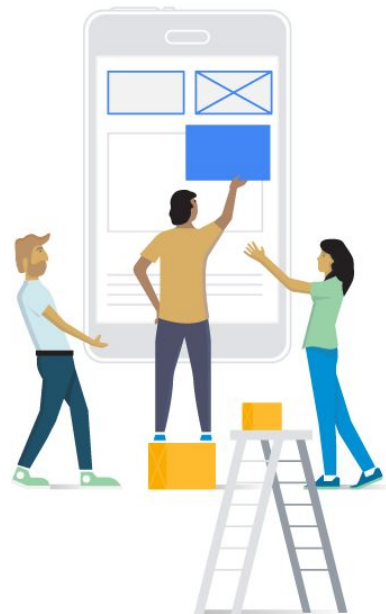




# Key techniques for fast websites

Antoine Brossault  
abrossault@google.com



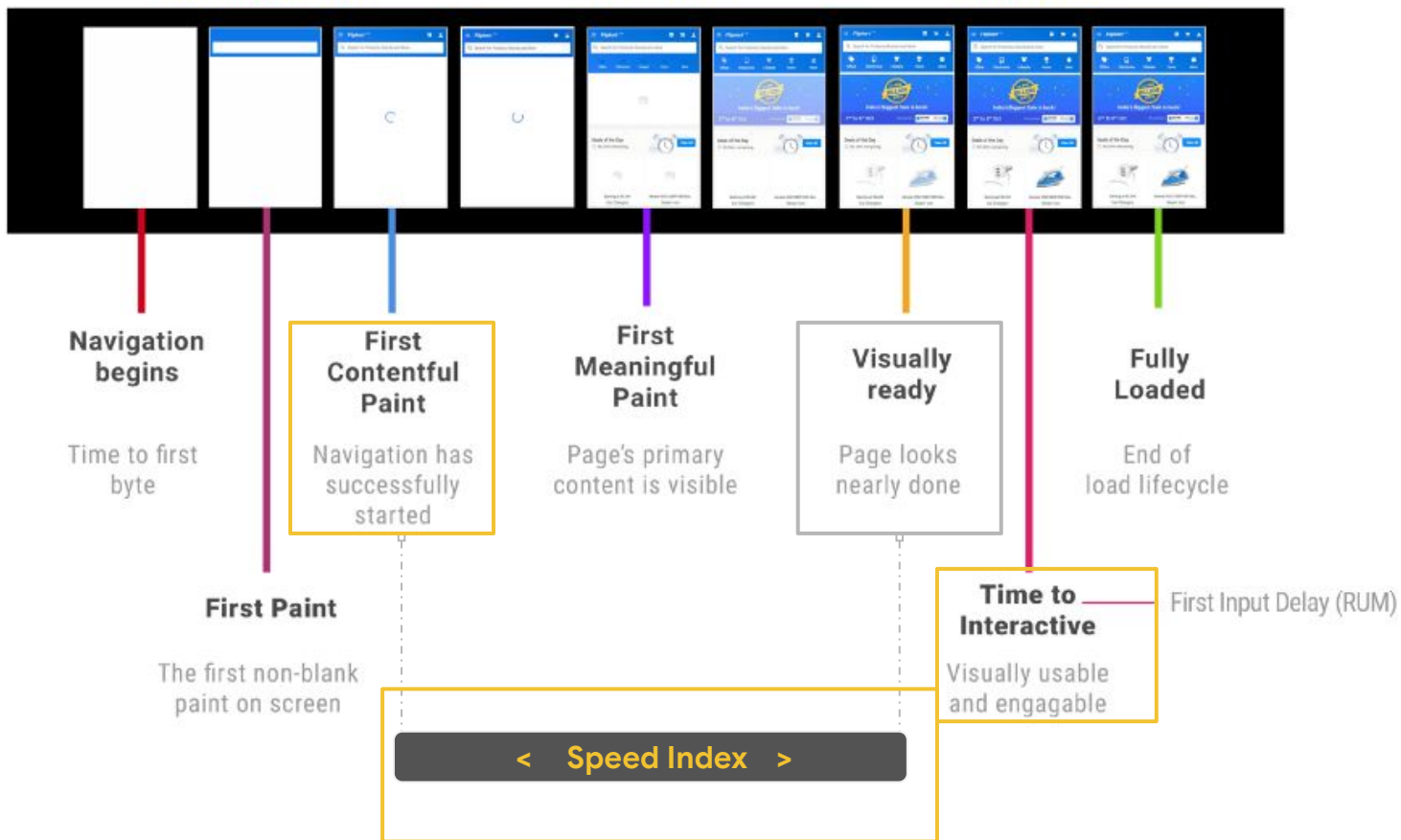
# Performance metrics



is it happening?

