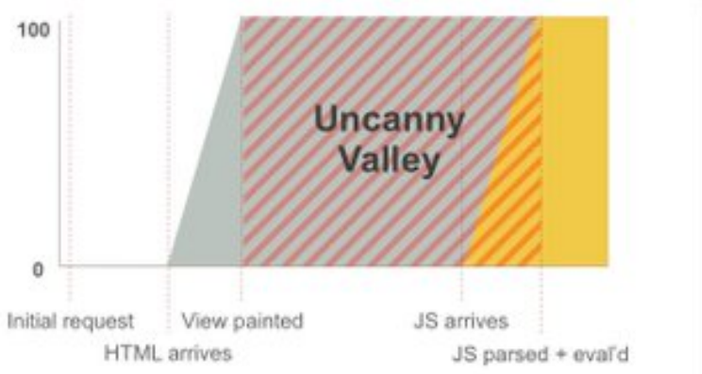
@addyosmani, Aug 2, 2016

PROGRESSIVE RENDERING



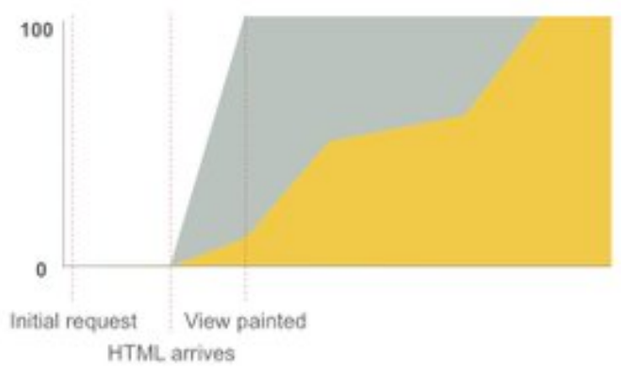
JS-based Render

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.



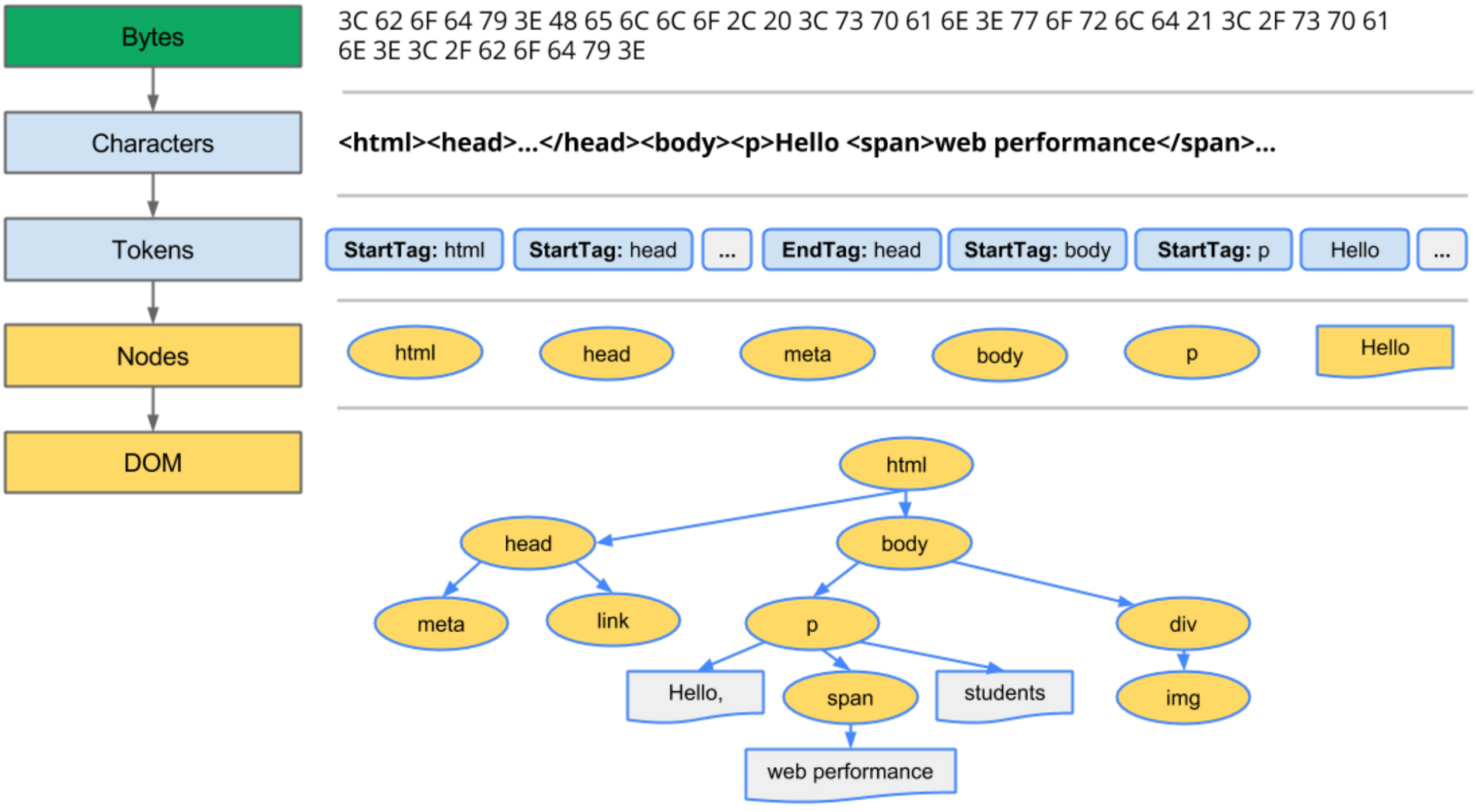
Progressive render + bootstrap

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.

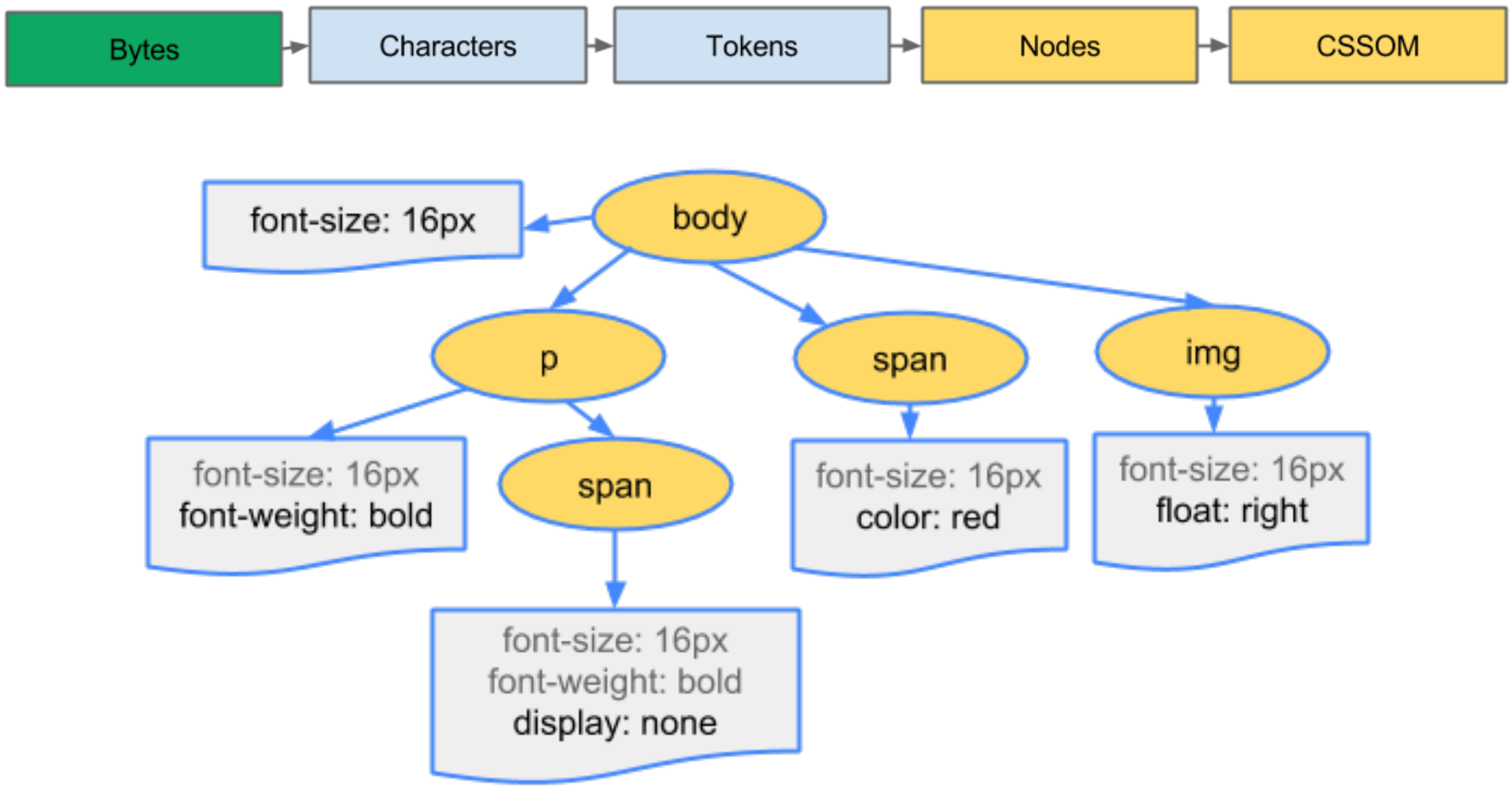
@aerotwist, May 9, 2016

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

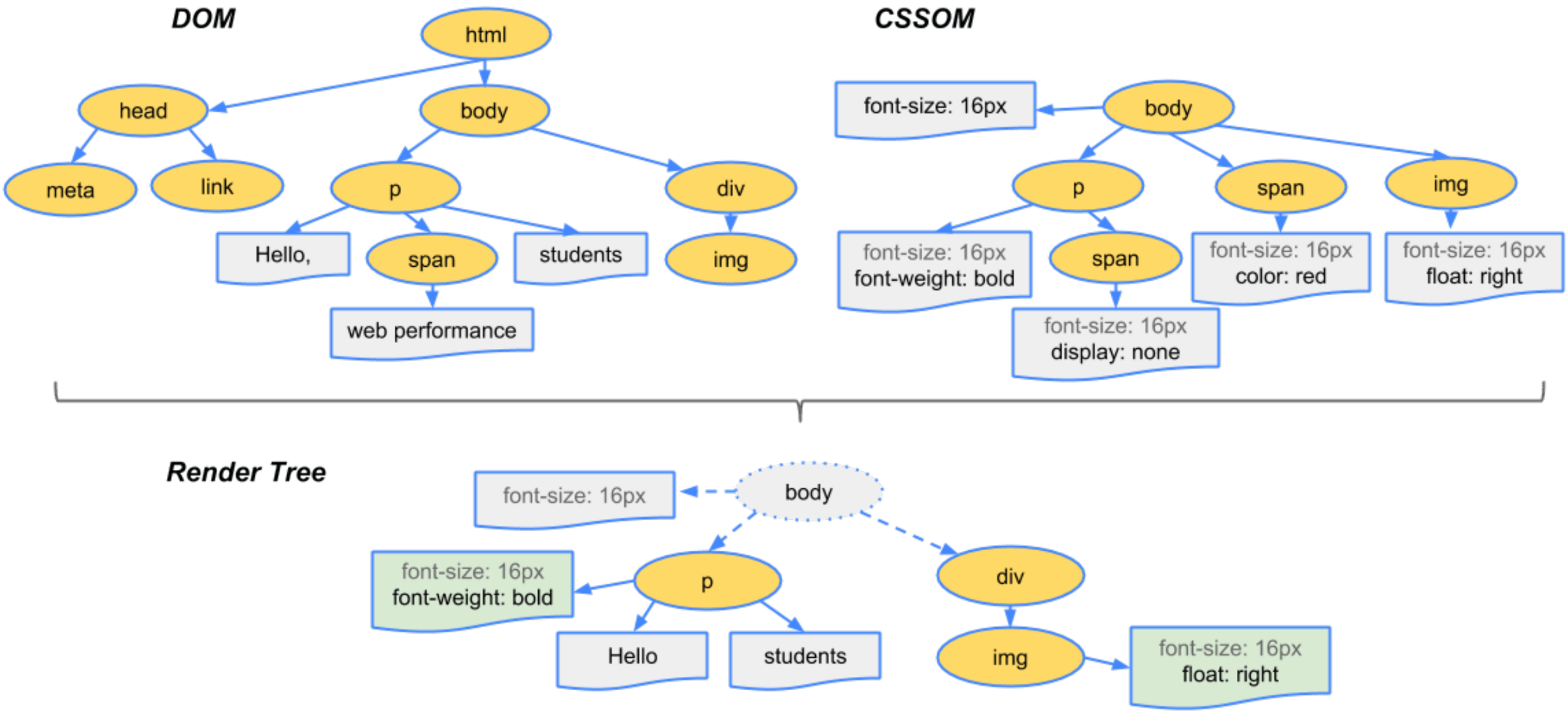
DOM


Bytes
Characters
Tokens
Nodes
DOM
3C 62 6F 64 79 3E 48 65 6C 6C 6F 2C 20 3C 73 70 61 6E 3E 77 6F 72 6C 64 21 3C 2F 73 70 61 6E 3E 3C 2F 62 6F 64 79 3E
<html><head>...</head><body><p>Hello web performance...
StartTag: html
StartTag: head
...
EndTag: head
StartTag: body
StartTag: p
Hello
...
html
head
meta
body
p
Hello
html
head
body
meta
link
p
div
Hello,
span
students
img
web performance

CSSOM


Bytes
Characters
Tokens
Nodes
CSSOM
font-size: 16px
body
p
span
img
font-size: 16px
font-weight: bold
span
font-size: 16px
color: red
font-size: 16px
float: right
font-size: 16px
font-weight: bold
display: none

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)

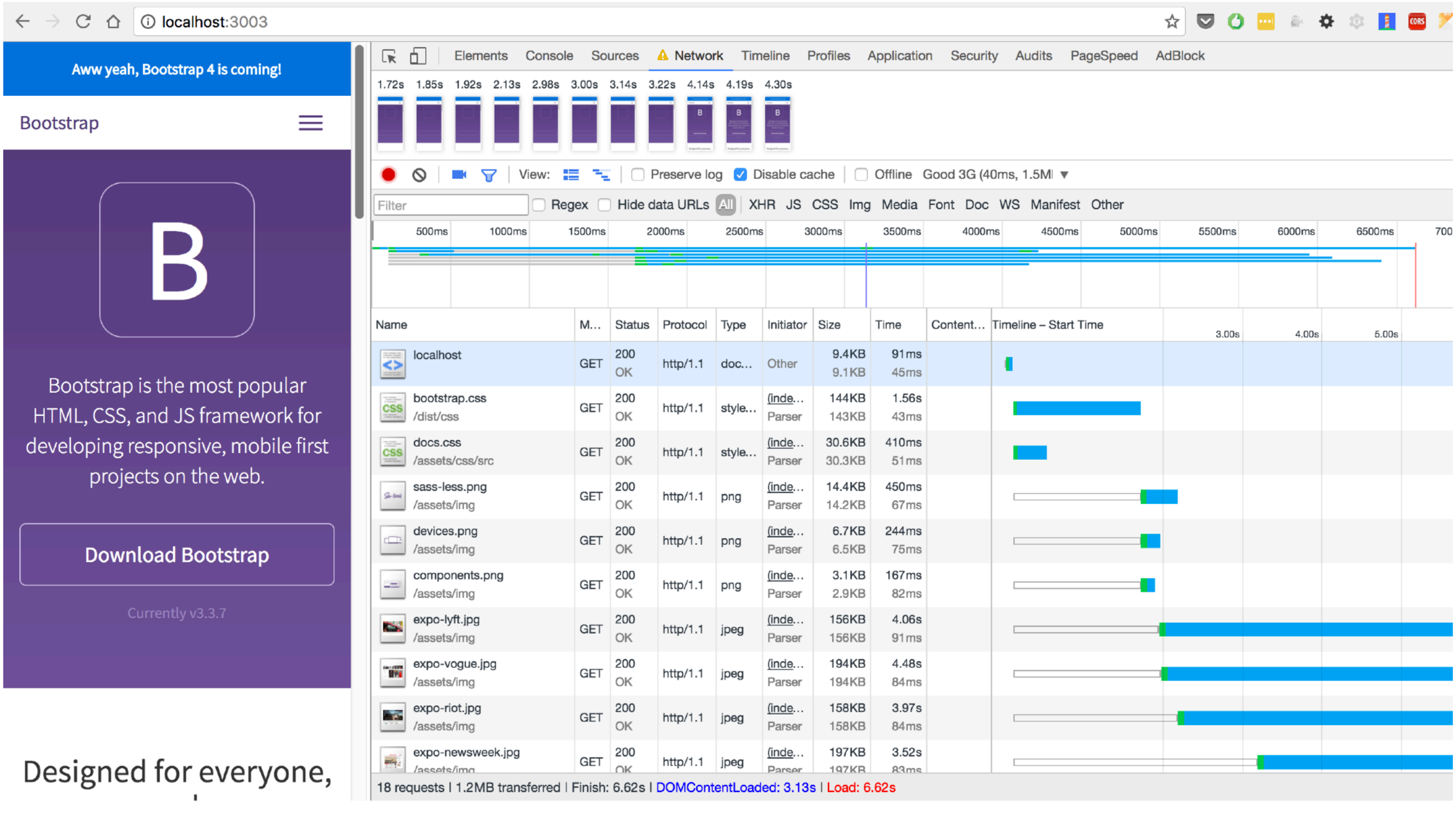
How fast is the primary content available

Time to interactive(TTI)

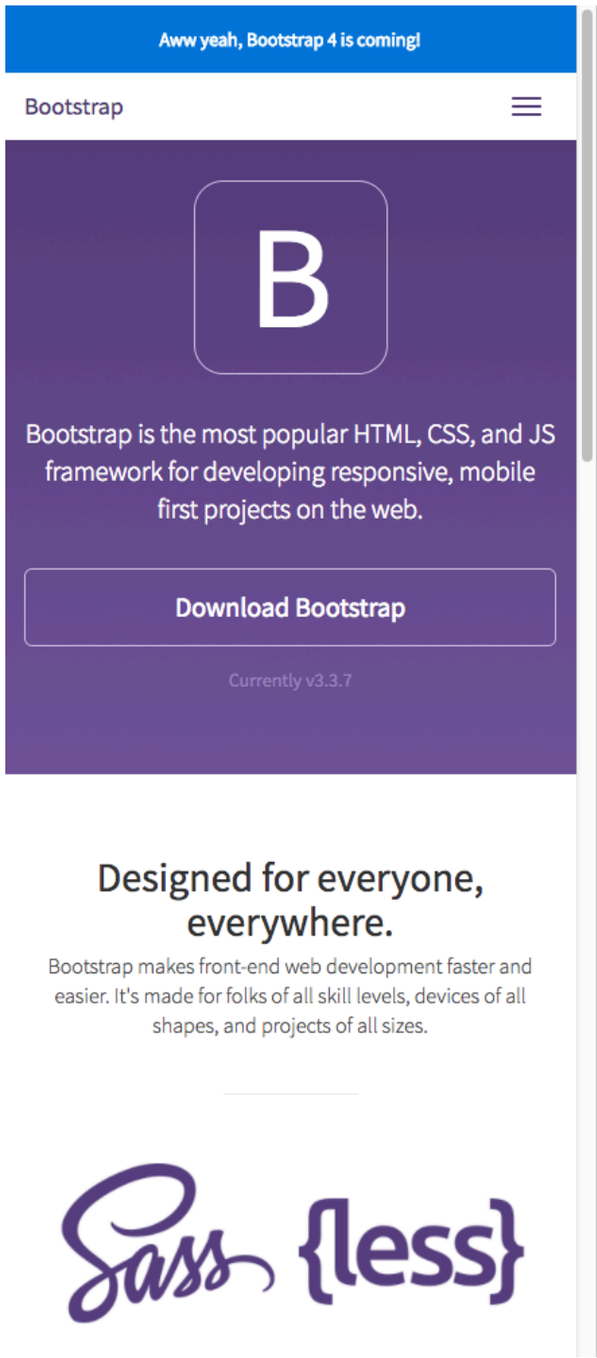
How long does it take for a page to be interactive

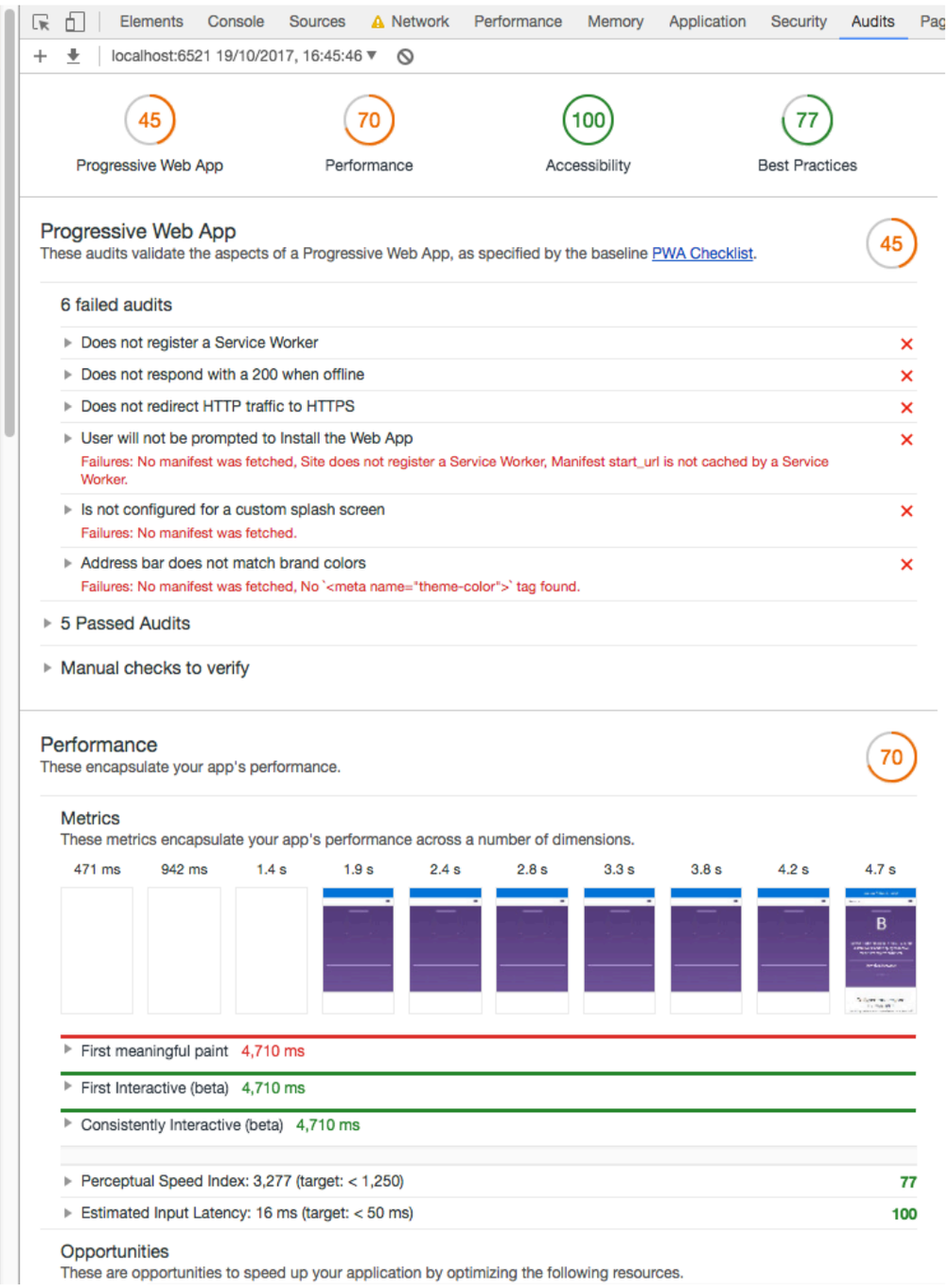
TOOLING

CHROME DEV TOOLS

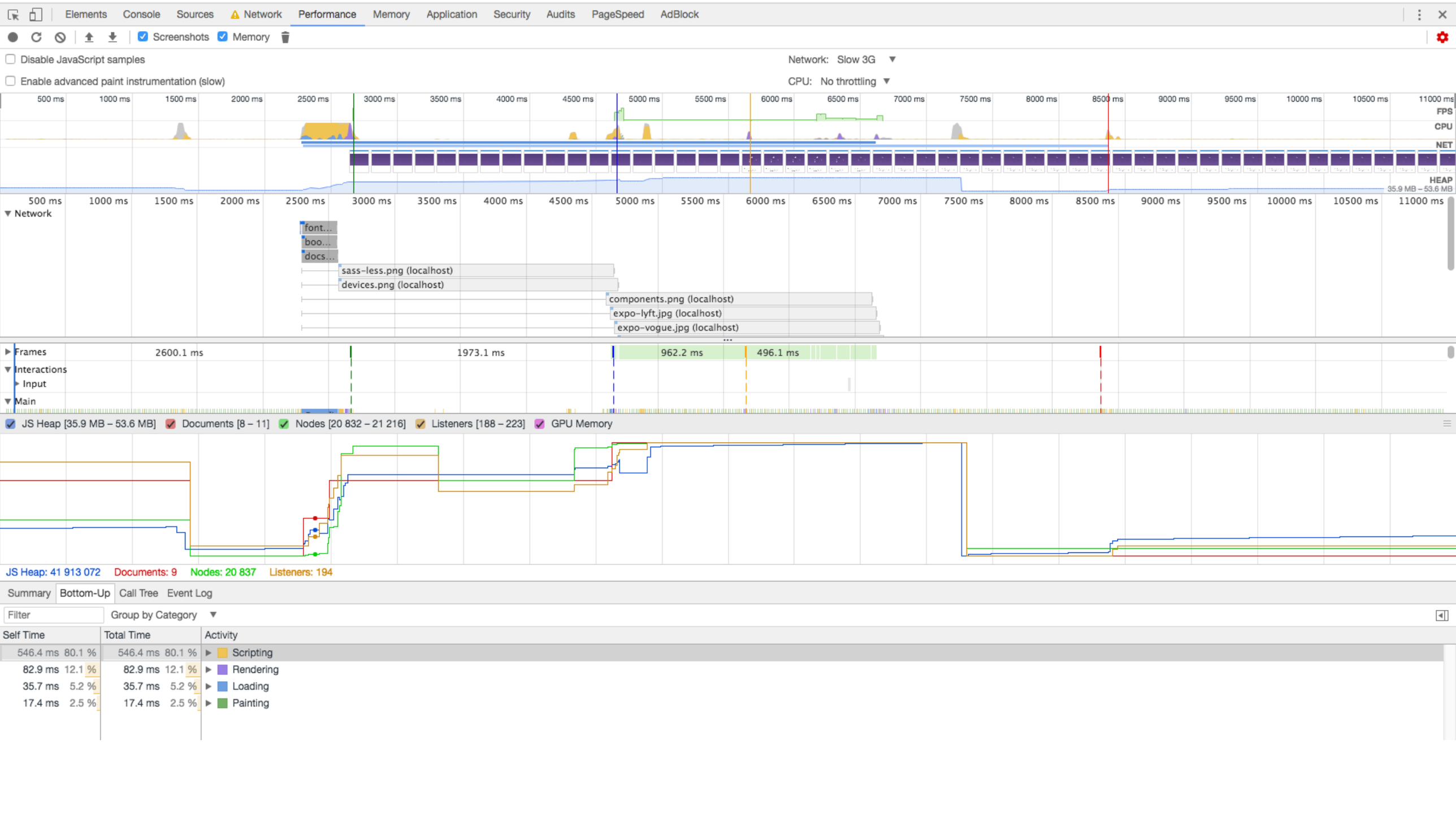


CHROME AUDITS





CHROME PERFORMANCE



WEBPAGETEST

Need help improving?

Web Page Performance Test for

getbootstrap.com

A	A	A	F	F	✓
First Byte Time	Keep-alive Enabled	Compress Transfer	Compress Images	Cache static content	Effective use of CDN

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

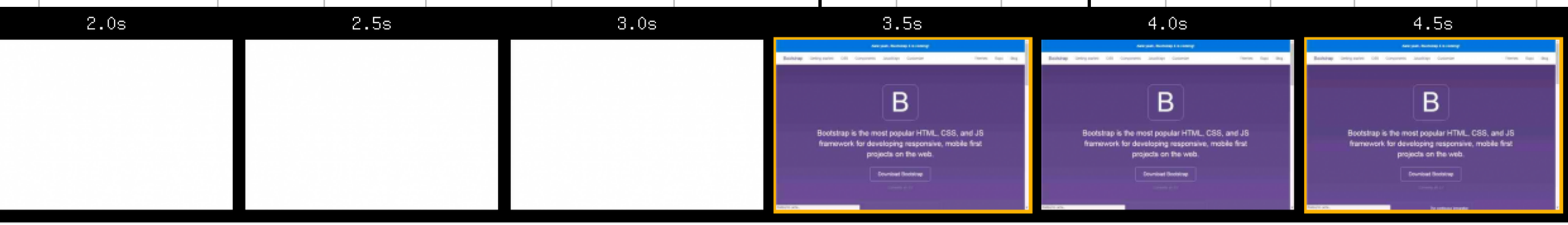
Summary | **Details** | **Performance Review** | **Content Breakdown** | **Domains** | **Processing Breakdown** | **Screen Shot**

Tester: VPS16230-93.191.133.233
Test runs: **5**
Re-run the test

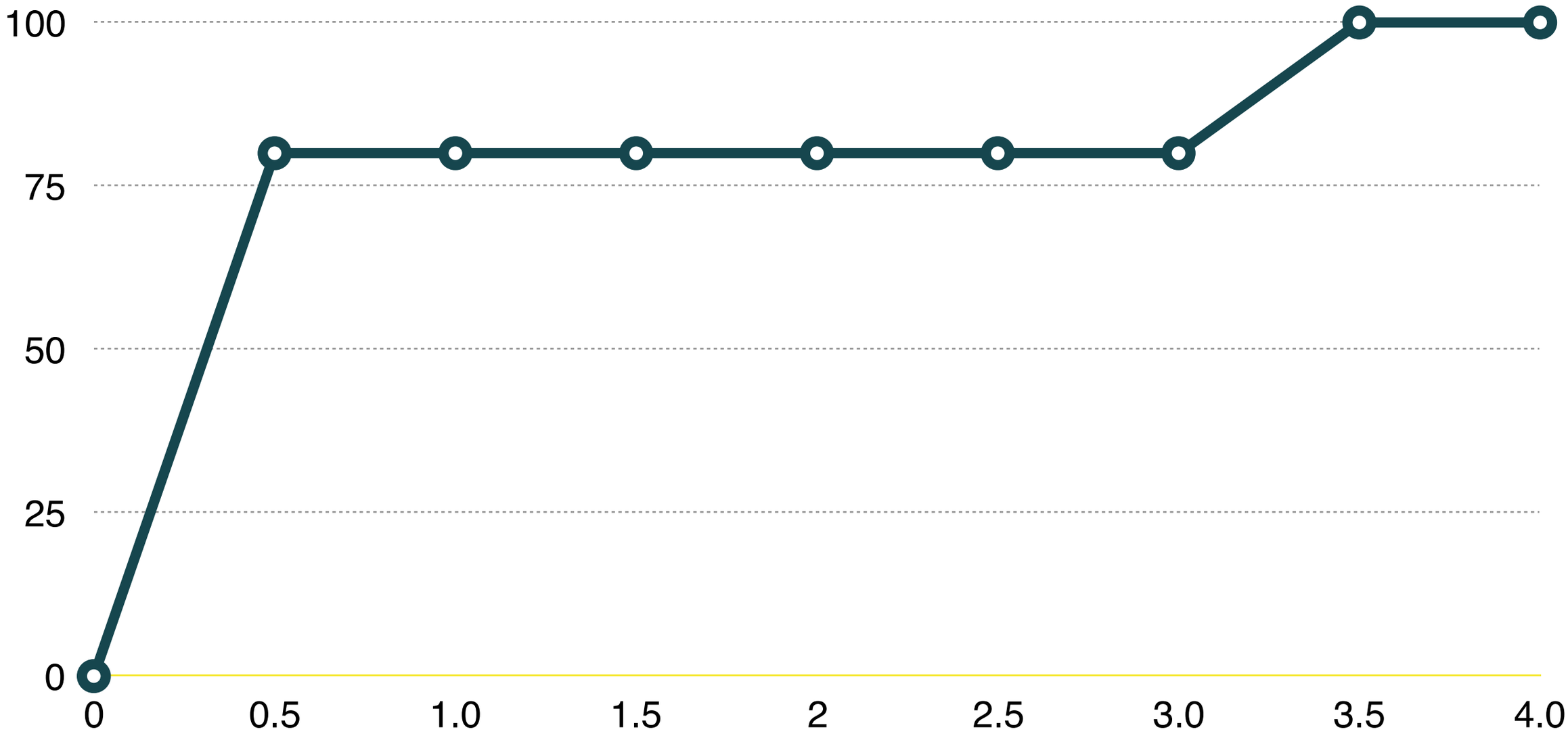
Raw page data - Raw object data
Export HTTP Archive (.har)
View Test Log

Performance Results (Median Run)

						Document Complete			Fully Loaded				
	Load Time	First Byte	Start Render	Speed Index	DOM Elements	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	



SPEED INDEX



Latest SRP

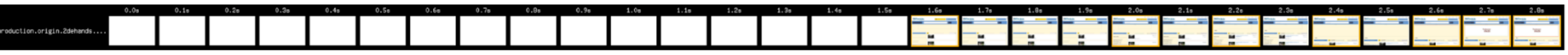
PAINT
1.6 Sec.

USABLE PAGE
2.2 Seconds

Min + gzip

PAINT
1.6 Sec.

USABLE PAGE
2.0 Seconds



CriticalCSS + min + gzip

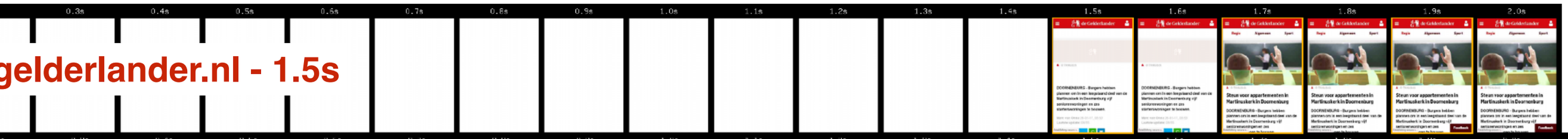
PAINT
0.6 Sec.

USABLE PAGE
1.3 Seconds

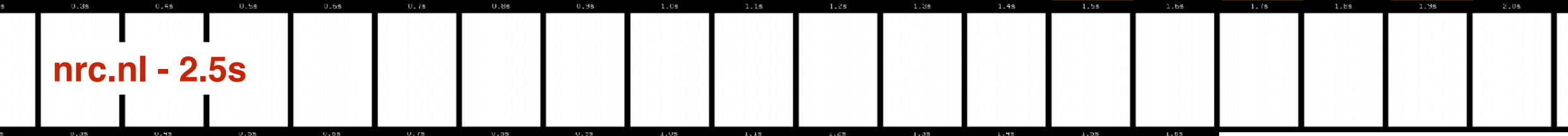
Critical + FOIT + min + gzip

PAINT and USABLE PAGE
0.7 Sec.