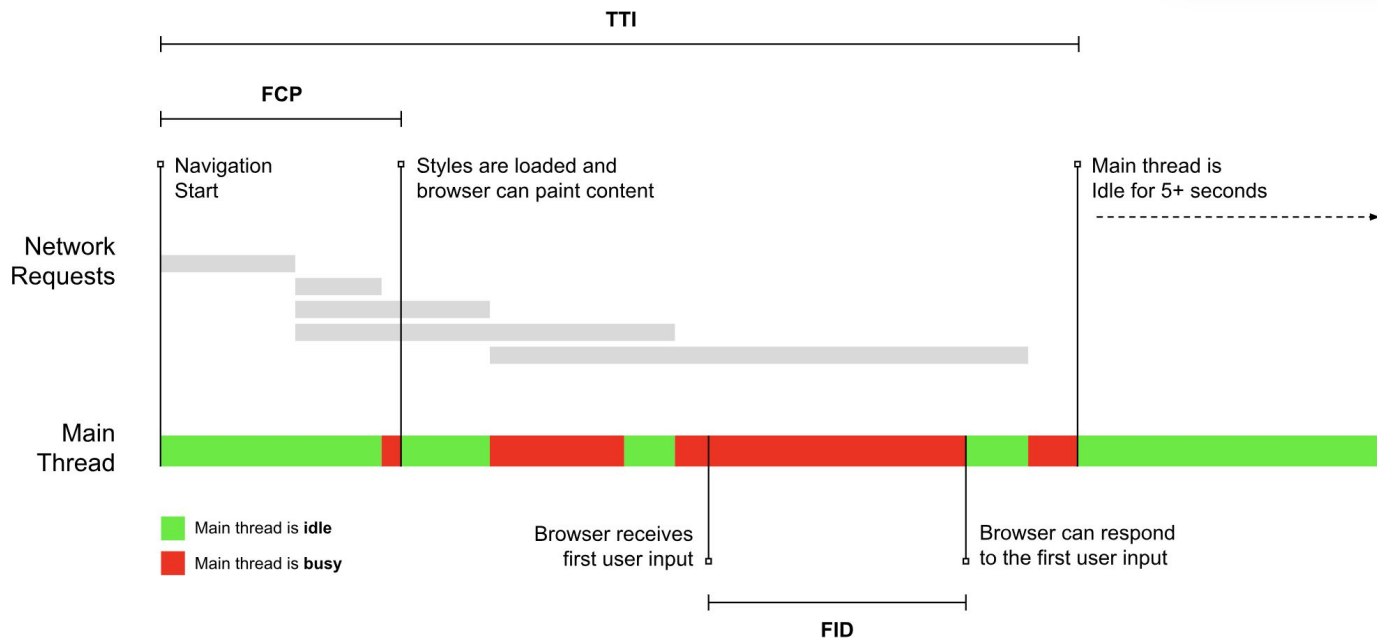
is it useful?

is it usable?



Because without JS  
your site can be unusable...we **also** care about  
**how fast JavaScript is executed**





### First Input Delay :

If a user tries to interact with the page during that time (e.g. click on a link), there will likely be a **delay between when the click happens and when the main thread is able to respond.**

← Goals

#1 : Visually render the site asap

#2 : JS execution should be fast



## Lab data

(aka metrics from the lab)



## Field data

(aka metrics from the wild)



## Test a website's performance

[Advanced Testing](#)[Simple Testing](#)[Visual Comparison](#)[Traceroute](#)**Test Location**[Select from Map](#)**Browser****Advanced Settings ▶**

3 runs, First View only, 4G connection

[START TEST](#)

Provided by

**INTERNETVIKINGS**  
SERVICES FOR INTERNET PROFESSIONALS

Run a free website speed test from multiple locations around the globe using real browsers (IE and Chrome) and at real consumer connection speeds. You can run simple tests or perform advanced testing including multi-step transactions, video capture, content blocking and much more. Your results will provide rich diagnostic information including resource loading waterfall charts, Page Speed optimization checks and suggestions for improvements.

If you have any performance/optimization questions you should visit the [Forums](#) where industry experts regularly discuss Web Performance Optimization.

### Recent Industry Blog Posts

Managing DNS Records For The People With Cloudflare Apps  
Traffic Acceleration with Cloudflare Mobile SDK  
How my team wrote 12 Cloudflare apps with fewer than 20 lines of code  
Why performance matters  
More consistent LuaJIT performance  
[more...](#)

### Recent Discussions

Problem with Hong Kong location - Timeout  
render-blocking Javascript Wordpress  
TimeToFirstByte value between Pingdom & WebPageTest  
Android device's Document Complete Error  
addition to CDN list  
[more...](#)

### WebPagetest Partners

# Web Page Performance Test for

<https://diversity.google/>

 From: Stockholm, Sweden - Chrome - Emulated Nexus 5 - 4G - Mobile  
 17/12/2018, 11:41:21

[Summary](#) [Details](#) [Performance Review](#) [Content Breakdown](#) [Domains](#) [Processing Breakdown](#) [Screen Shot](#) [Image Analysis](#) [Request Map](#)

Tester: vps1.webpagetest.org-185.12.150.24

First View only

Test runs: 3

[Re-run the test](#)
[Raw page data](#) - [Raw object data](#)
[Export HTTP Archive \(.har\)](#)
[View CSI data](#)
[View Test Log](#) - [Lighthouse Test Log](#)



## Performance Results (Median Run)

	Load Time	First Byte	Start Render	Speed Index	First Interactive (beta)	Document Complete			Fully Loaded			
						Time	Requests	Bytes In	Time	Requests	Bytes In	Cost
First View ( <a href="#">Run 2</a> )	5.346s	0.764s	2.100s	3.472s	> 3.970s	5.346s	37	2,060 KB	5.555s	38	2,065 KB	\$\$\$\$\$

[Plot Full Results](#)

## Test Results

Run 1:

	Waterfall	Screen Shot	Video
First View (5.641s)  <a href="#">Timeline (view)</a> <a href="#">Processing Breakdown</a>  <a href="#">Trace (view)</a>		 <p>Our accelerated approach to diversity and inclusion</p> <p>Google's mission is to organize the world's information and make it universally accessible and</p>	<a href="#">Filmstrip View</a> - <a href="#">Watch Video</a>

## Web Page Performance Test for

<https://diversity.google/>

From: Stockholm, Sweden - Chrome - Emulated Nexus 5 - 4G - Mobile  
 17/12/2018, 11:41:21

[Need help improving?](#)

**Summary** Details Performance Review Content Breakdown Domains Processing Breakdown Screen Shot Image Analysis Request Map

Tester: vps1.webpagetest.org-185.12.150.24

First View only

Test runs: 3

[Re-run the test](#)[Raw page data](#) - [Raw object data](#)[Export HTTP Archive \(.har\)](#)[View CSI data](#)[View Test Log](#) - [Lighthouse Test Log](#)

## Performance Results (Median Run)

	Load Time	First Byte	Start Render	Speed Index	First Interactive (beta)	Document Complete			Fully Loaded			
						Time	Requests	Bytes In	Time	Requests	Bytes In	Cost
First View ( <a href="#">Run 2</a> )	5.346s	0.764s	2.100s	3.472s	> 3.970s	5.346s	37	2,060 KB	5.555s	38	2,065 KB	\$\$\$\$\$

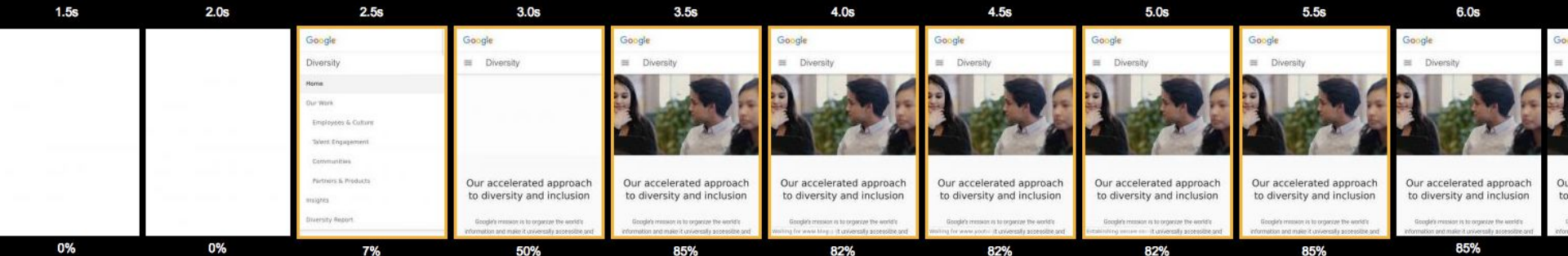
[Plot Full Results](#)

## Test Results

Run 1:

	Waterfall	Screen Shot	Video





☐ Slow Motion

Create Video

Advanced customization options...

Export filmstrip as an Image...