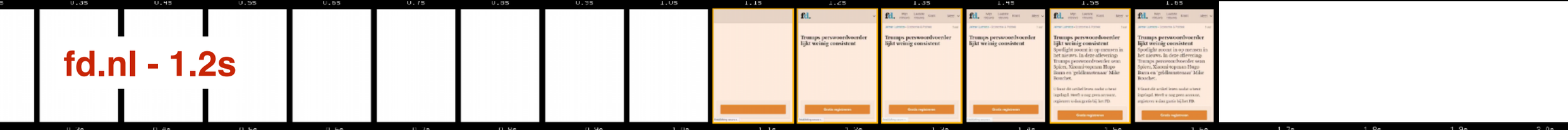
gelderlander.nl - 1.5s



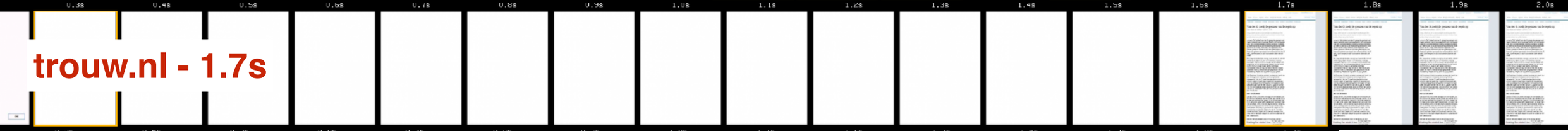
nrc.nl - 2.5s



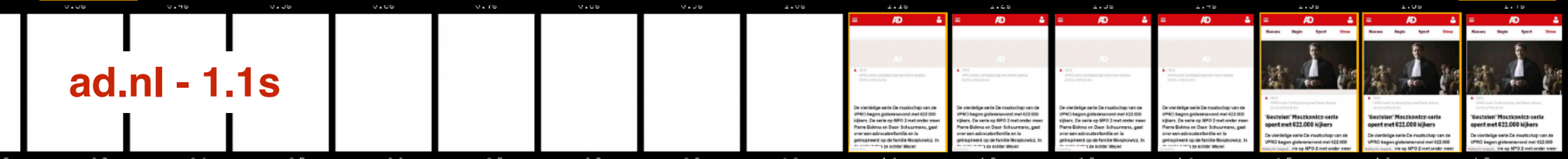
fd.nl - 1.2s



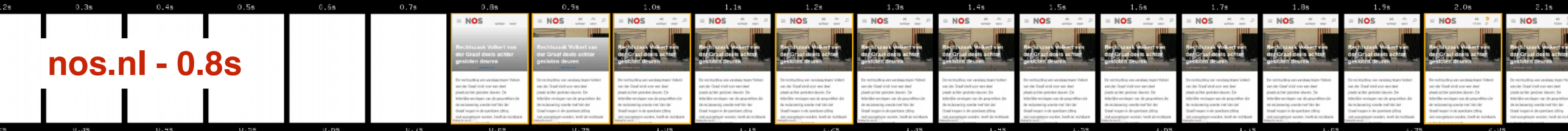
trouw.nl - 1.7s



ad.nl - 1.1s

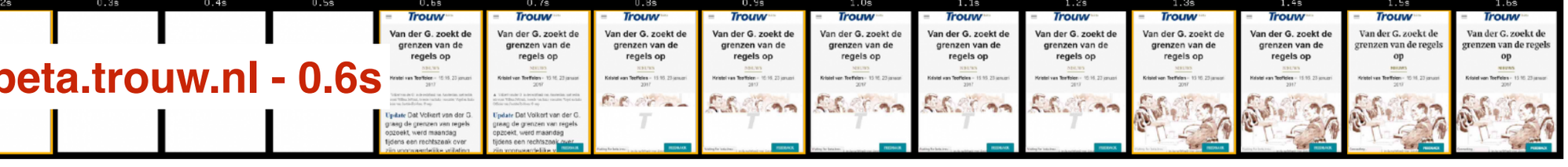


nos.nl - 0.8s



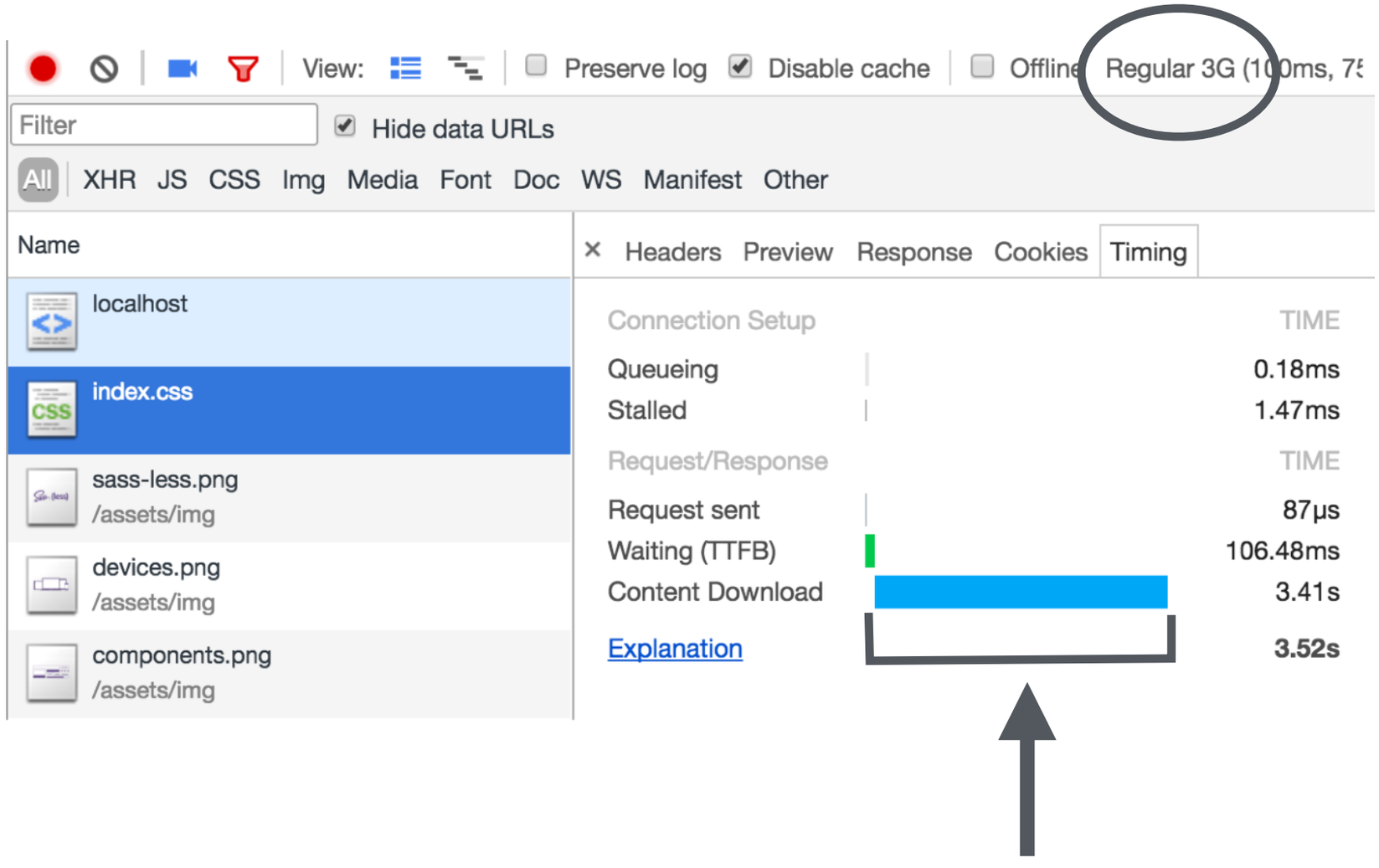
beta.hln.be - 0.7s

beta.trouw.nl - 0.6s



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/

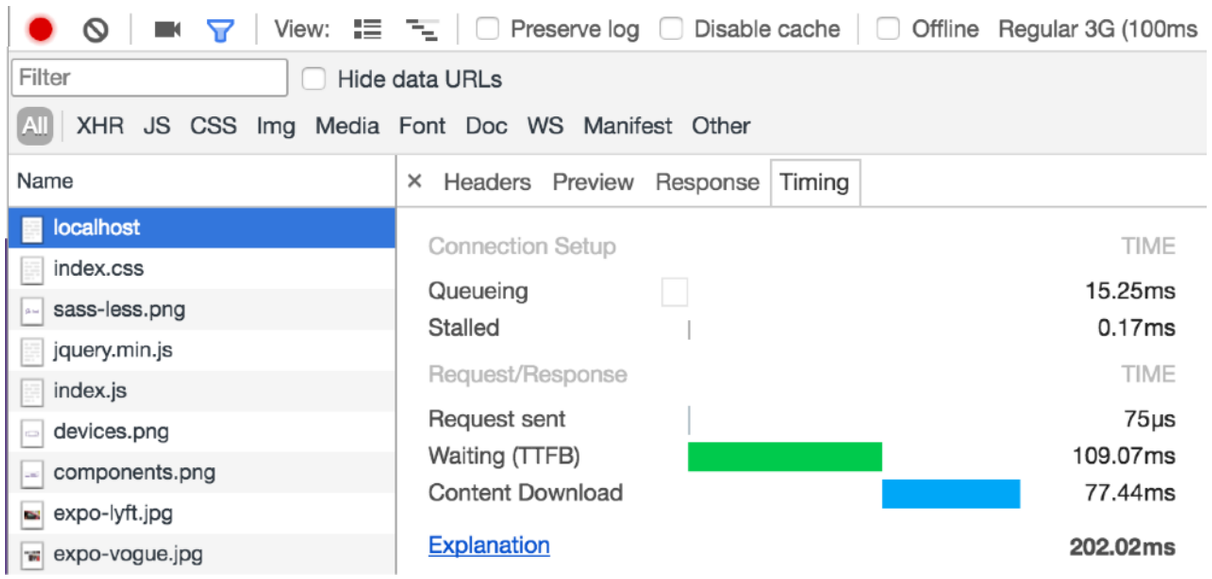
HTML AFTER

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

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

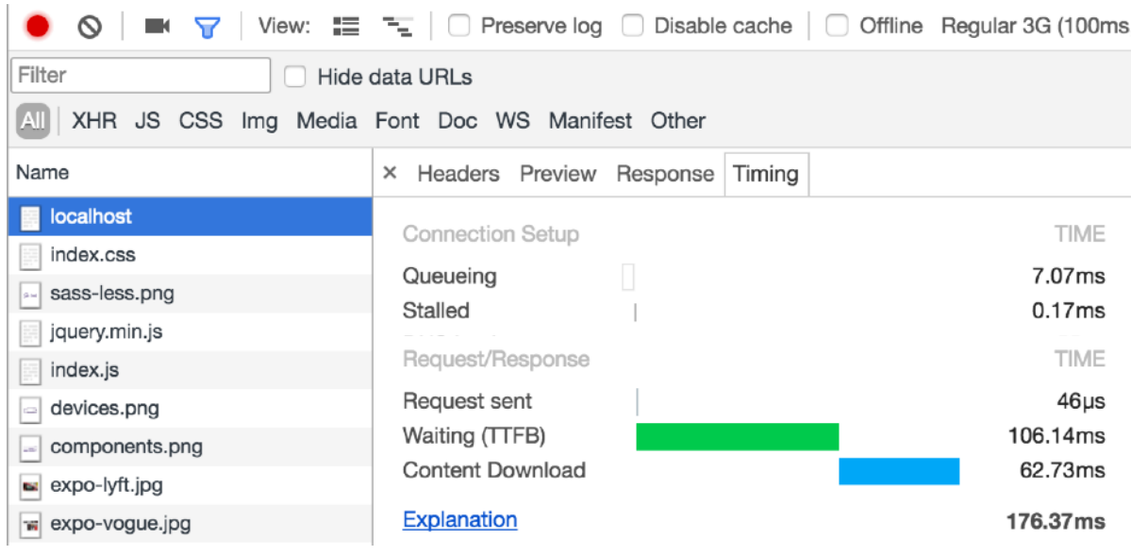
COMPARE: HTML

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

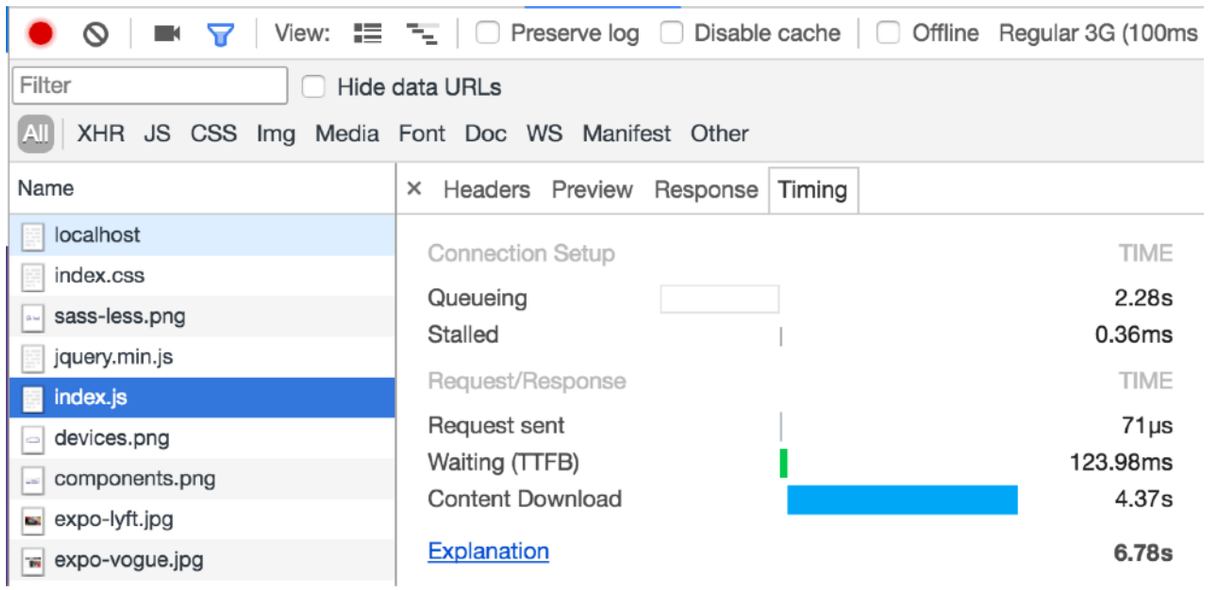
```
function foo(n,o){return n+o}var x=42;
```