Thumbnail Size

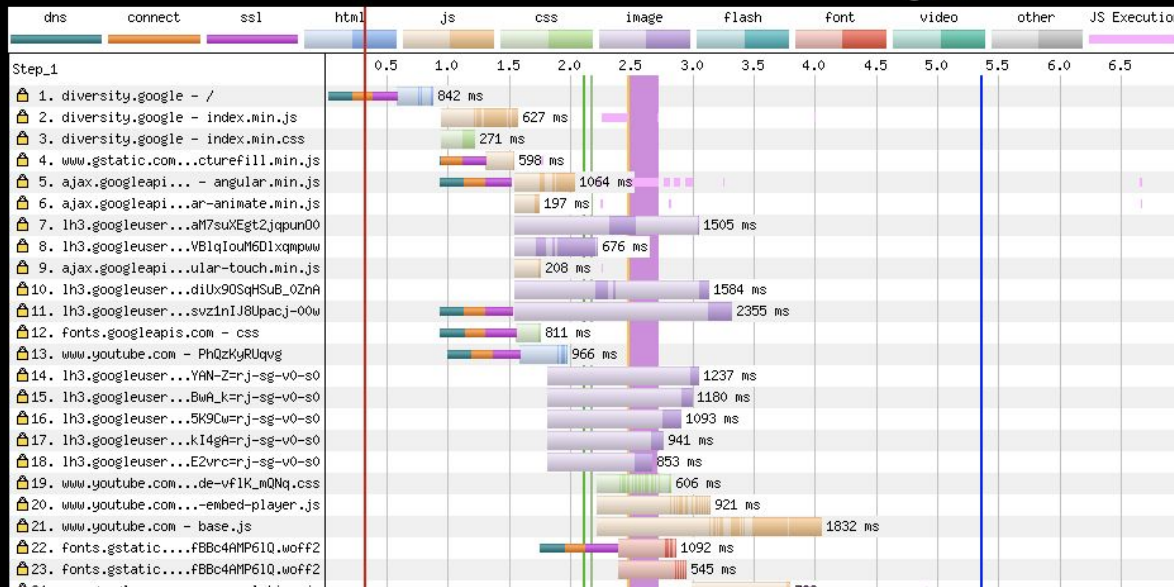
Thumbnail Interval

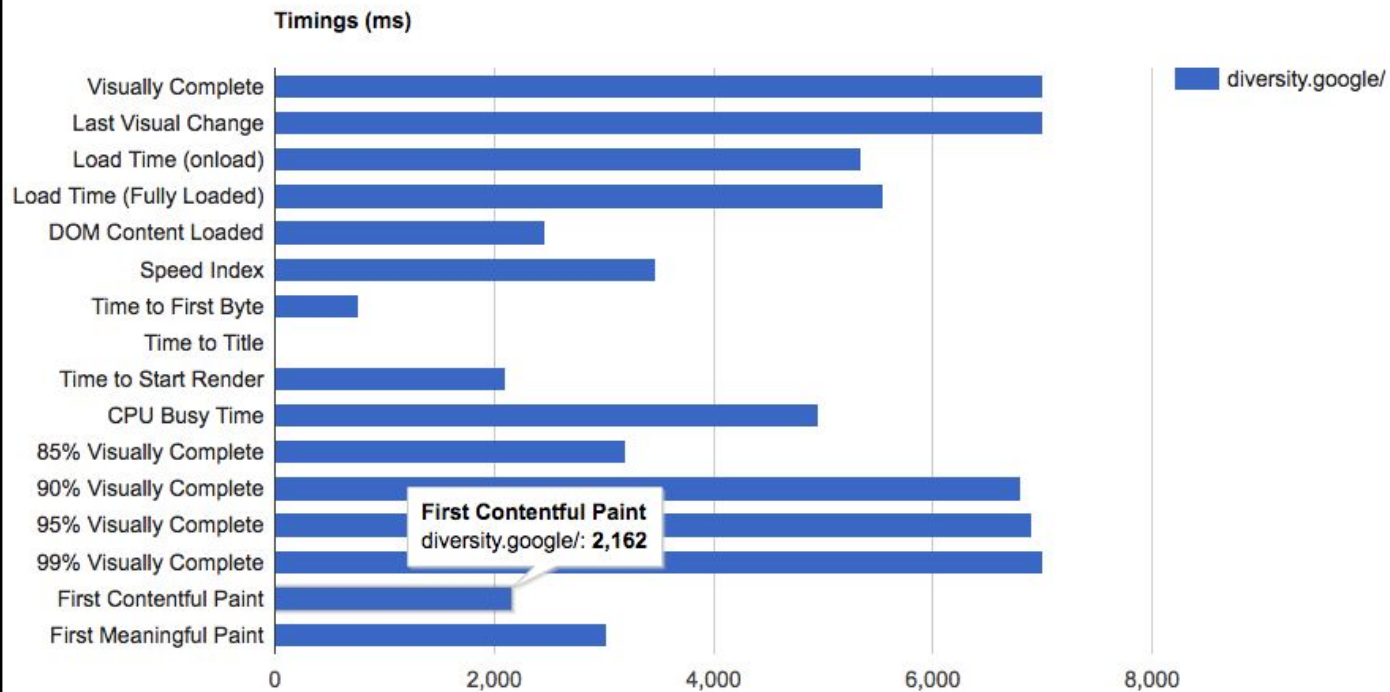
Comparison End Point

- ☐ Small
- ☐ Medium
- ☒ Large

- ☐ 60 FPS
- ☐ 0.1 sec
- ☒ 0.5 sec
- ☐ 1 sec
- ☐ 5 sec

- ☒ Visually Complete
- ☐ Last Change
- ☐ Document Complete
- ☐ Fully Loaded







## Web Page Performance Test for

<https://diversity.google/>

From: Stockholm, Sweden - Chrome - Emulated Nexus 5 - 4G - Mobile  
 17/12/2018, 11:41:21