demos.forbeslindesay.co.uk/uglify-js

alistapart.com/article/better-javascript-minification

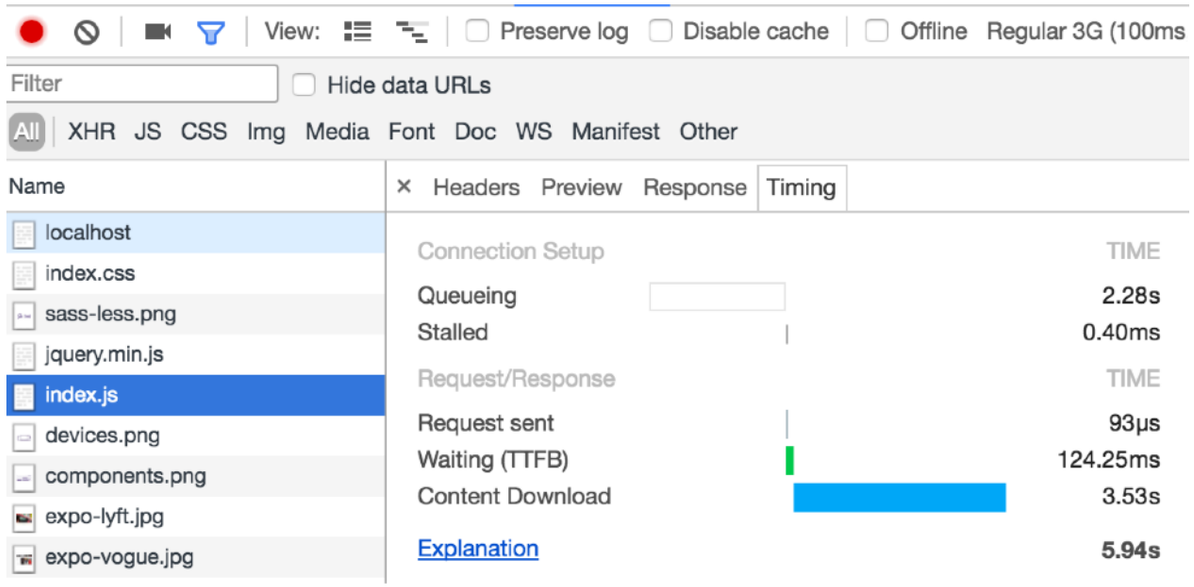
COMPARE: JS

BEFORE



115KB ~6.8 sec

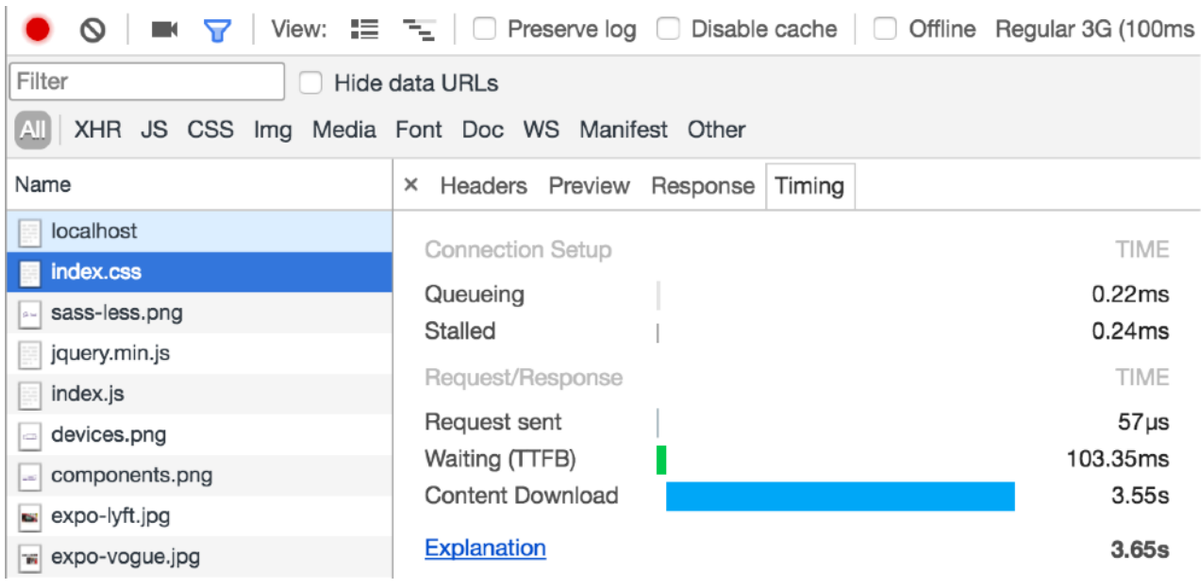
AFTER



81.8KB ~5.9 sec

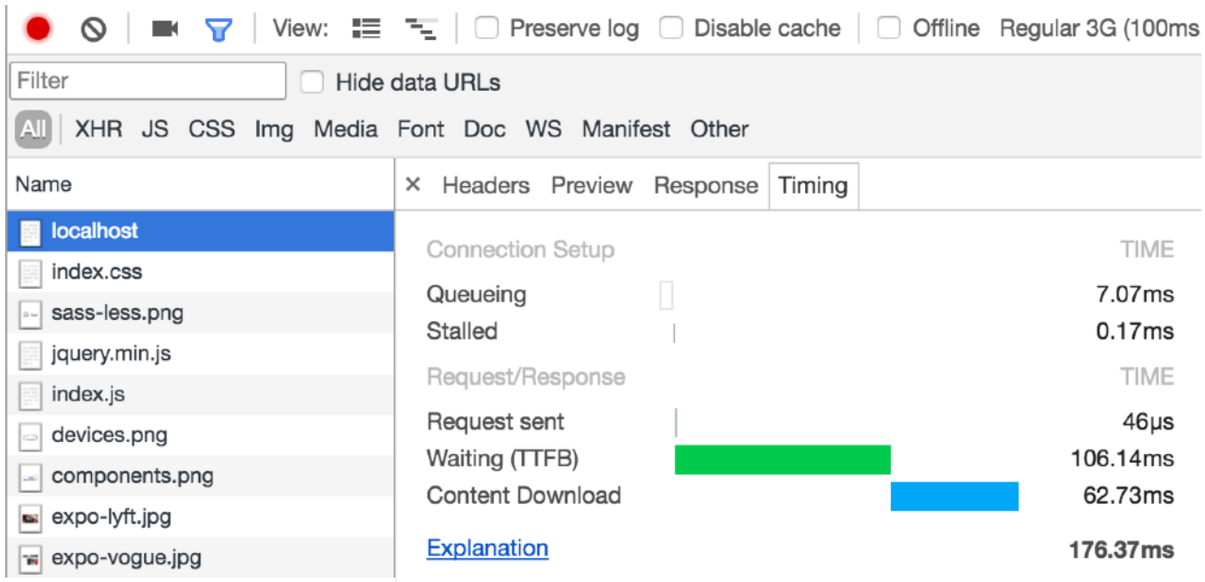
COMPARE: CSS

BEFORE



166KB ~3.6 sec

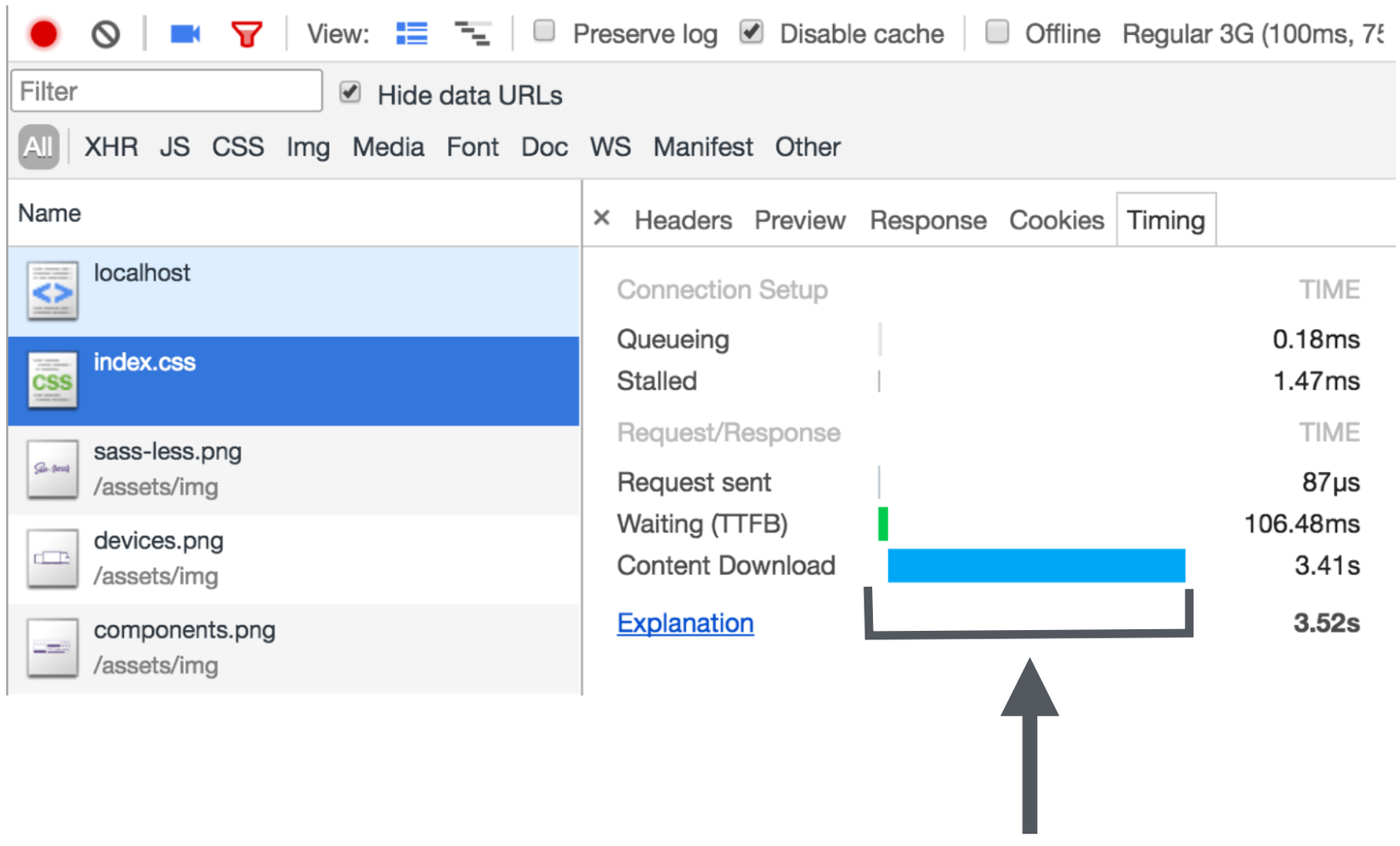
AFTER



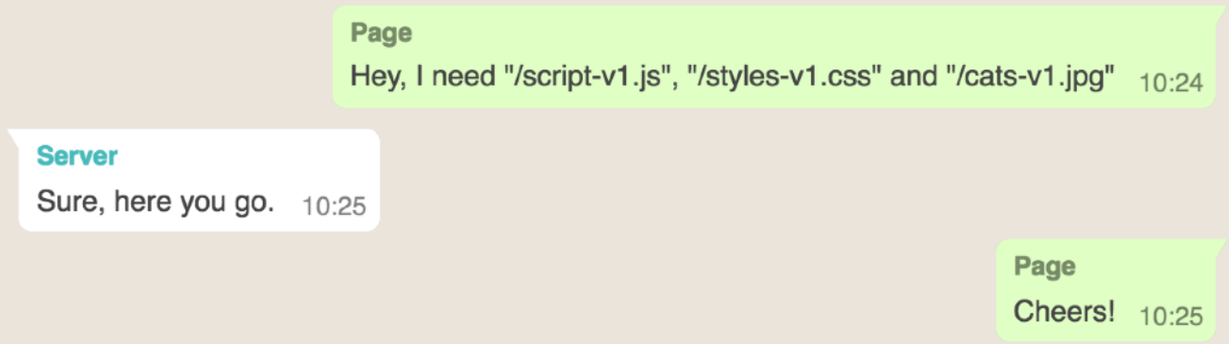
132KB ~2.9 sec

HTTP CACHING

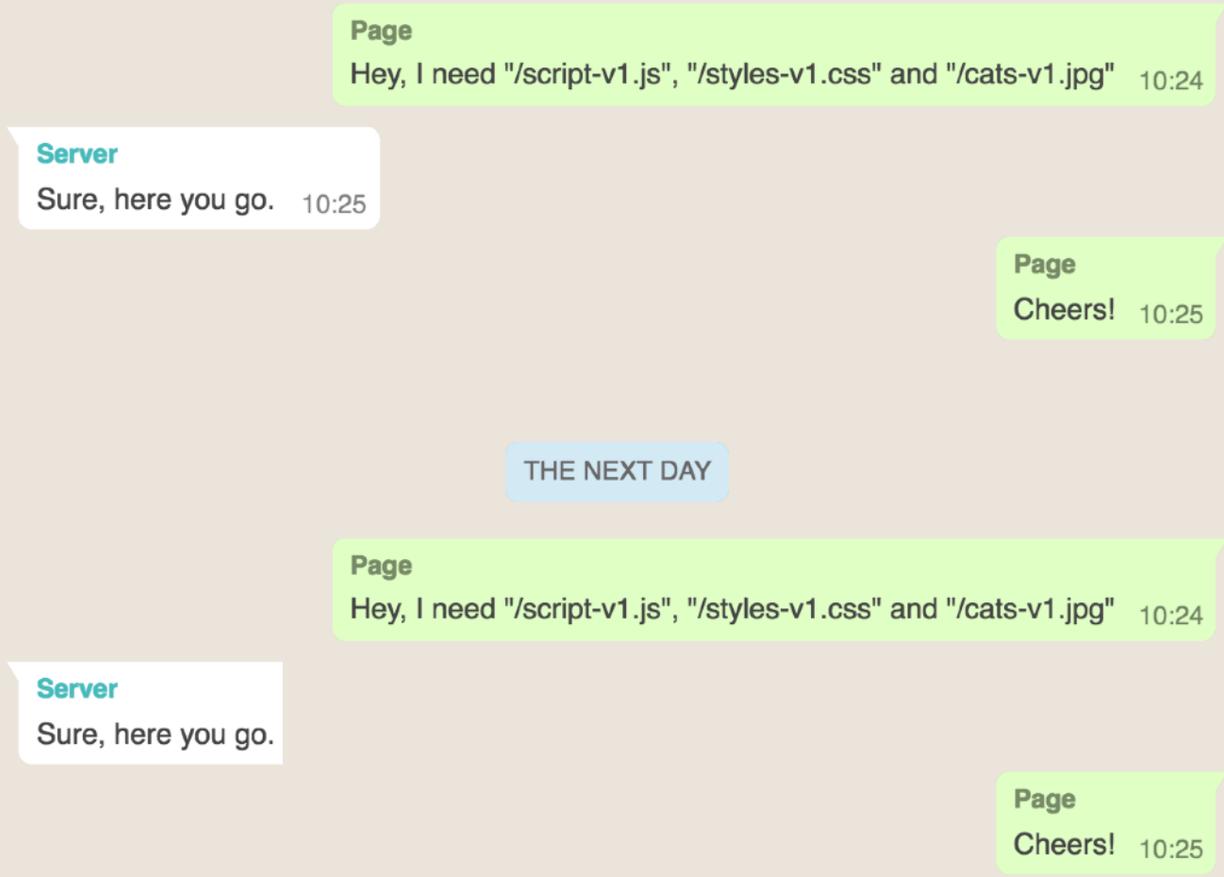
PROBLEM: NO CACHE



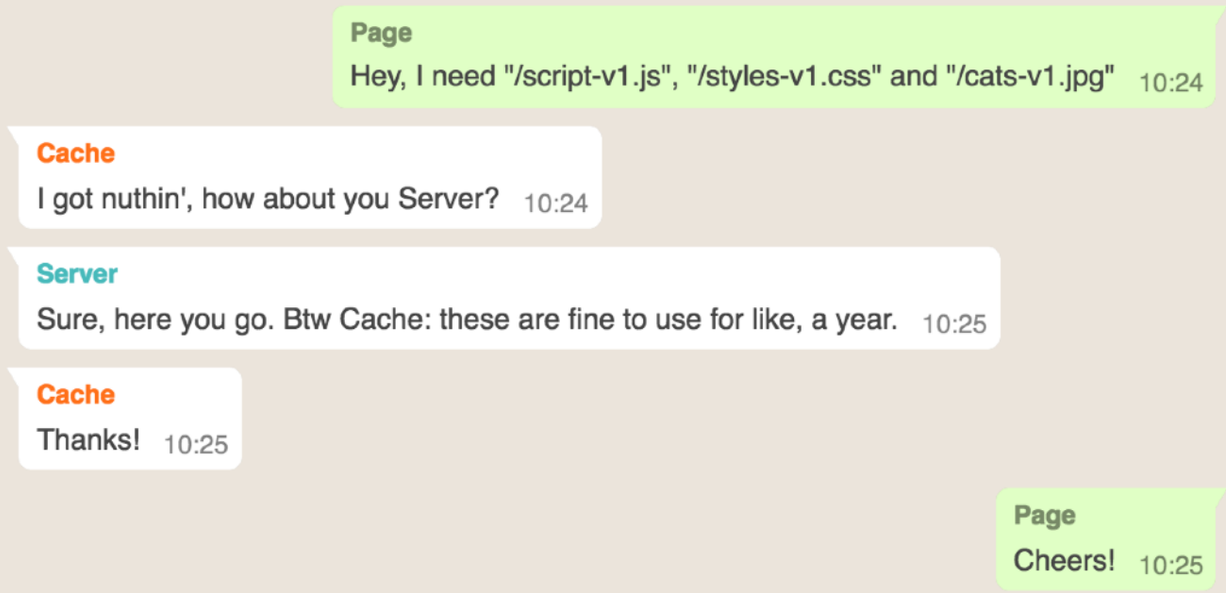
connect, wait and download



adapted from
jakearchibald.com/2016/
caching-best-practices/



adapted from
jakearchibald.com/2016/
caching-best-practices/



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

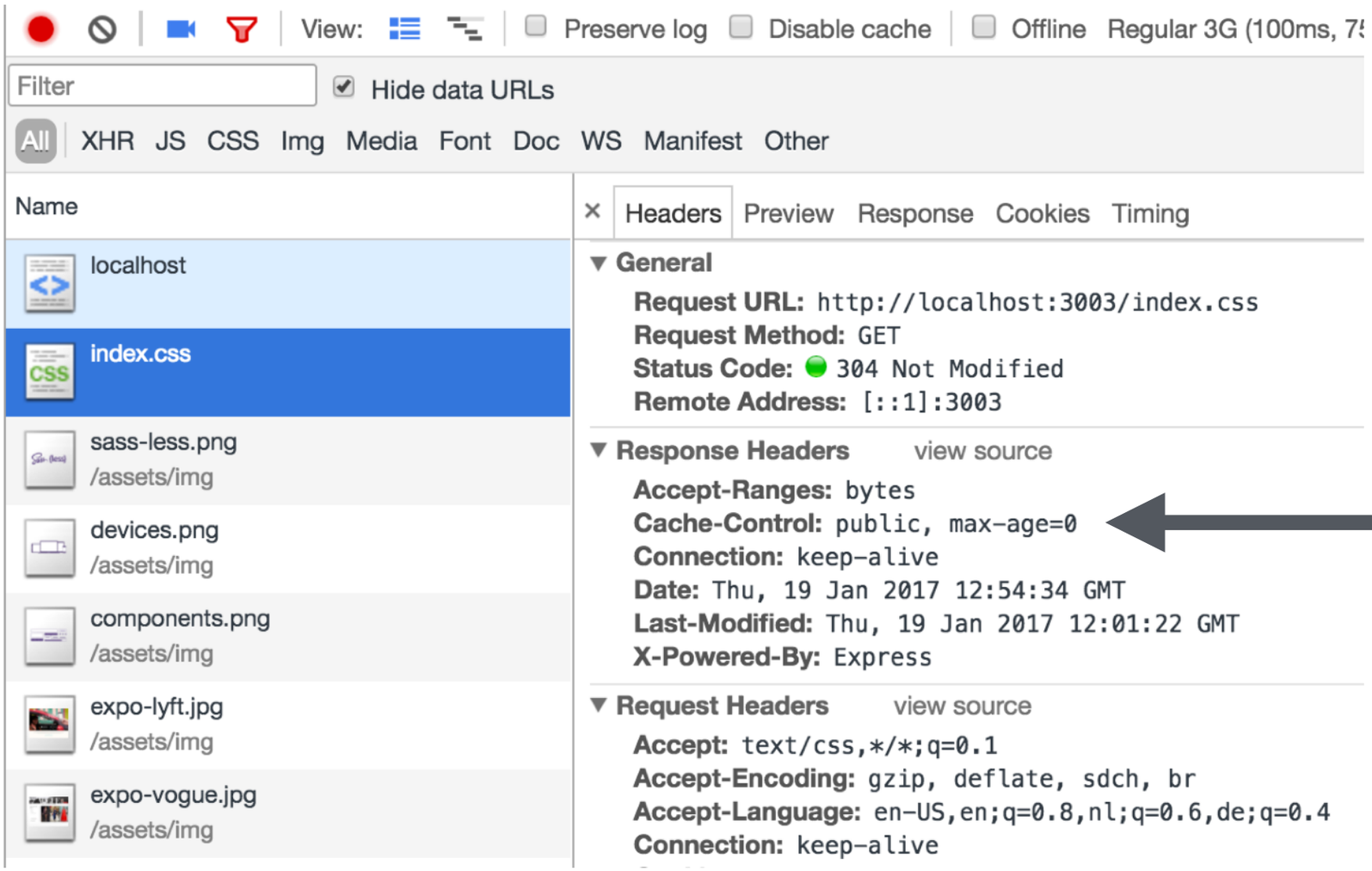
Page
Thanks! 08:15

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/

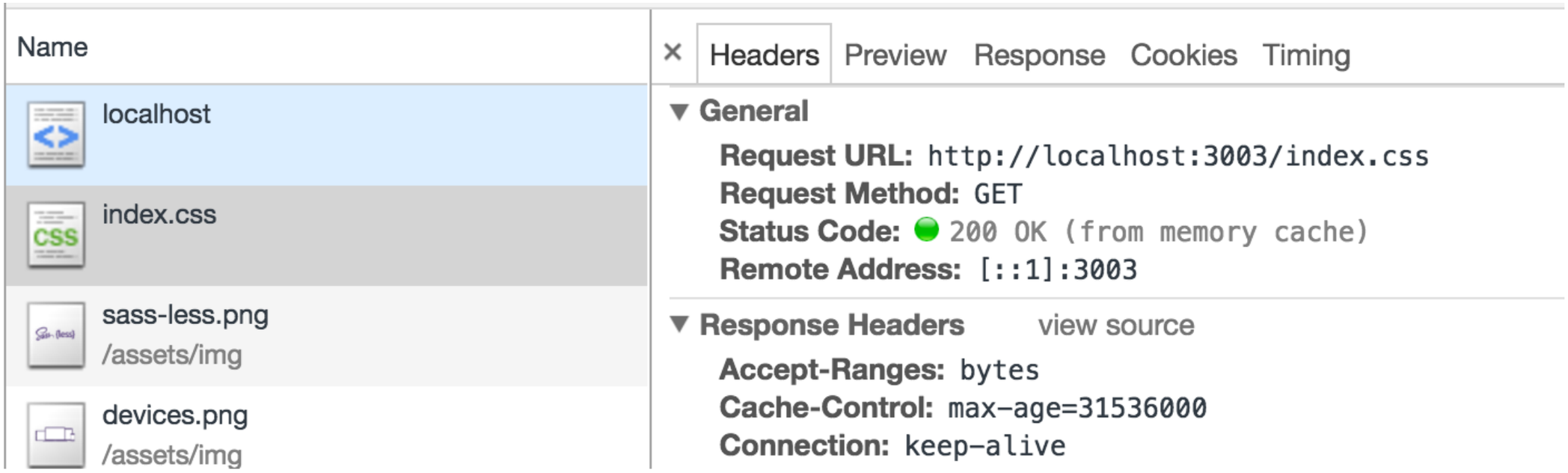
CACHE CONTROL



no caching

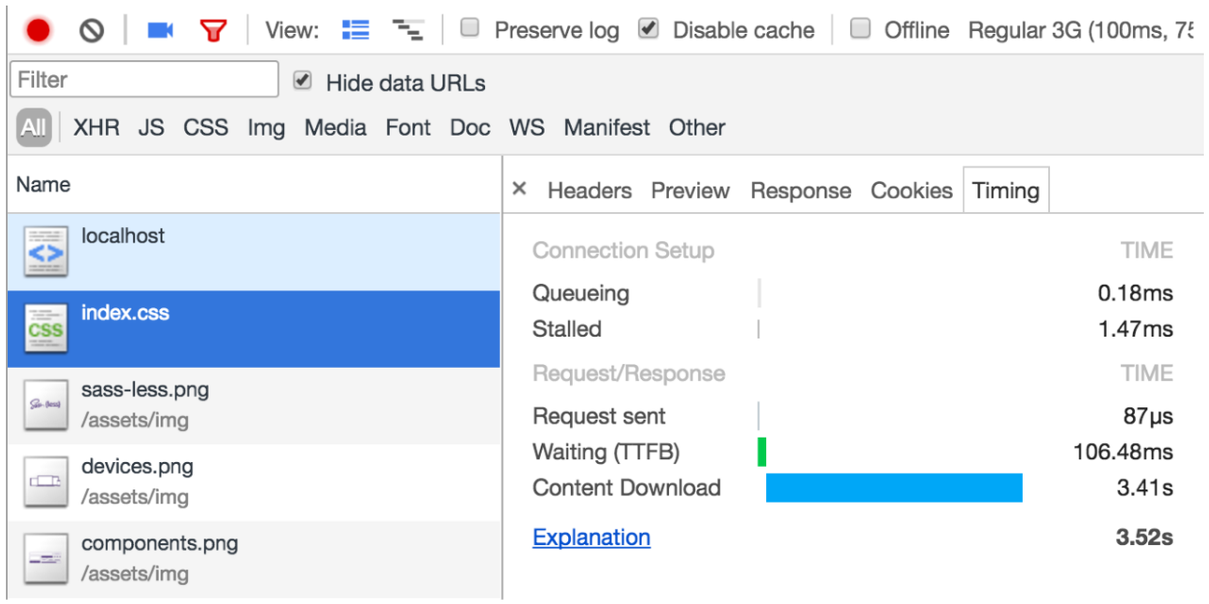
SET CACHE CONTROL

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



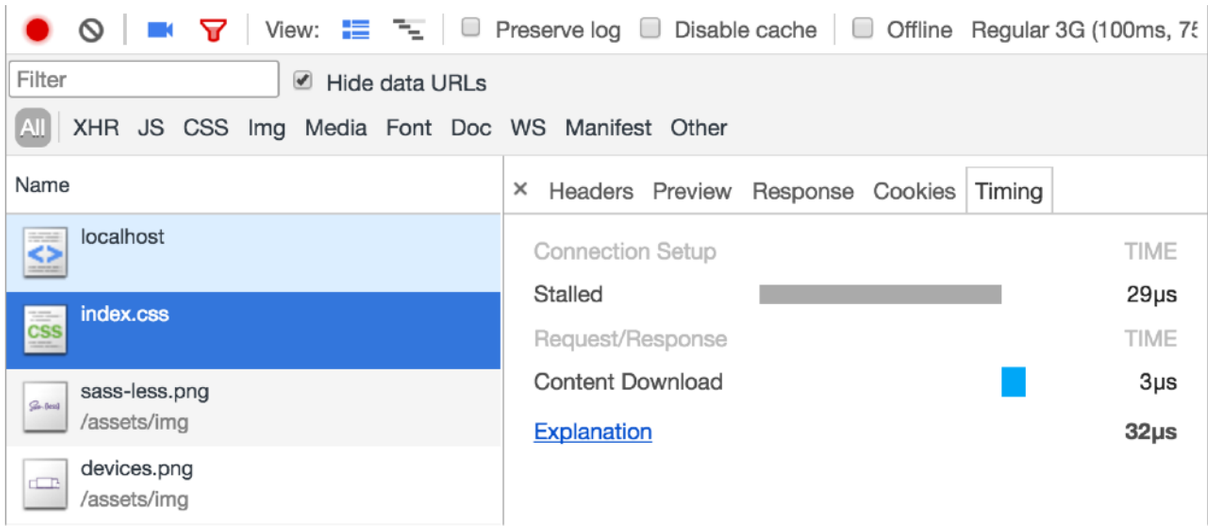
COMPARE

BEFORE



~3.5 sec

AFTER



~ 0.0 sec

REVISIONING

PROBLEM: CACHE INVALIDATION

CSS	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

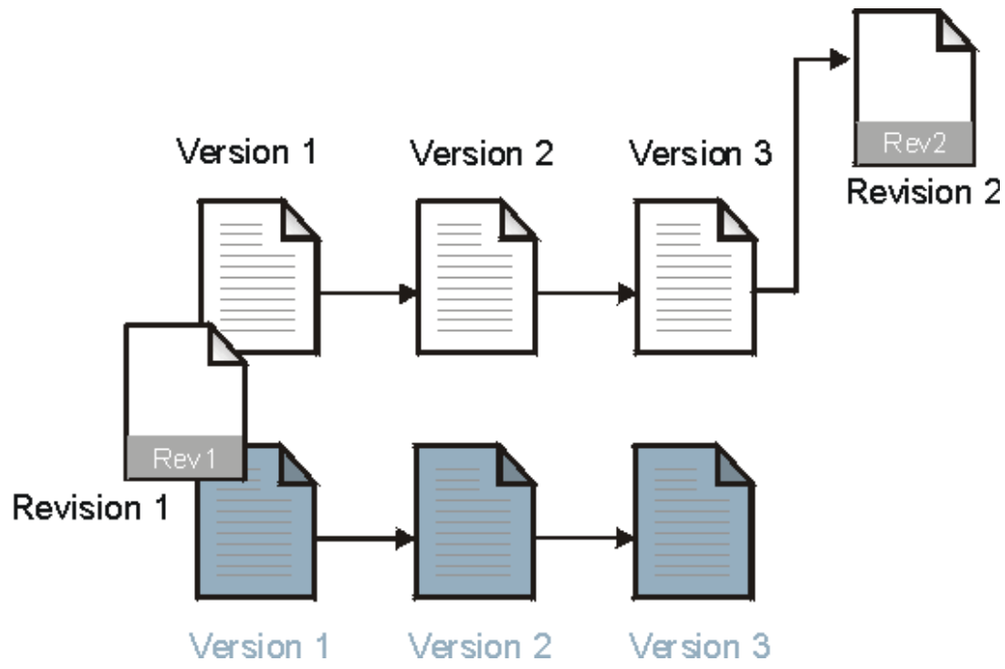
FILE REVISIONING

index.css

index.css?v=2

index-2.css

…



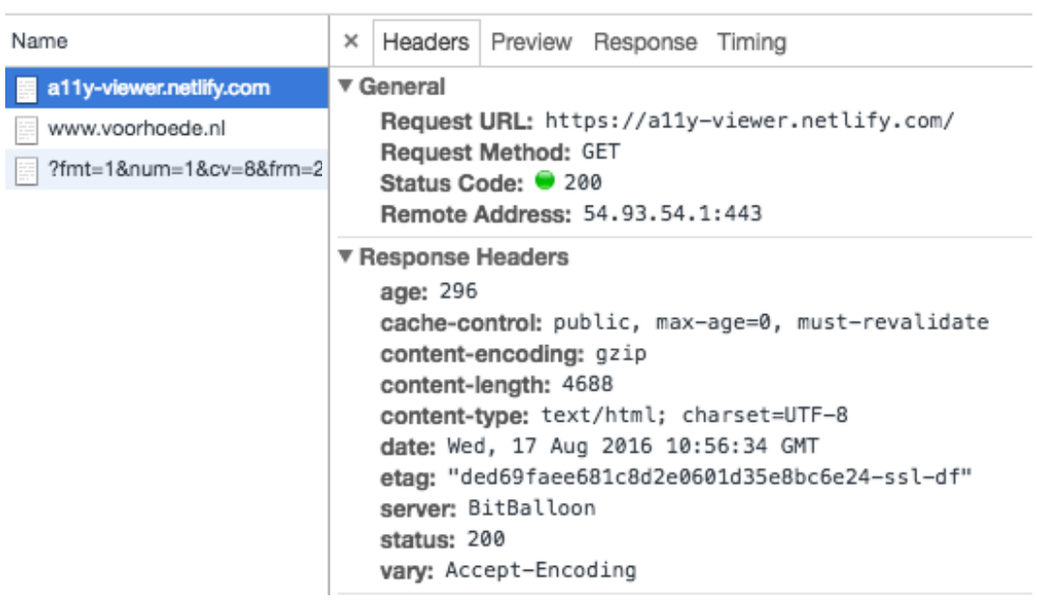
REVISIONED FILES

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

USING ETAG

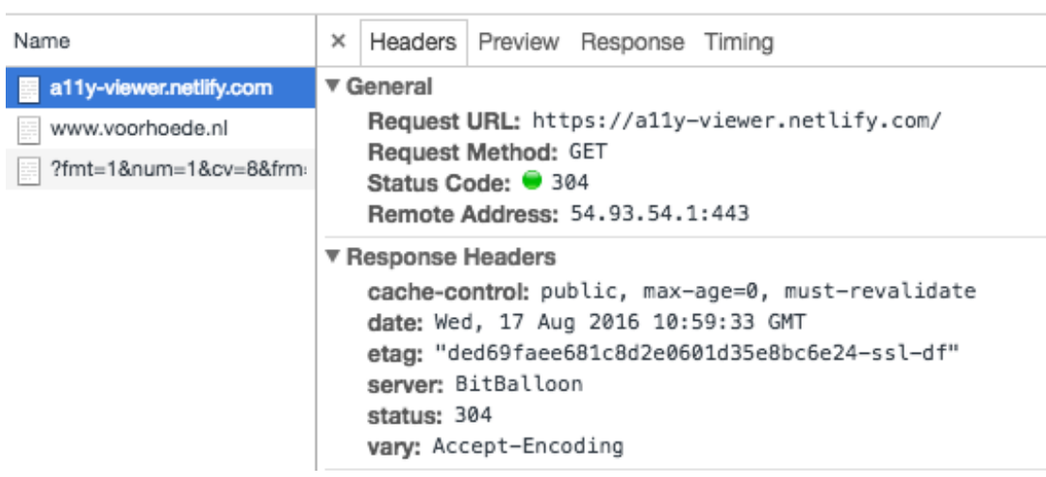
first visit:

- status: 200
- size: 4 688 bytes



repeat visit:

- status: 304
- size: 47 bytes

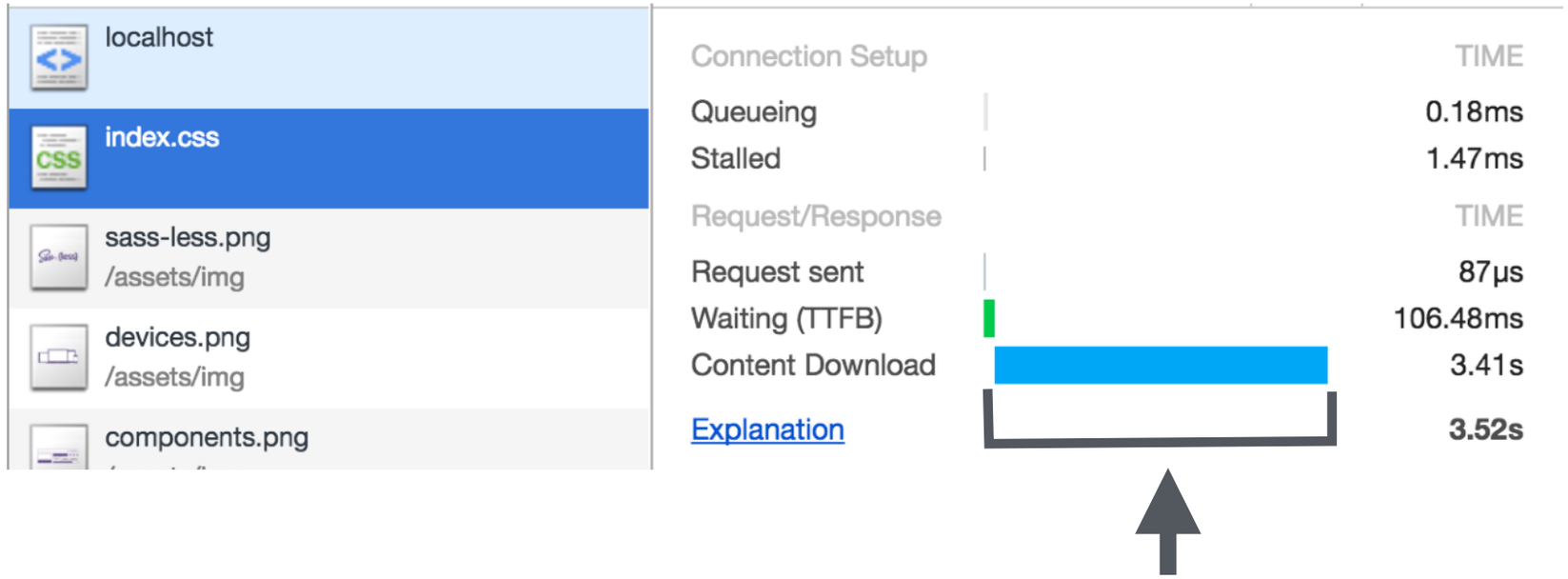


example on netlify.com

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

More effective lossless compression algorithm than gzip and deflate.

Usage % of all users

Global 85%

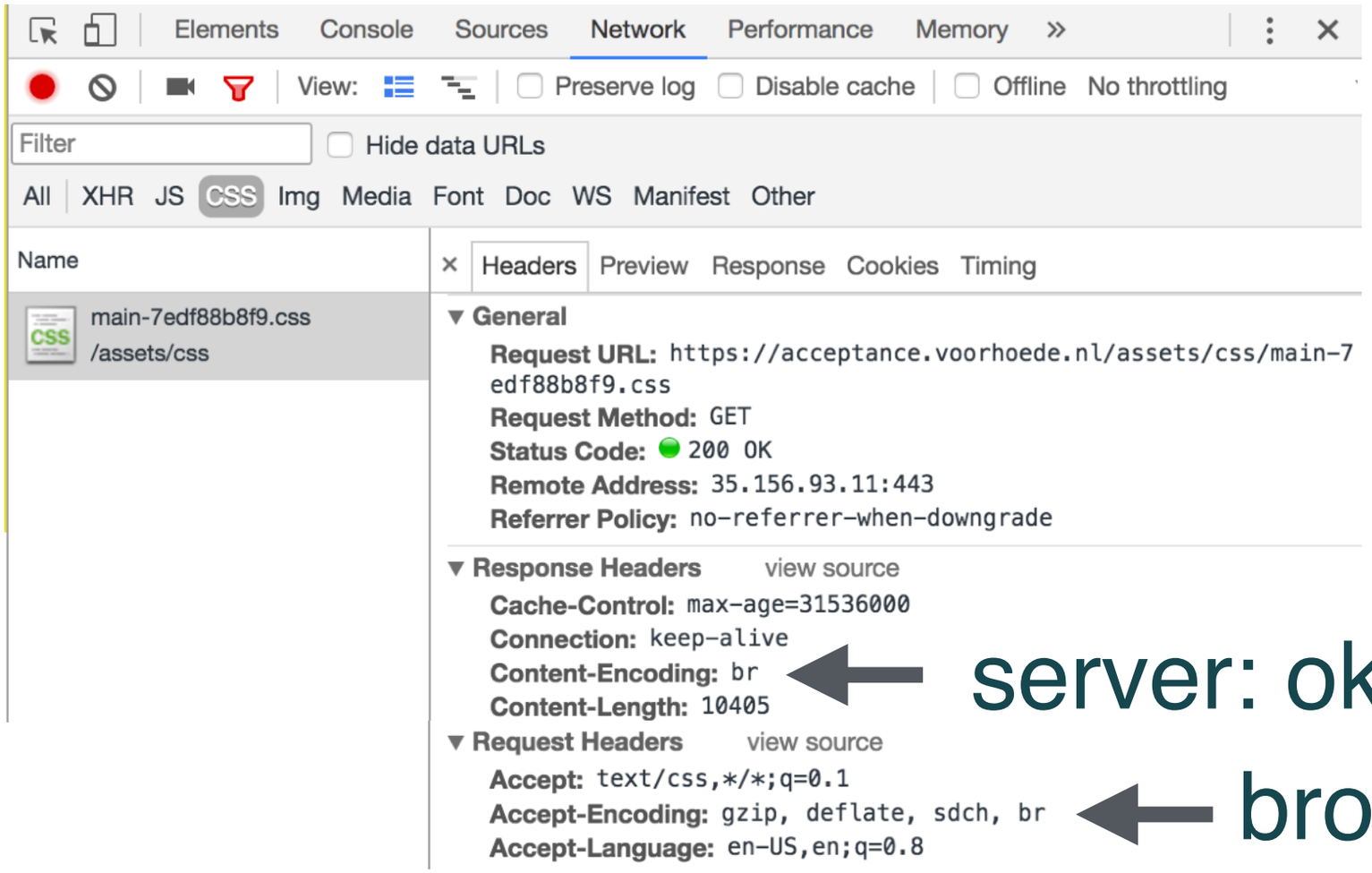
Current aligned | Usage relative | Date relative | Apply filters | Show all | ?

IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			1 49						
			63						
			67						
		61	68	3 11.1	11.2				4
11	17	62	69	3 12	11.4	all	69	11.8	7.2
	18	63	70	3 TP	12				
		64	71						
			72						

*requires HTTPS

caniuse.com/#feat=brotli

ENCODING HEADERS



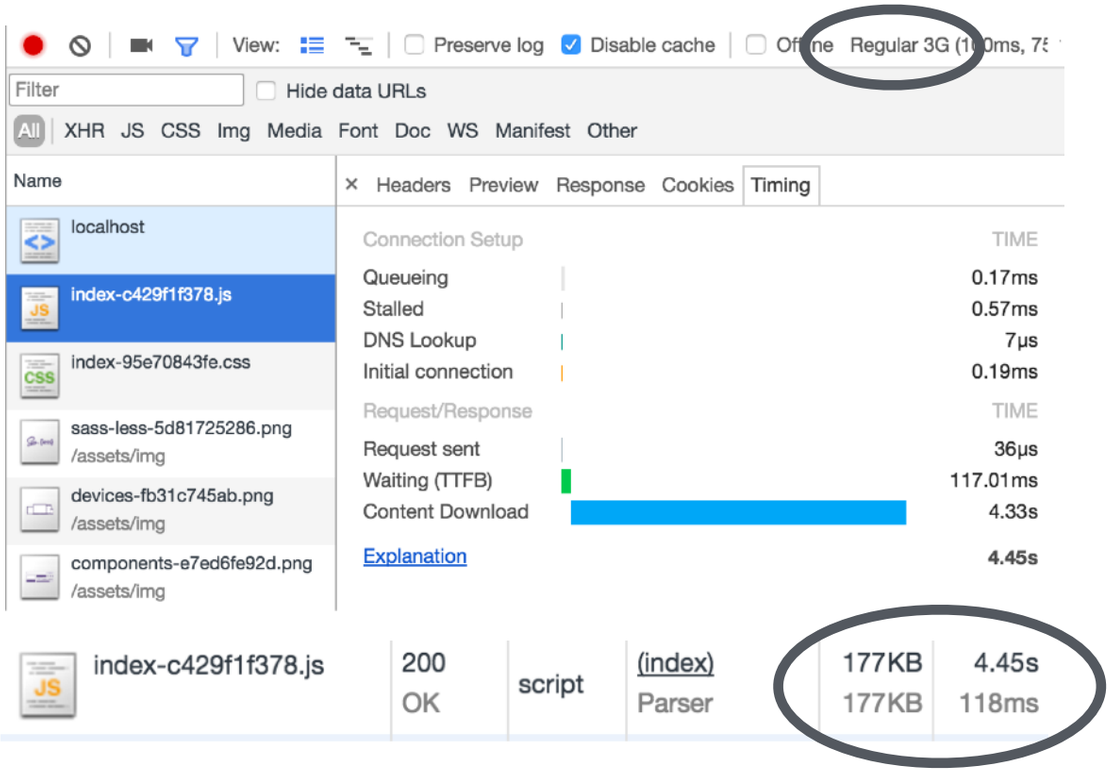
server: okay, I'm using this

browser: I know how to decompress these

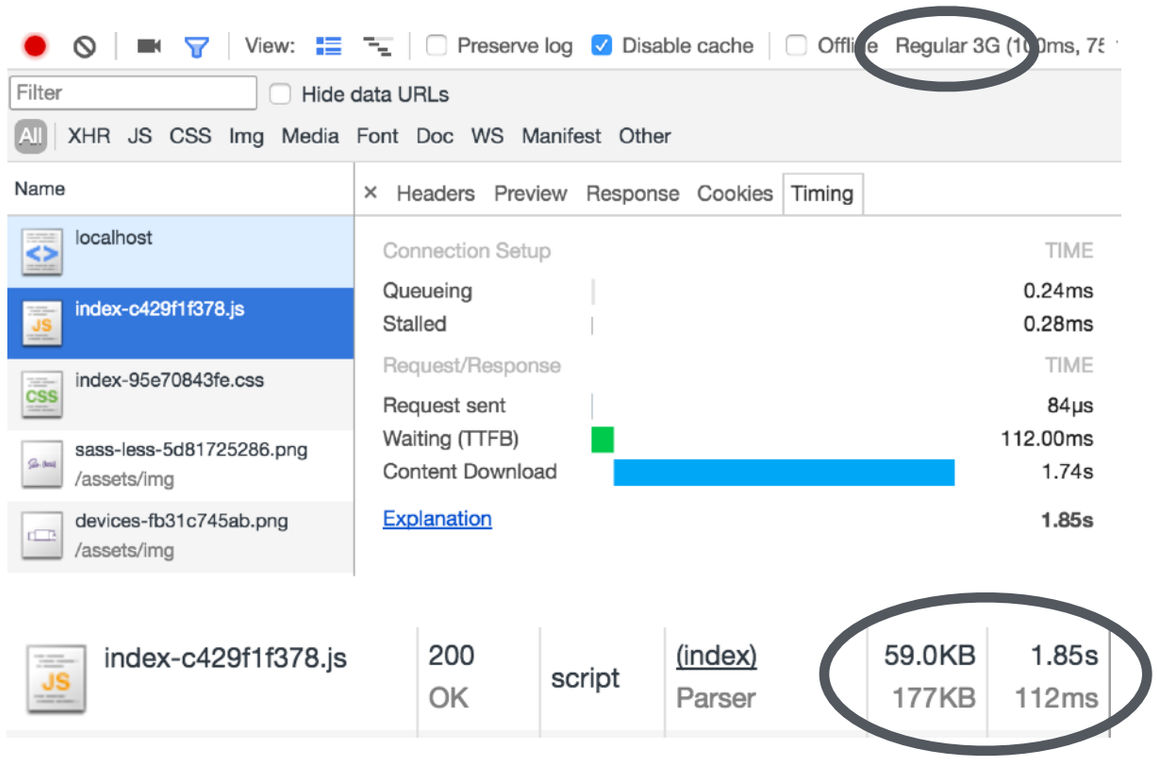
acceptance.voorhoede.nl/en/blog/static-site-implosion-with-brotli-and-gzip/

COMPARE

BEFORE



AFTER

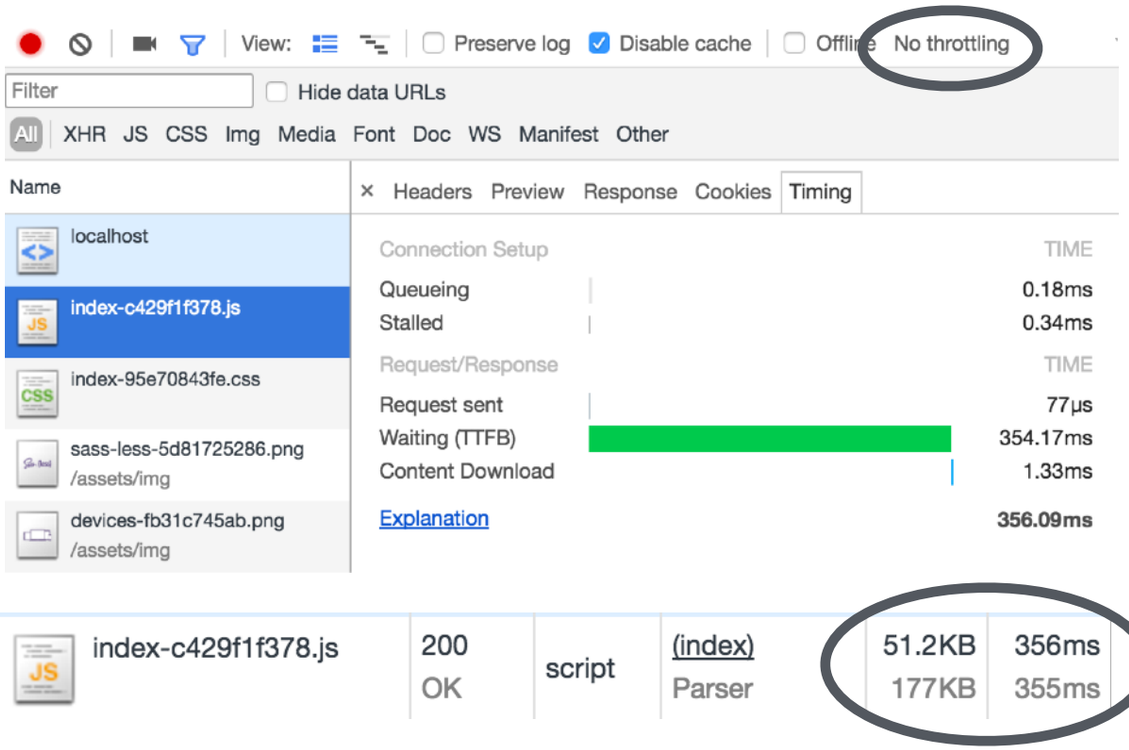


no compression
117 ms wait - 4.45 sec total

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

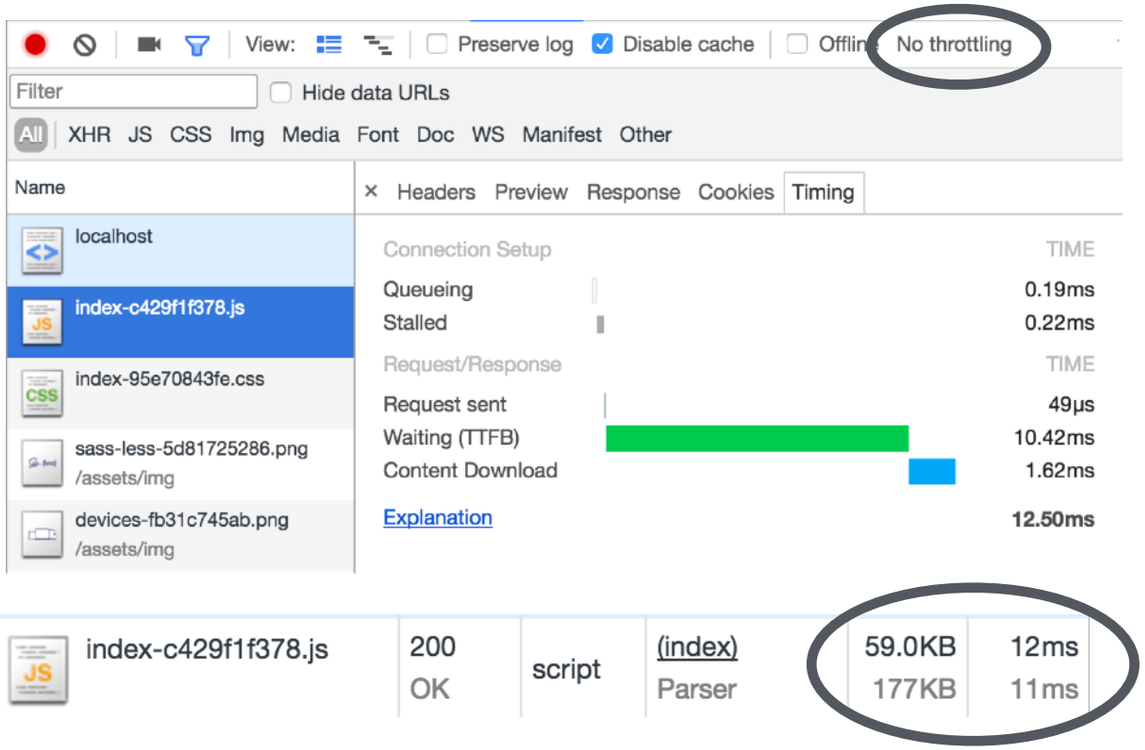
COMPARE

MAX COMPRESS



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

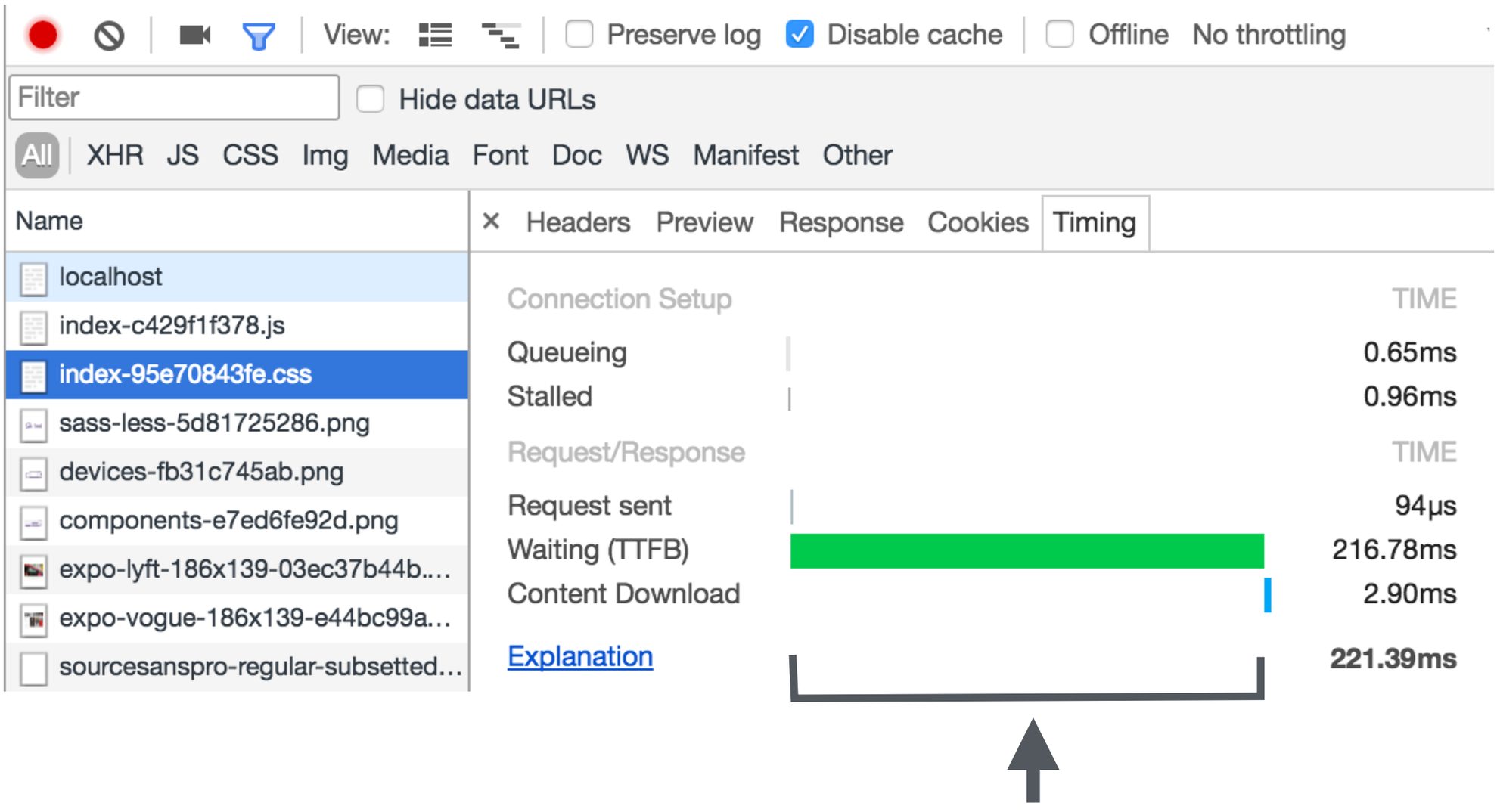
DEFAULT + CACHE



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

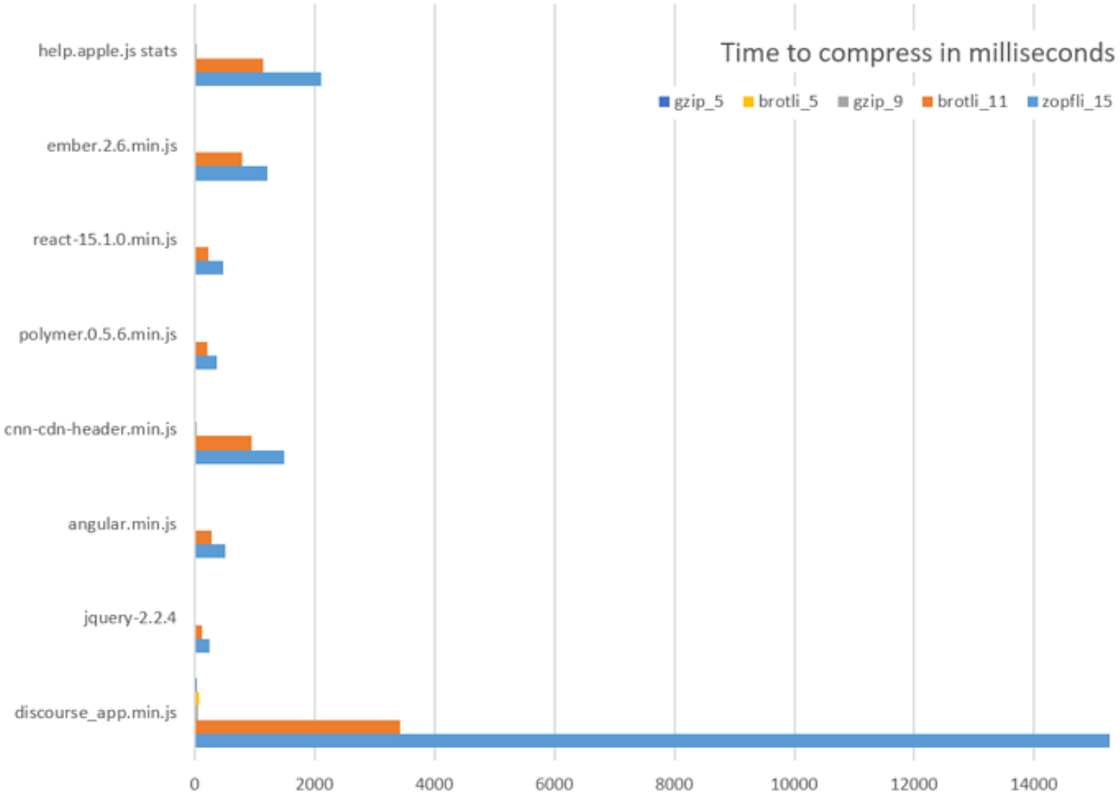
PRECOMPRESSION

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

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:

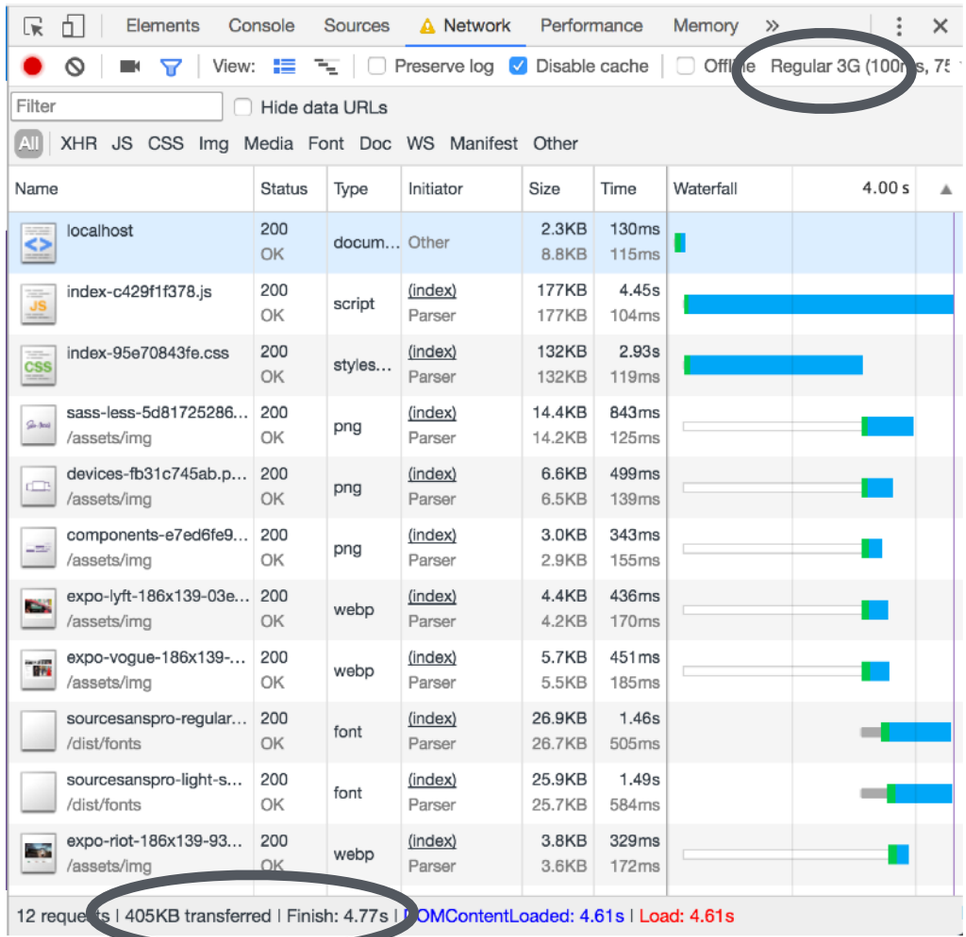
voorhoede.nl/en/blog/static-site-implosion-with-brotli-and-gzip/

PRECOMPRESSED FILES

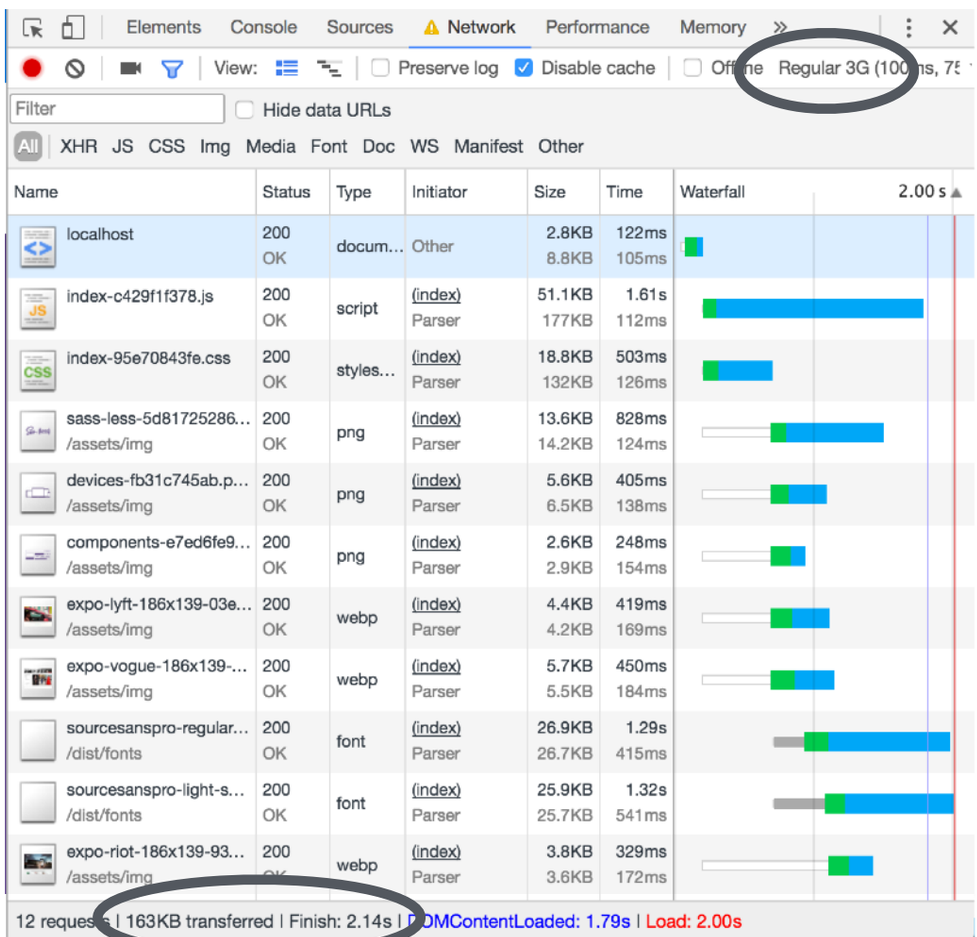
- cache/
  - …
  - index.css
  - index-67760ac4ed.css
  - index-67760ac4ed.css.br
  - index-67760ac4ed.css.gz
  - index.js
  - index-40b67b7222.js
  - index-40b67b7222.js.br
  - index-40b67b7222.js.gz
  - …

COMPARE

DYNAMIC PRECOMPRESSED



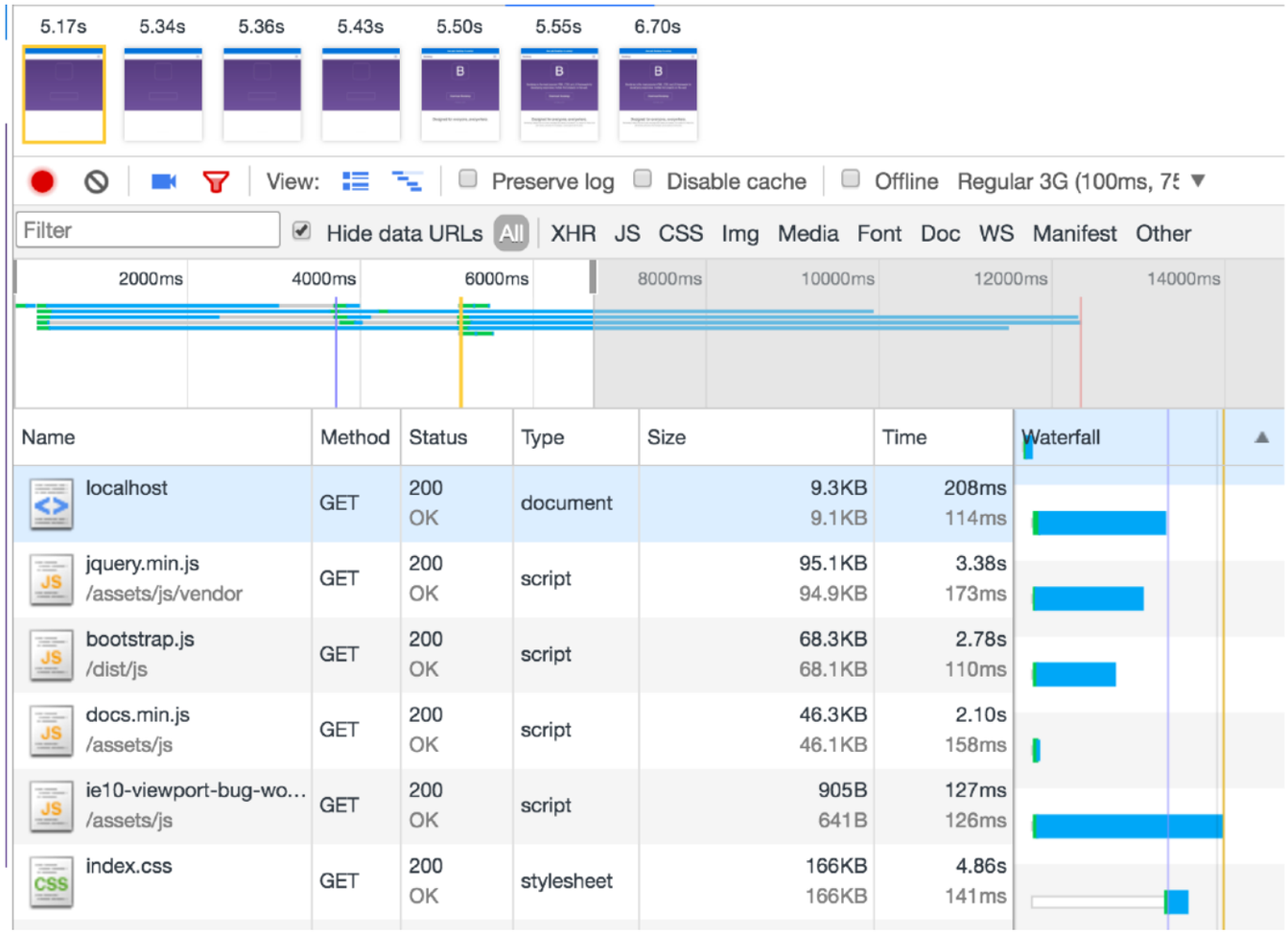
shrink-ray default settings
total 405KB, finish: 4.77 sec



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!

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

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.

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

ASYNC SCRIPTS

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

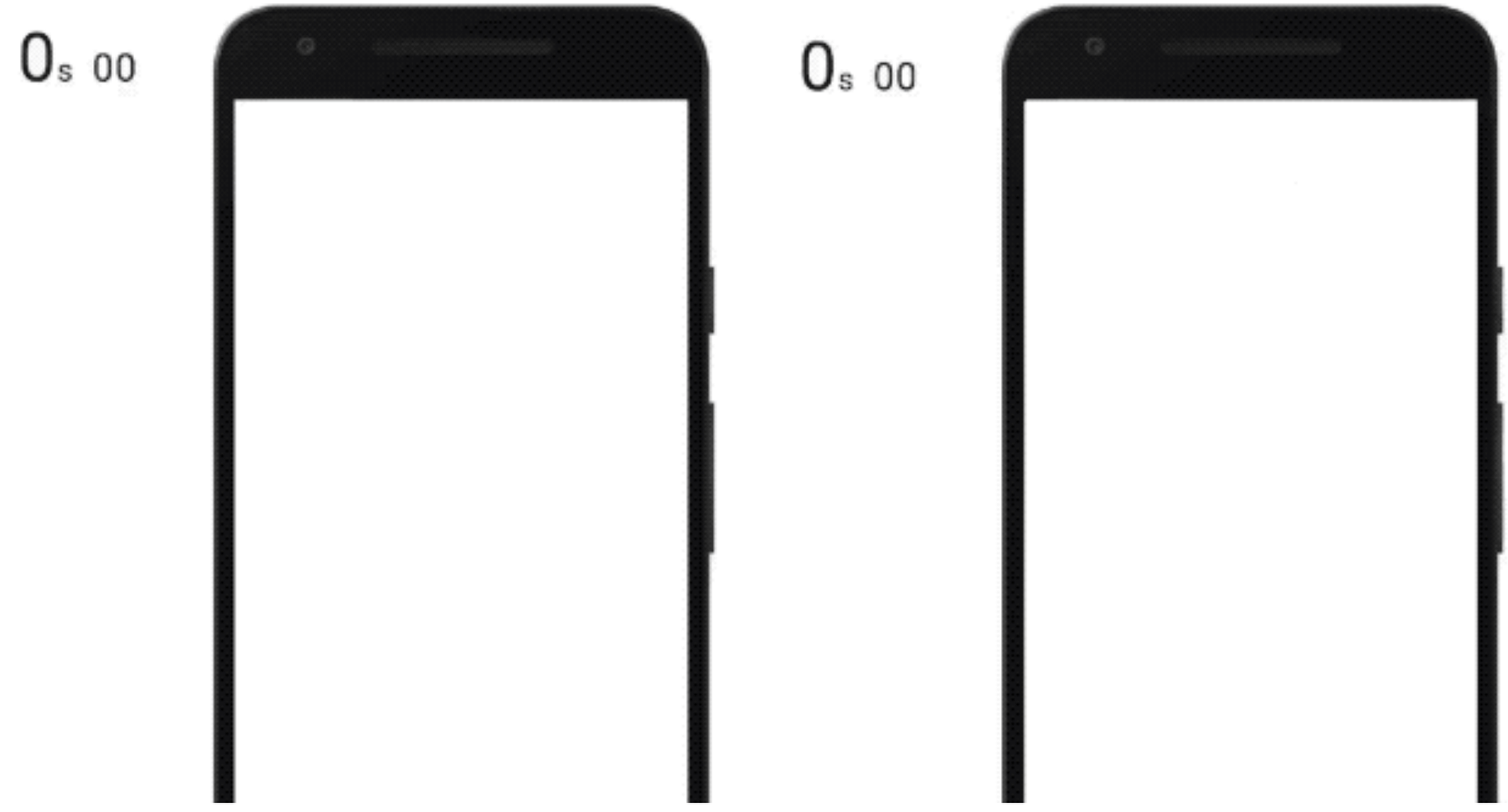
Spec says: Download together, execute in whatever order they download in.

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.

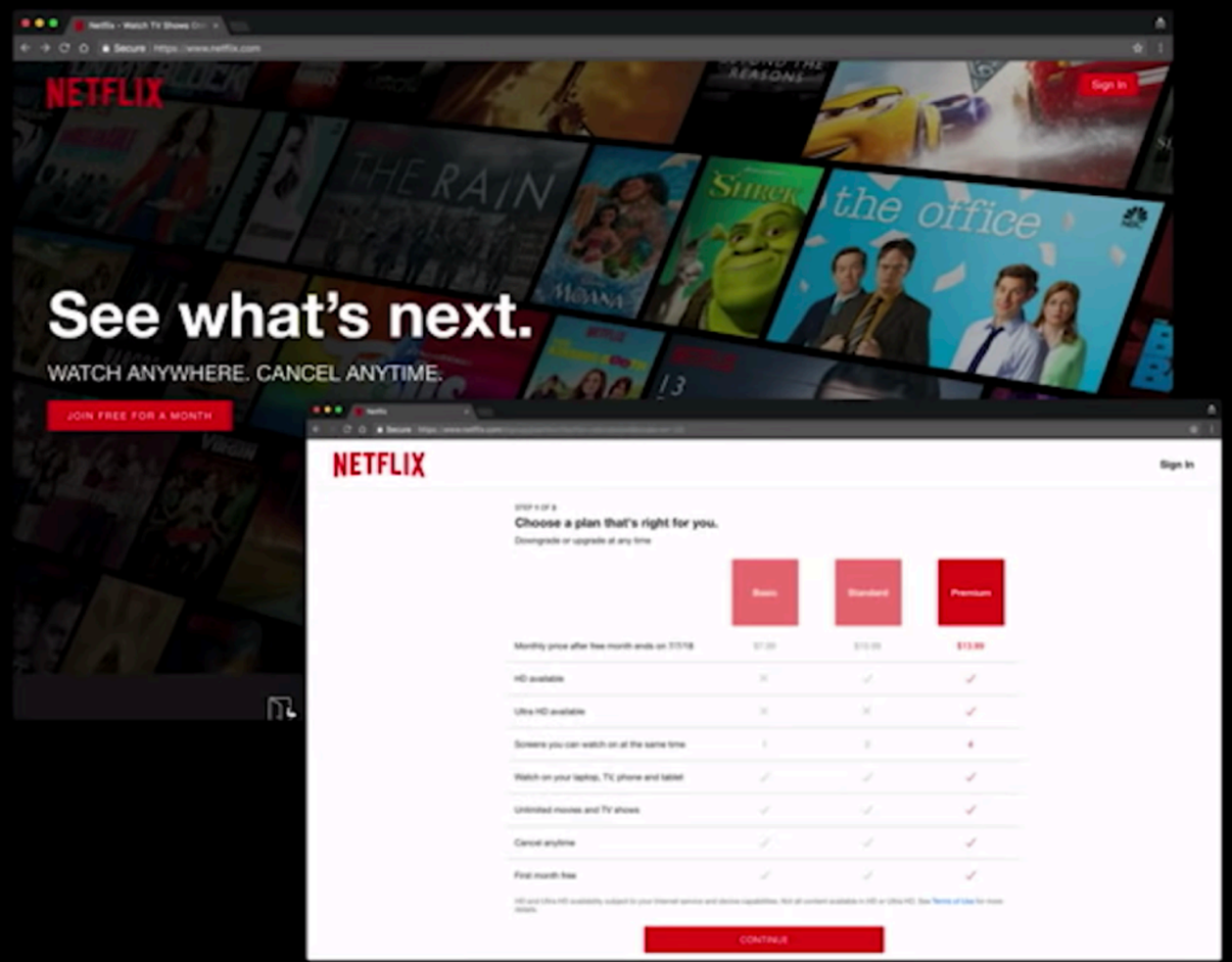
html5rocks.com/en/tutorials/speed/script-loading/