[Need help improving?](#)
[Summary](#)
[Details](#)
[Performance Review](#)
[Content Breakdown](#)
[Domains](#)
[Processing Breakdown](#)
[Screen Shot](#)
[Image Analysis](#)
[Request Map](#)

Tester: vps1.webpagetest.org-185.12.150.24

First View only

Test runs: 3

[Re-run the test](#)[Raw page data](#) - [Raw object data](#)[Export HTTP Archive \(.har\)](#)[View CSI data](#)[View Test Log](#) - [Lighthouse Test Log](#)

## Performance Results (Median Run)

	Load Time	First Byte	Start Render	Speed Index	First Interactive (beta)	Document Complete			Fully Loaded			
						Time	Requests	Bytes In	Time	Requests	Bytes In	Cost
First View ( <a href="#">Run 2</a> )	5.346s	0.764s	2.100s	3.472s	> 3.970s	5.346s	37	2,060 KB	5.555s	38	2,065 KB	\$\$\$\$\$

[Plot Full Results](#)

## Test Results

Run 1:

Waterfall	Screen Shot	Video

# Use Google Analytics for RUM monitoring



## Perfume.js

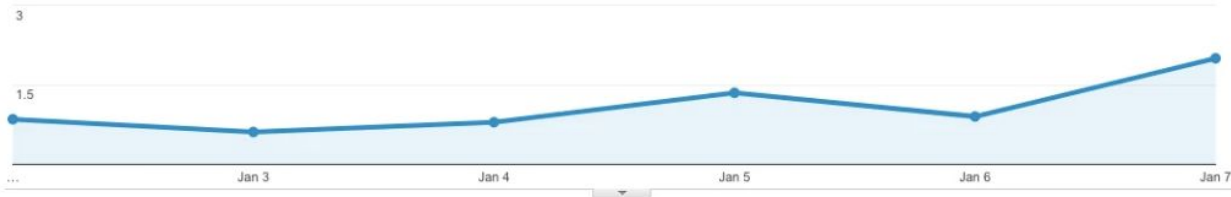
JavaScript library that measures First (Contentful) Paint ([FP/FCP](#)) and First Input Delay (FID). Annotates components' performance for Vanilla and Angular applications, into the DevTools timeline. Reports all the results to Google Analytics or your favorite tracking tool.

```
const perfume = new Perfume({
  googleAnalytics: {
    enable: true,
    timingVar: "userId"
  }
});
```



Google Analytics

● Avg. User Timing (sec)



Primary Dimension: **Timing Category** Timing Variable Timing Label

Plot Rows

Secondary dimension

Sort Type: Default

advanced

<input type="checkbox"/>	Timing Category ?	Avg. User Timing (sec) ?	User Timing Sample ?
		0.95 Avg for View: 0.95 (0.00%)	154 % of Total: 100.00% (154)
<input type="checkbox"/>	1. firstPaint	1.21	76 (49.35%)
<input type="checkbox"/>	2. fibonacci	<0.01	37 (24.03%)
<input type="checkbox"/>	3. firstContentfulPaint	1.38	16 (10.39%)
<input type="checkbox"/>	4. togglePopover	0.02	14 (9.09%)
<input type="checkbox"/>	5. timeToInteractive	2.91	11 (7.14%)

# Performance Objectives across the whole site

Is this happening?



First Contentful Paint

 < 2000ms

Is it useful?



Speed Index

 < 3000ms

Is it usable?



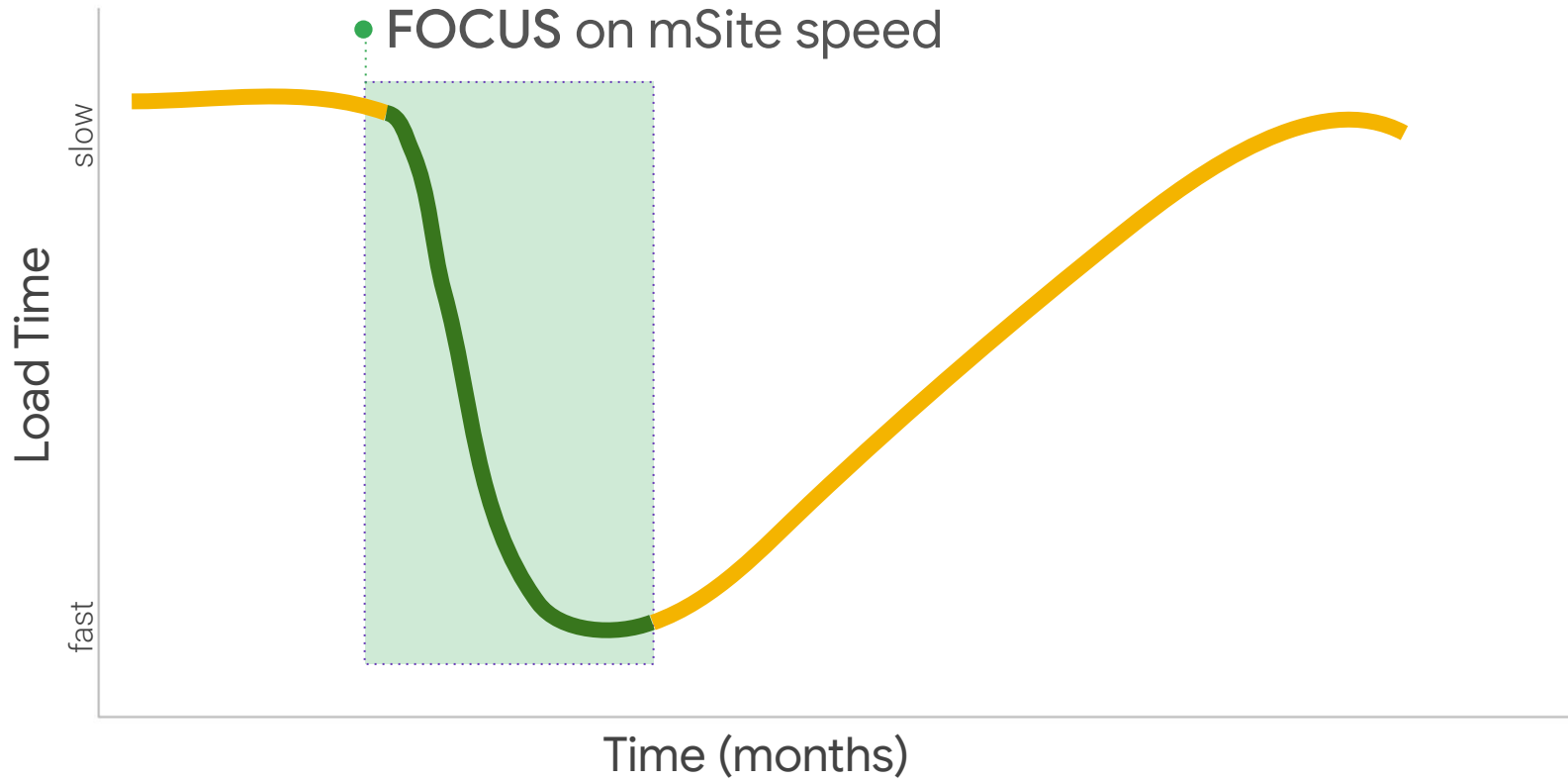
Time to Interactive

 < 5000ms

User-centric performance metrics

# Performance budgets





## Examples of budgets

You should have a budget in place for different types of pages on your site since the content will vary. For example:

- Our product page must ship less than 170 KB of JavaScript on mobile
- Our search page must include less than 2 MB of images on desktop
- Our home page must load and get interactive in < 5 s on slow 3G on a Moto G4 phone
- Our blog must score > 80 on Lighthouse performance audits





```
performance: {  
  maxAssetSize: 10000,  
  maxEntrypointSize: 30000,  
  hints: 'error'  
},
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: node



```
[126] multi babel-polyfill ./src/js/main.js 40 bytes {0} [built]
[329] ./src/js/main.js 3.1 KiB {0} [built]
[330] ./src/css/app.scss 39 bytes {0} [built]
      + 331 hidden modules
```

ERROR in asset size limit: The following asset(s) exceed the recommended size limit (9.77 KiB).  
This can impact web performance.

Assets:

```
900f8add35323c44ee840c611ed9a7a5.ttf (89 KiB)
pattern.bd0f788.svg (69.6 KiB)
app.fa9b4efe33c0ac934164.css (48.7 KiB)
app.bundle.e7080a6f56e54560eb26.js (91.8 KiB)
index.html (13.8 KiB)
```

ERROR in entrypoint size limit: The following entrypoint(s) combined asset size exceeds the recommended limit (29.3 KiB). This can impact web performance.

Entrypoints:

```
app (141 KiB)
  app.fa9b4efe33c0ac934164.css
  app.bundle.e7080a6f56e54560eb26.js
```

# Lighthouse Bot

This repo contains the frontend and backend for running Lighthouse in CI and integration with Github Pull Requests. An example web service is hosted for demo purposes.

**Note:** This repo was previously named "lighthouse-ci".

## Auditing GitHub Pull Requests

Please note: This drop in service is considered **Beta**. There are no SLAs or uptime guarantees. If you're interested in running your own CI server in a Docker container, check out [Running your own CI server](#).

Lighthouse can be setup as part of your CI on **Travis only**. As new pull requests come in, the **Lighthouse Bot tests the changes and reports back the new score**.

and more submitted by pushing to the `main` branch with subsequent merges.



**All checks have passed**

[Hide all checks](#)

2 successful checks



**Lighthouse** — Passed. New Lighthouse score would be 100/100.

[Details](#)



**continuous-integration/travis-ci/pr** — The Travis CI build passed

[Details](#)



**This branch has no conflicts with the base branch**

Merging can be performed automatically.

Merge pull request



You can also [open this in GitHub Desktop](#) or view [command line instructions](#).



Write

Preview

AA B i



[github.com/GoogleChrome/eLabs/lighthousebot](https://github.com/GoogleChrome/eLabs/lighthousebot)

# Server Response Time

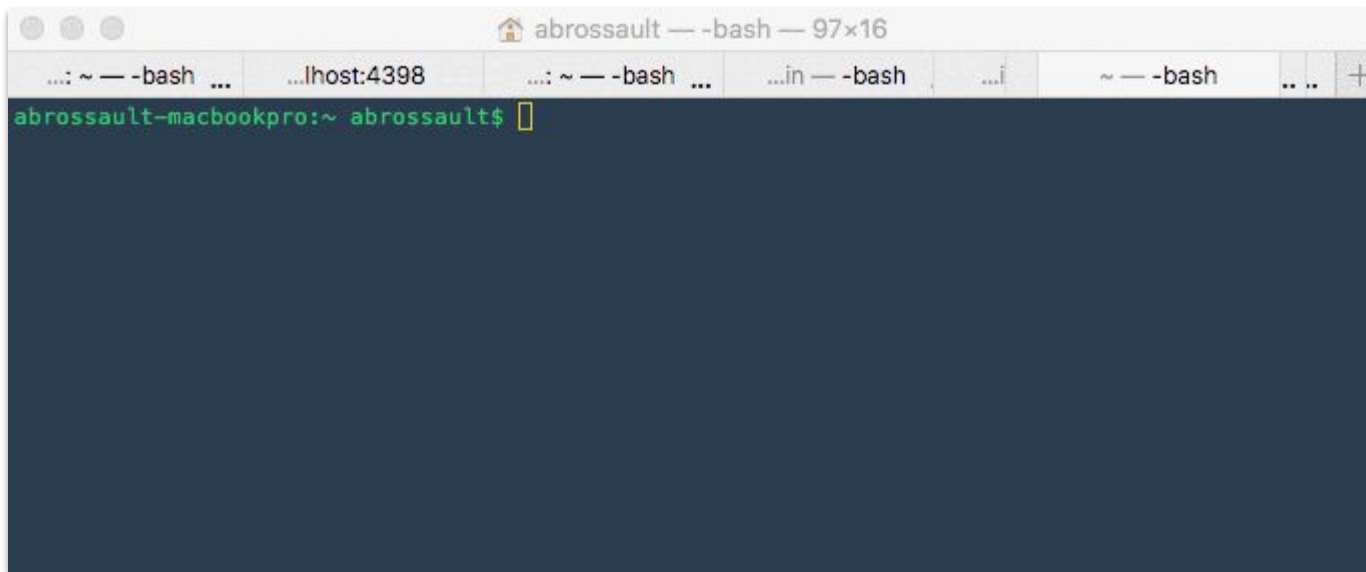


Your server should respond in  
**200ms** or less


# Quickly verify the Time To First Byte

```
curl -s -L -o /dev/null -w "  
  HTTP code : %{http_code} \n  
  Number of Redirects : %{num_redirects} \n  
  Last url : %{url_effective} \n  
  Look up time : %{time_namelookup} \n  
  Connect: %{time_connect} \n  
  TTFB: %{time_starttransfer} \n  
  🖱 Total time: %{time_total} \n \n" https://google.fr
```

# Quickly verify the Time To First Byte

A screenshot of a terminal window titled 'abrossault — -bash — 97x16'. The window has a dark blue background and a light gray title bar. The prompt 'abrossault-macbookpro:~ abrossault\$' is visible at the top left of the terminal area, followed by a yellow cursor. The terminal area is mostly empty, with some faint, illegible text visible in the background.