TIME TO INTERACTIVE



The cost of javascript

NETFLIX



REMOVING CLIENT-SIDE REACT, PRE-FETCHING IT FOR FUTURE PAGES

IMPROVED TIME-TO-INTERACTIVE BY 50%

NON-MEMBER LANDING PAGE

REACT WAS STILL USED SERVER-SIDE

The cost of javascript

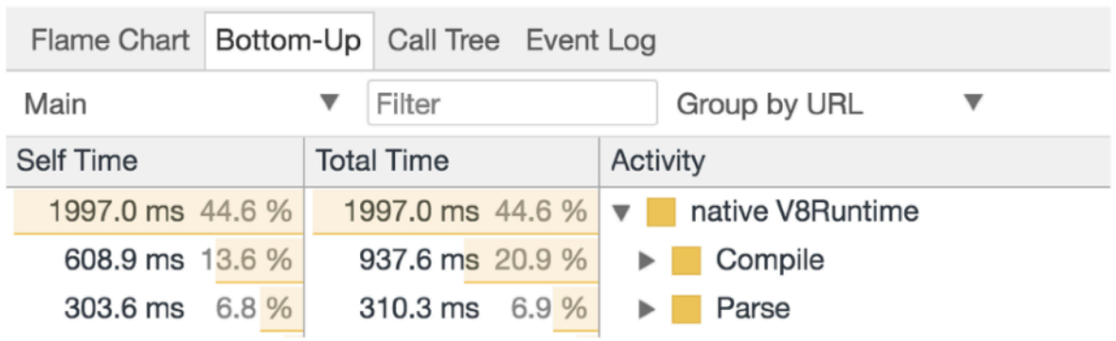
JavaScript bytes !== JPEG bytes

~170KB

~170KB

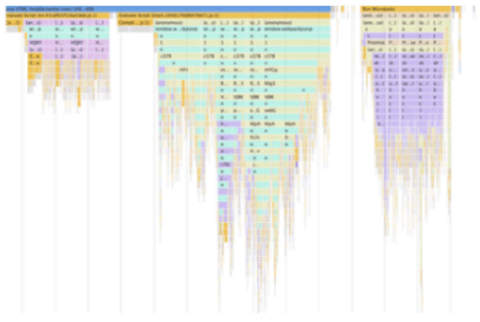
Network Transmission

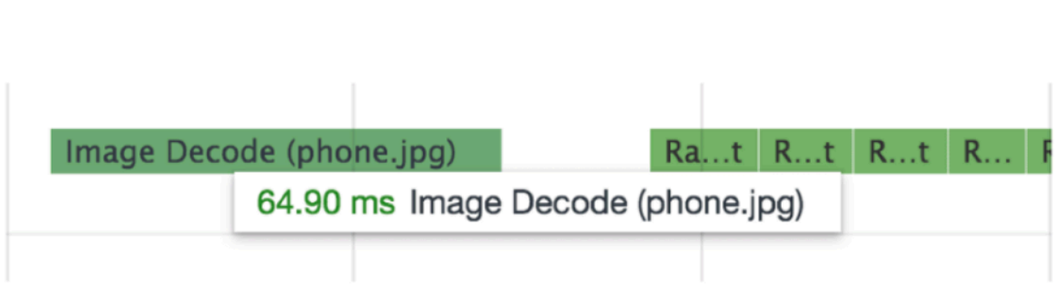
Resource Processing



~2s in Parse/Compile

~1.5s in Execution



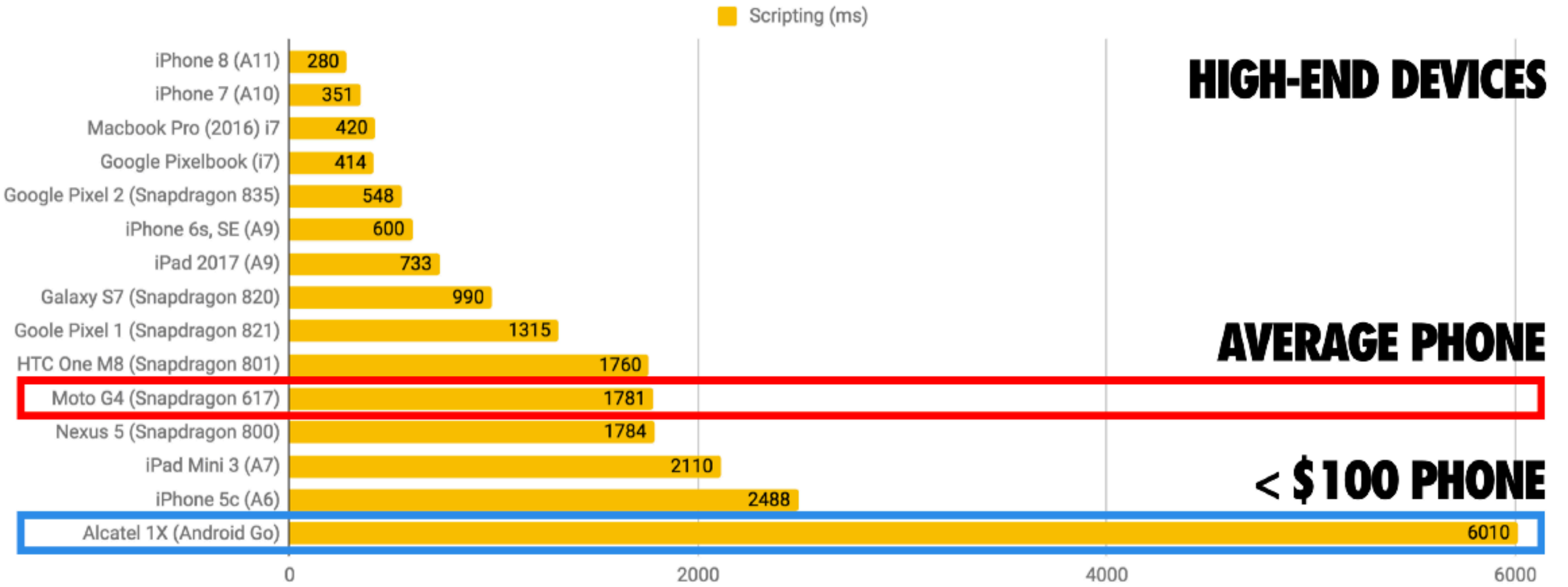


0.064s in Image Decode

0.028s in Rasterize Paint

[The cost of javascript]

2018 JAVASCRIPT PROCESSING TIMES

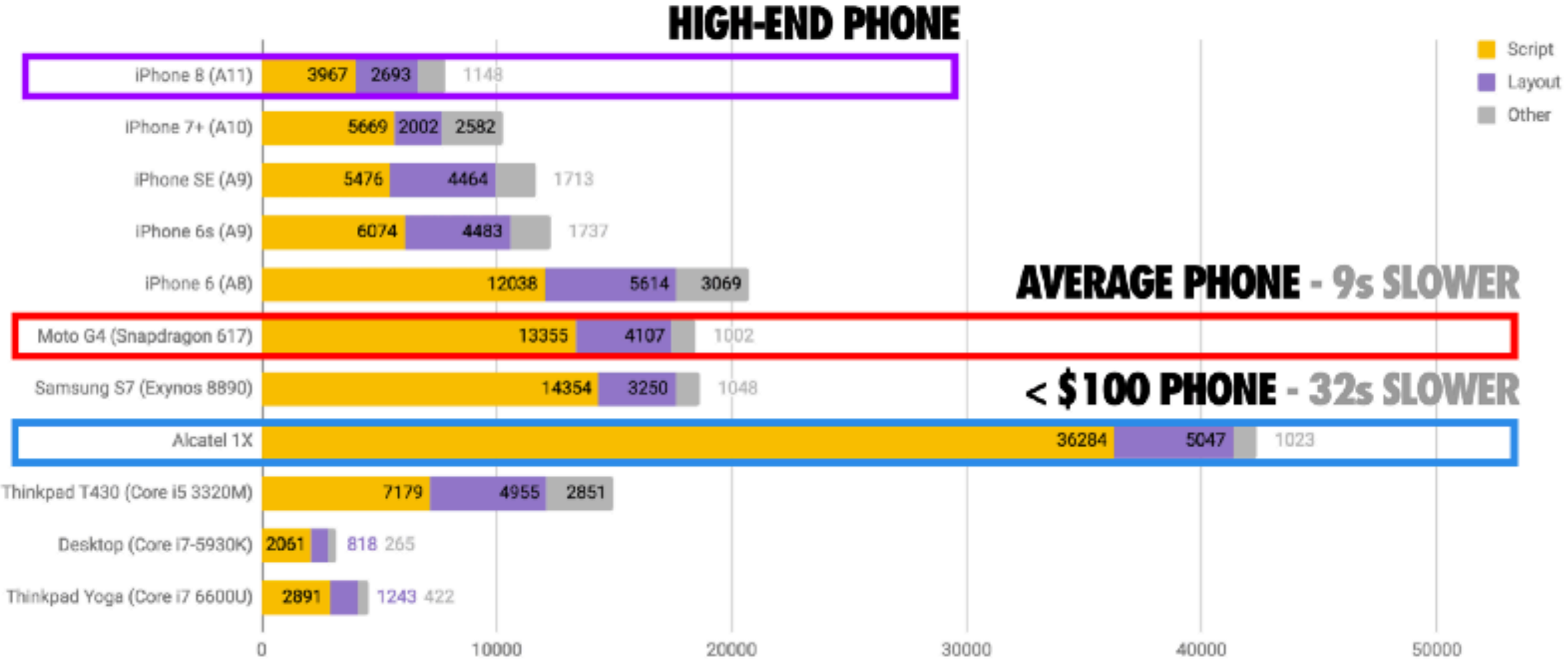


Tests run during July, 2018 on hardware running the latest versions of Android and iOS available

1MB JS UNCOMPRESSED (200KB min/compressed)

The cost of javascript

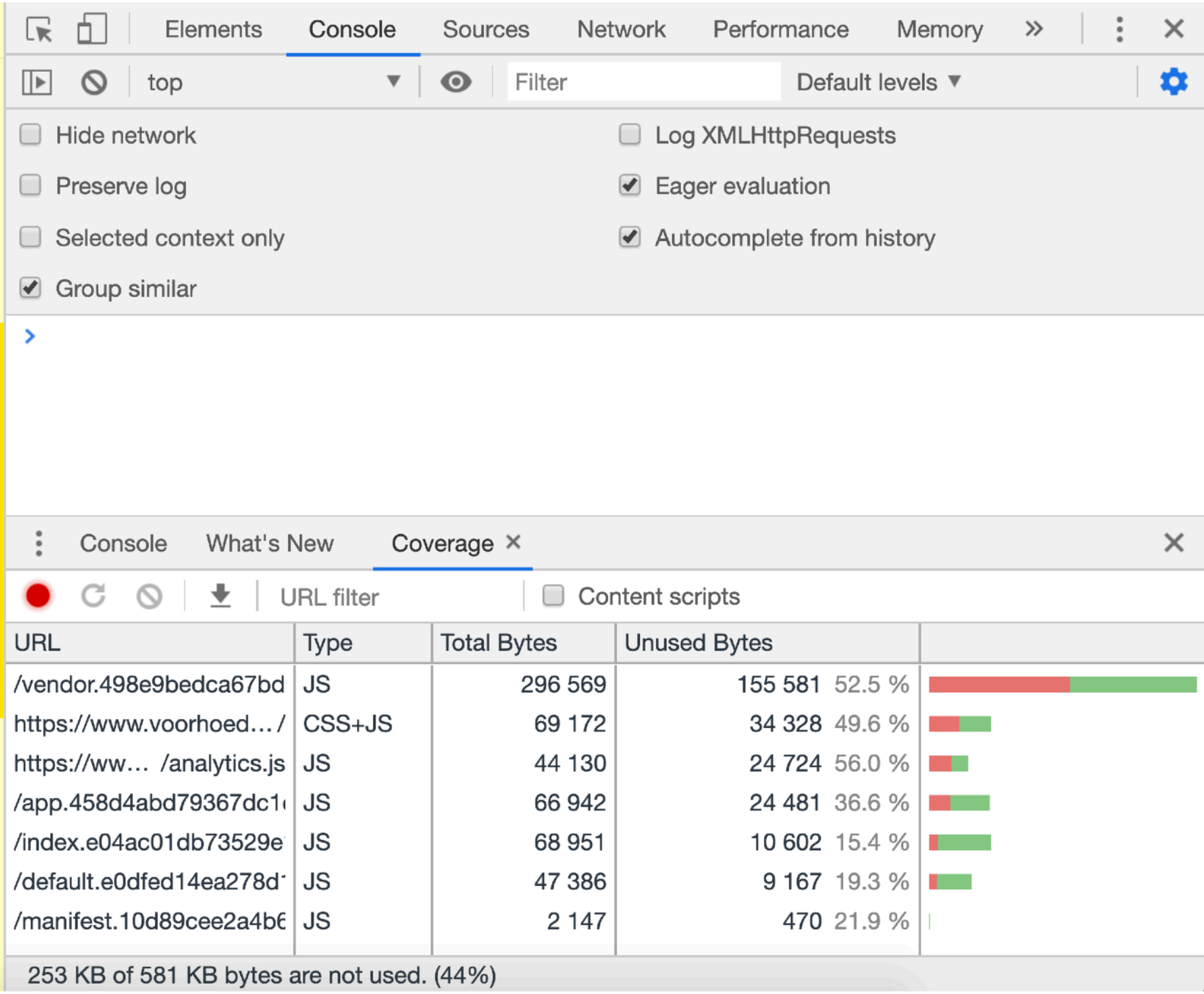
JS PROCESSING FOR CNN.COM



The cost of javascript

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

CHROME DEV TOOLS

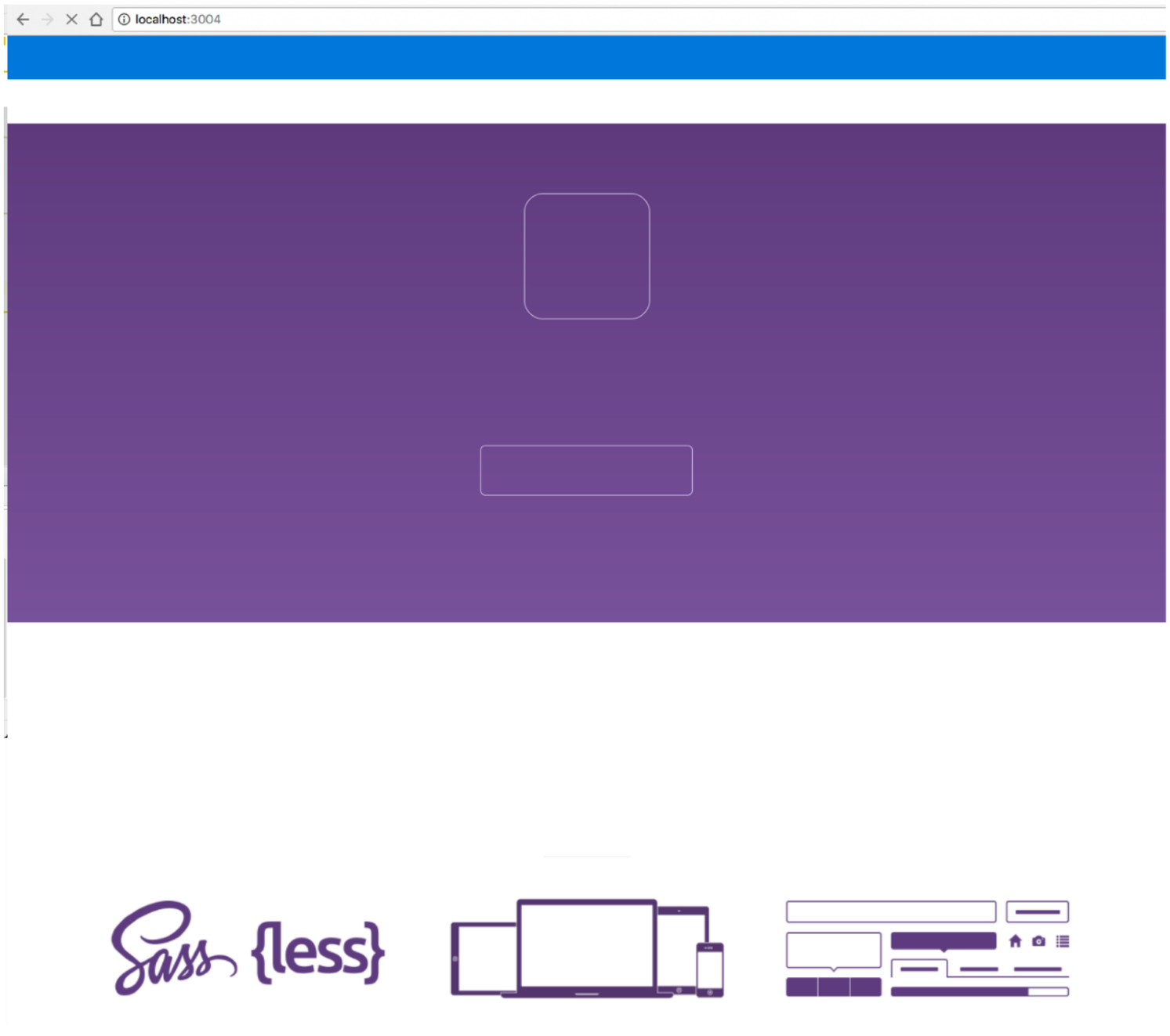


FONT LOADING

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

font squirrel font subsetting

FOIT PROBLEM

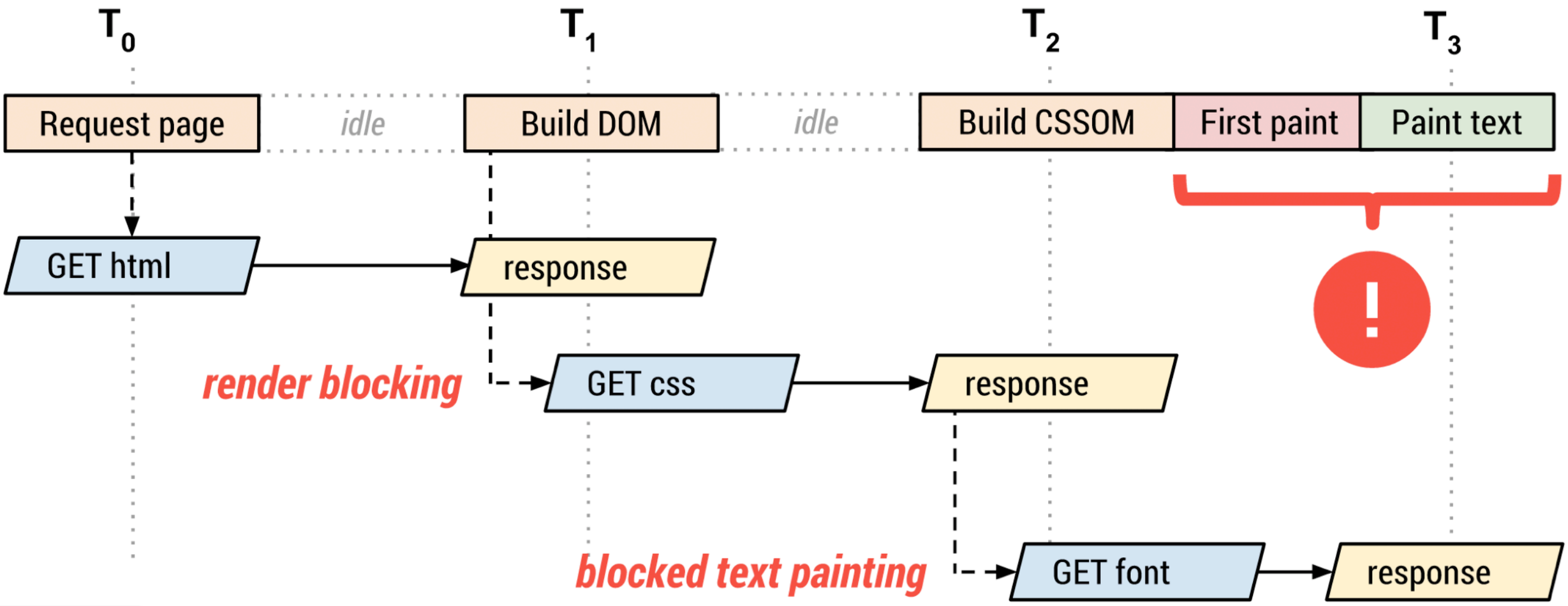


BLOCKING FONT LOADING

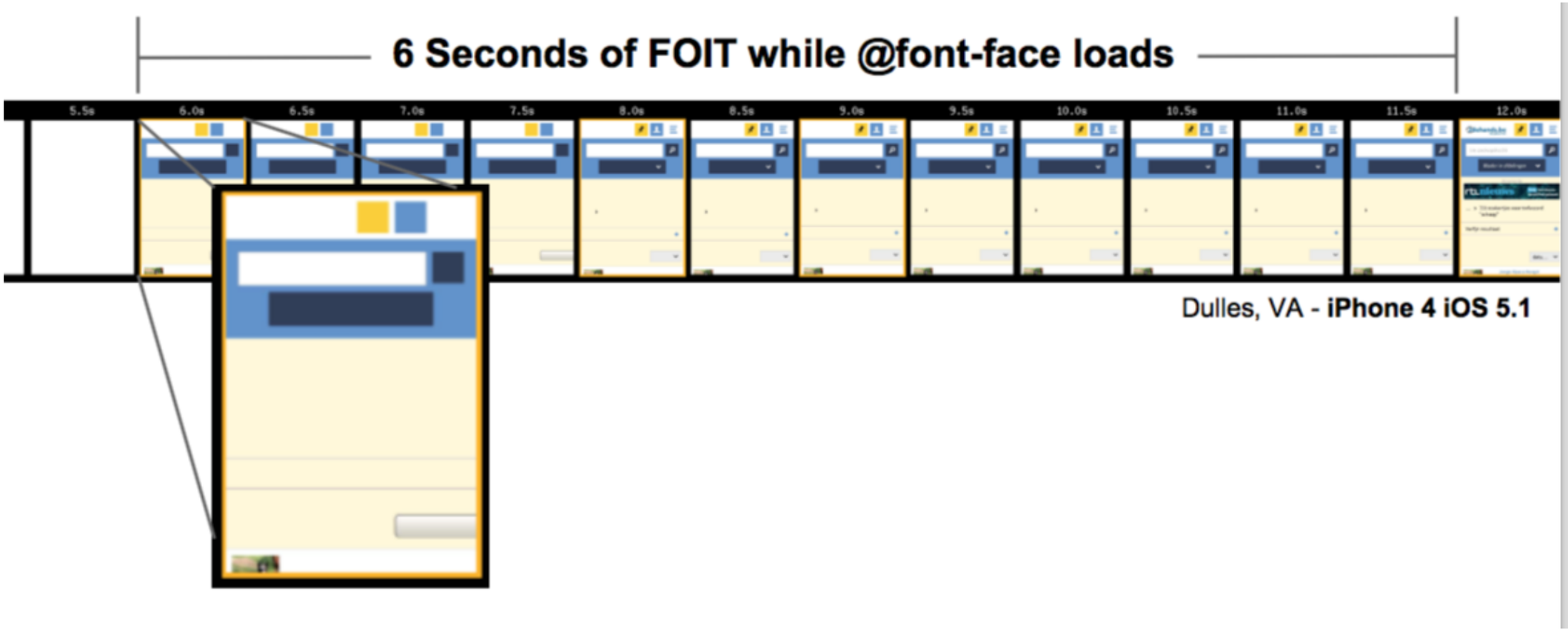
```
@font-face {
    font-family: 'source_sans_pro';
    src: url('sourcesanspro.woff2') format('woff2'),
         url('sourcesanspro.woff') format('woff');
}

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

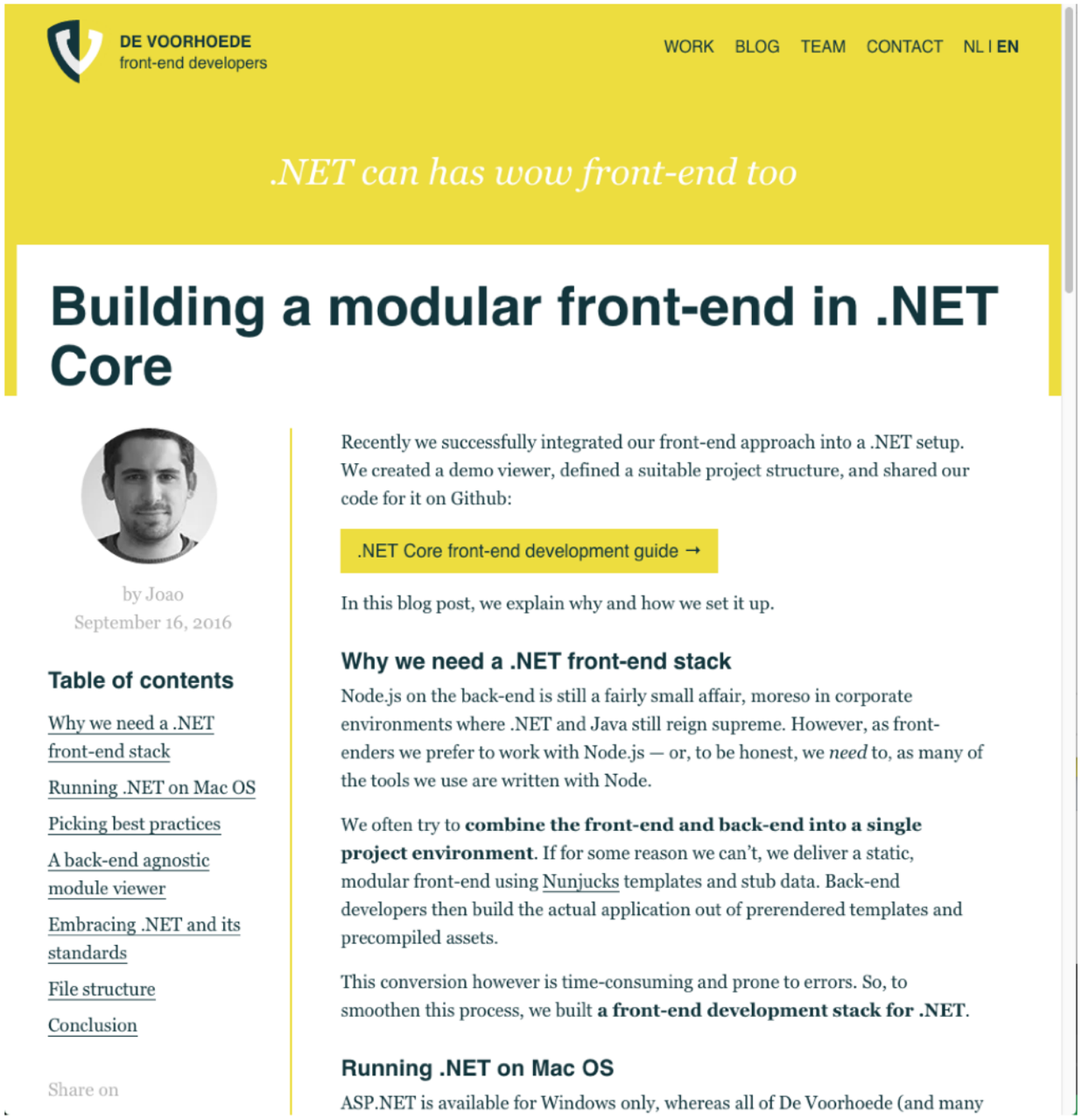
FONT DISPLAY

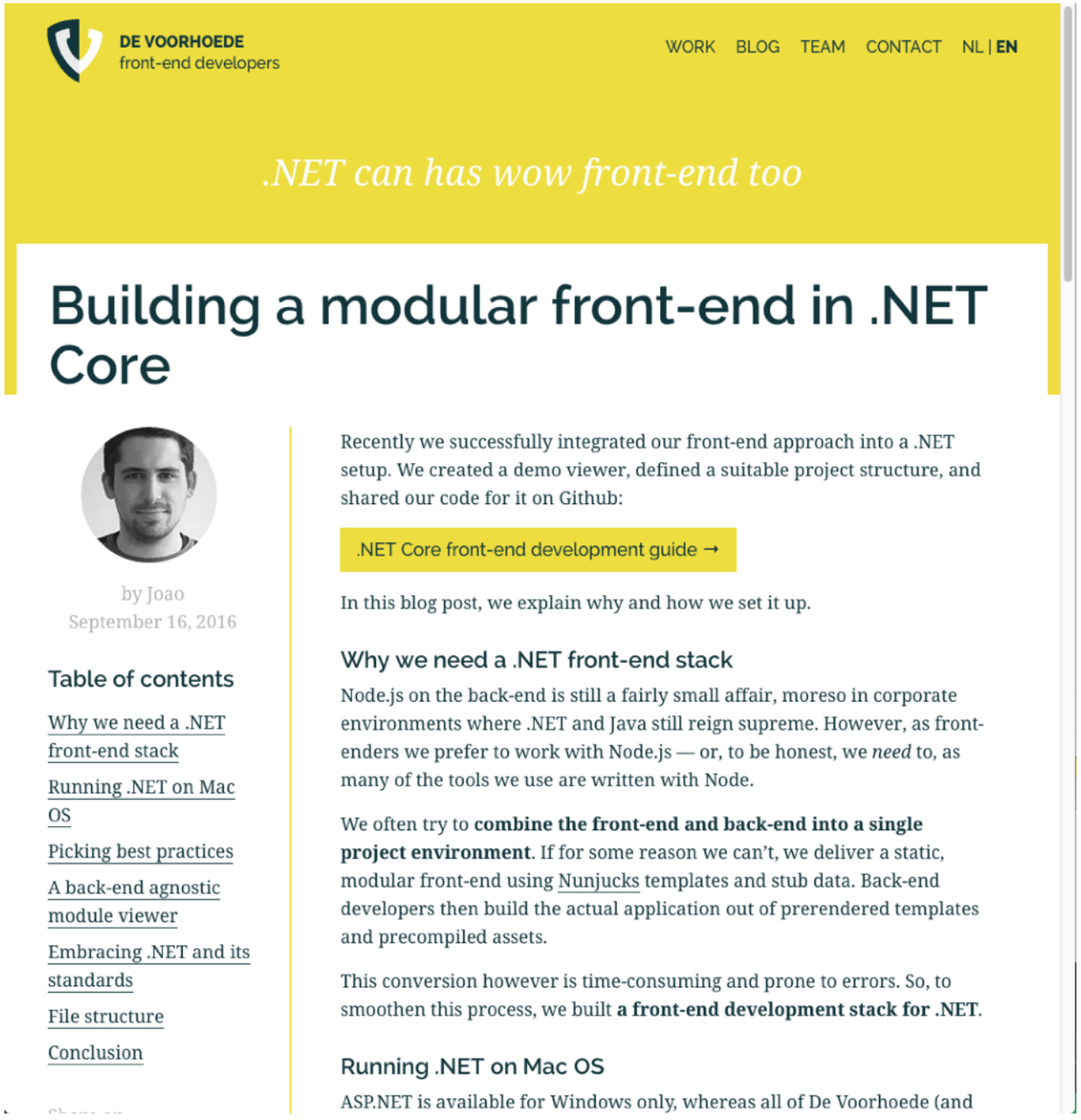
```
@font-face {
     font-family: 'Noto';
     src: url(‘/path/to/noto.woff2')
format(‘woff2’);
     font-display: swap;
}
```

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

Font display property

FOUT WITH FONT DISPLAY





FONT RENDERING CONTROLS

CSS font-rendering controls - WD

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

Usage % of all users
Global 74.78%

Current aligned | Usage relative | Date relative | Apply filters | Show all | ?

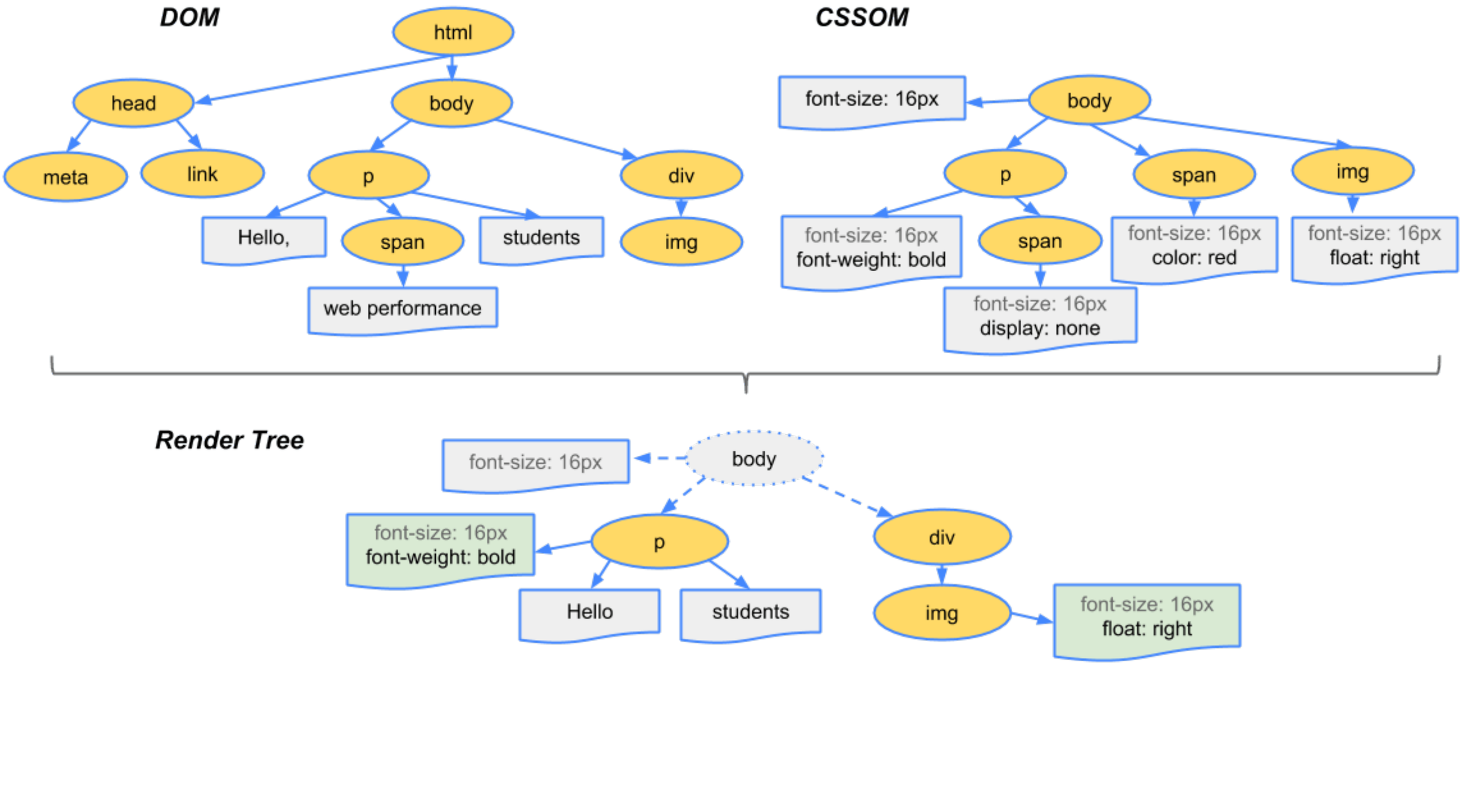
IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			1 49						
			63						
			67						
		61	68	11.1	11.2				4
11	17	62	69	12	11.4	all	69	11.8	1 7.2
	18	63	70	TP	12				
		64	71						
			72						

COMPARE FOIT

COMPARE FOUT

LOAD CSS

PROBLEM


DOM
html
head
body
meta
link
p
div
Hello,
span
students
img
web performance
CSSOM
font-size: 16px
body
p
span
img
font-size: 16px
font-weight: bold
span
font-size: 16px
color: red
font-size: 16px
float: right
font-size: 16px
display: none
Render Tree
font-size: 16px
body
font-size: 16px
font-weight: bold
p
div
Hello
students
img
font-size: 16px
float: right

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: Preserve log Disable cache Offline Regular 3G (100ms, 7

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 1100

Name	Met…	Stat…	Type	Initiator	Size	Time	Waterfall 4.00 s 6.00 s 8
localhost	GET	200	doc…	Other	9.4…	212…	
bootstrap.css	GET	200	styl…	(index)	144…	3.31s	
docs.css	GET	200	styl…	(index)	30.6…	967…	
expo-newsweek.jpg	GET	200	jpeg	(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-workaro…	GET	200	script	(index)	905B	128…	
sass-less.png	GET	200	png	(index)	14.4…	904…	
devices.png	GET	200	png	(index)	6.7…	514…	
components.png	GET	200	png	(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	jpeg	(index)	156…	8.31s	
expo-vogue.jpg	GET	200	jpeg	(index)	194…	9.13s	
expo-riot.jpg	GET	200	jpeg	(index)	158…	8.09s	
ZeroClipboard.swf?noCach…	GET	200	x-sh…	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: –>
  <link rel=“stylesheet” href=“index-
f3c2b4.css”>
</head>
```

```
<body>
  <!– Loading late means flash of unstyled
content: –>
  <link rel=“stylesheet” href=“index-
f3c2b4.css”>
</body>
```

LOAD STYLES ASYNC

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

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

filamentgroup/loadcss

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.

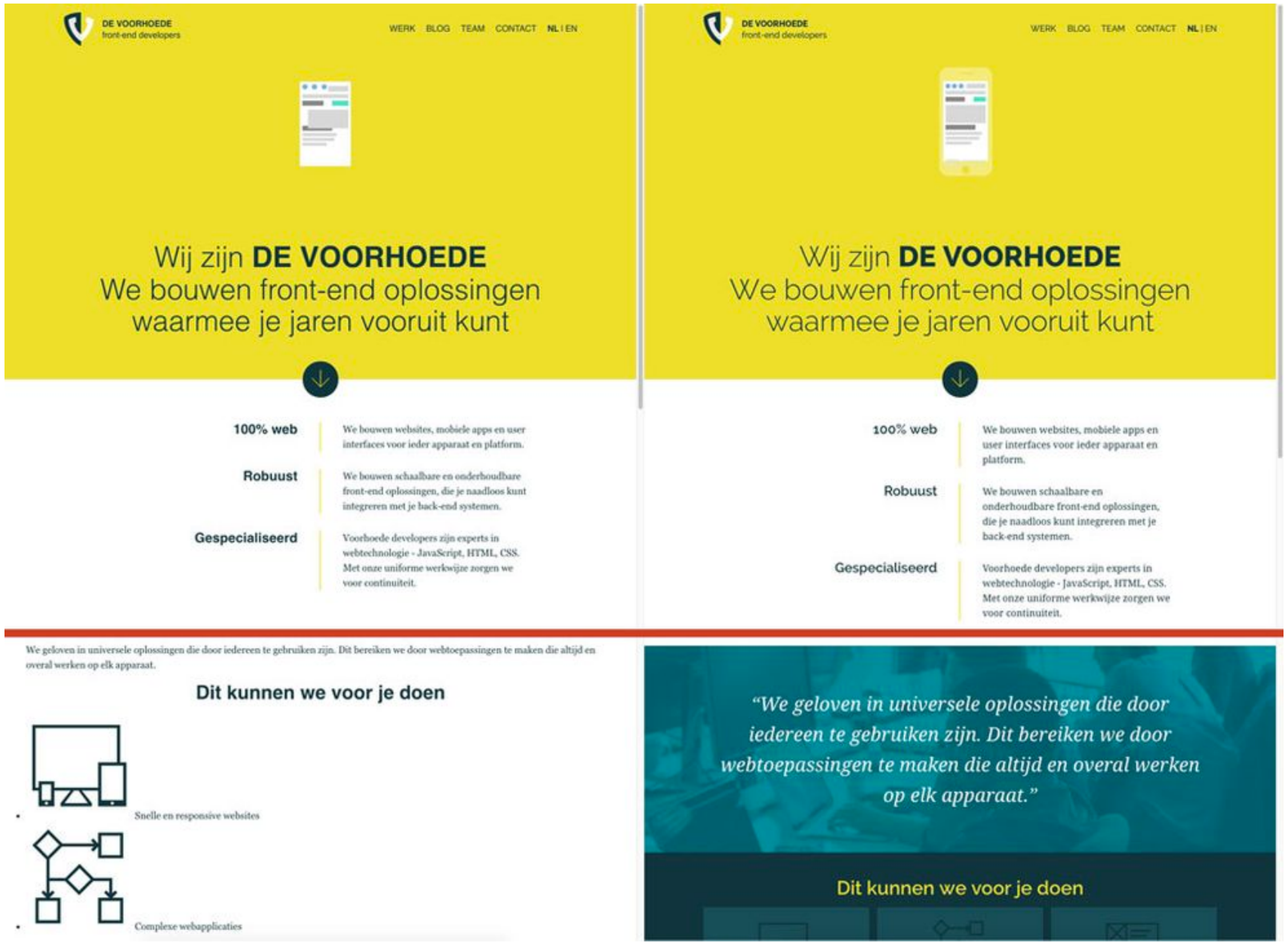
Bootstrap makes front-end web development faster and easier. It's made for folks of all skill levels, devices of all shapes, and projects of all sizes.

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



[Critical CSS on voorhoede.nl]

CONDITIONAL STYLE LOADER

```
<html><head><style>/*critical CSS*/</style><scrip
// (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:

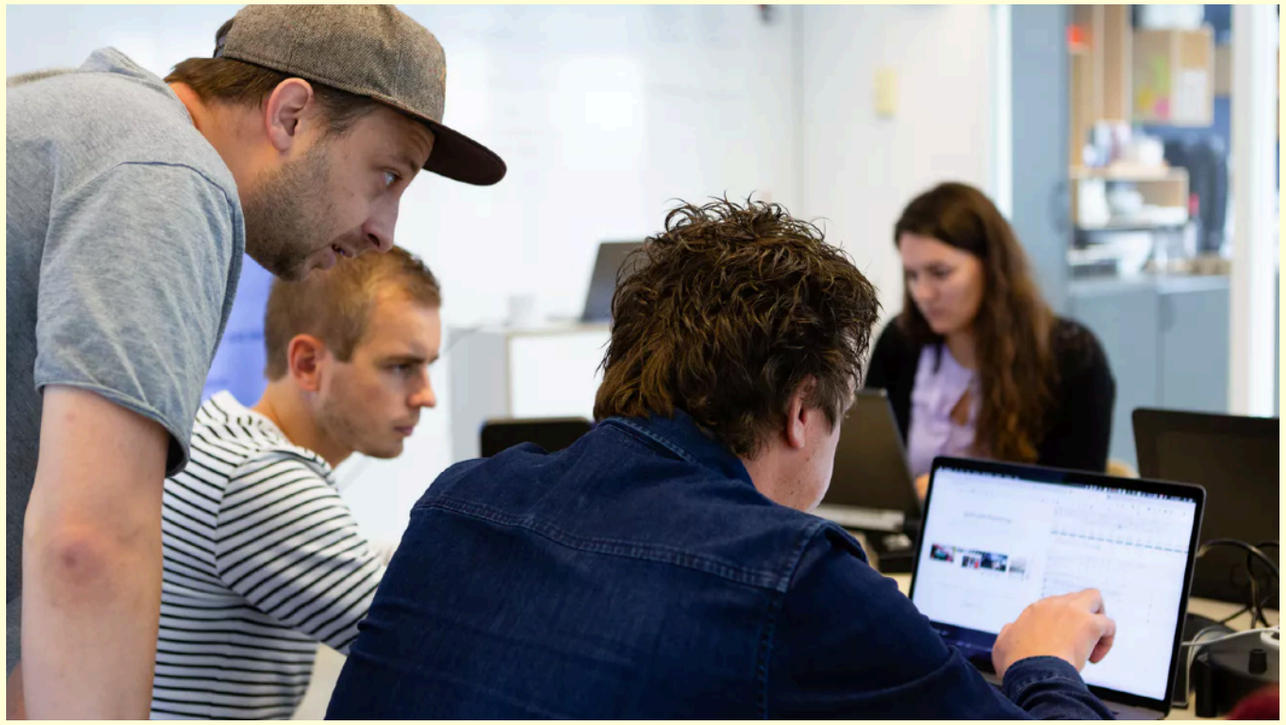
```
<link rel="stylesheet" href="index-f3c2b4.css">
```

onsole 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

De Voorhoede



thanks

Declan Rek

// @DeclanRek

// voorhoede.nl