```
abrossault-macbookpro:~ abrossault$
```

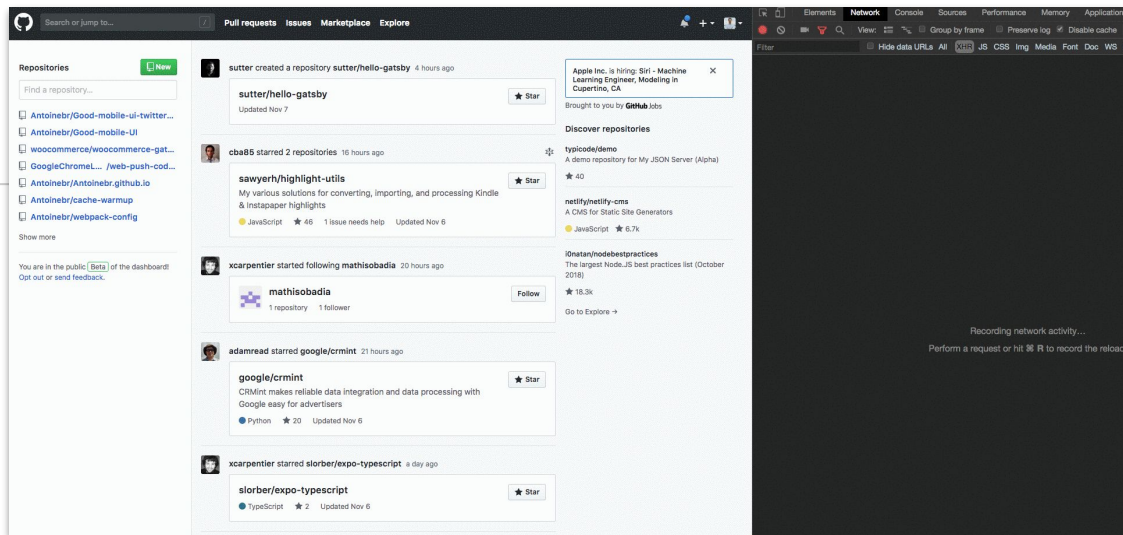
  
`npm install -g ttfb-test`

# Improve TTFB : strategies

**Look for uncached db requests**

**Move synchronous call to a web-services to frontend**

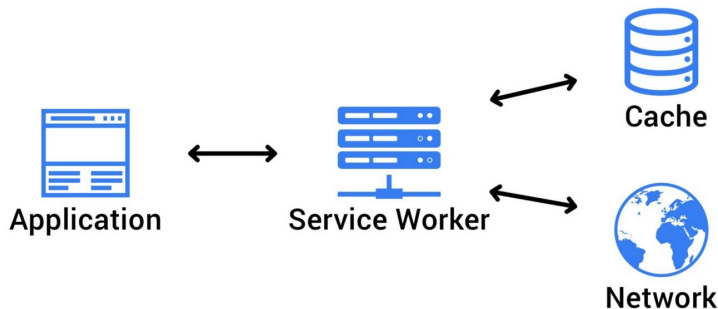
**Precache the slow routes with a Service Worker**





# Improve TTFB : Service Workers

Use the service-worker install event to cache less-critical but still important resources, such as assets for the routes your user is most likely to go to or below-the-fold content. If the install succeeds, these secondary resources will be available the instant they are needed.



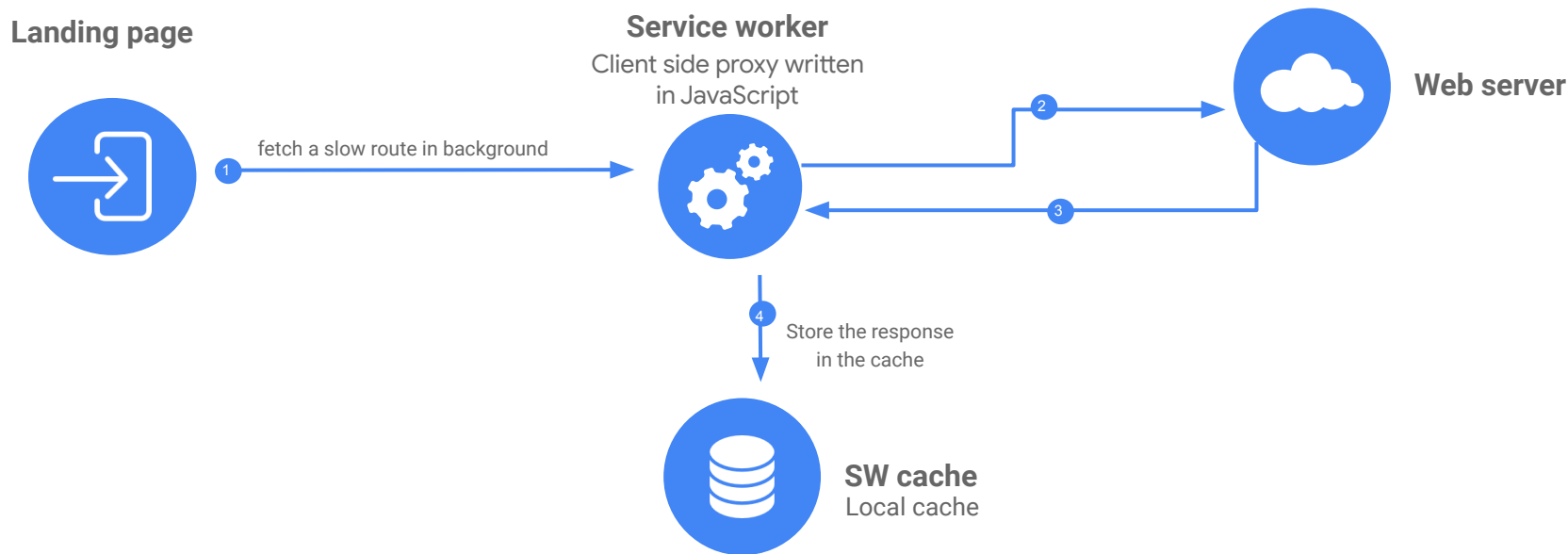
Register a  
Service Worker

```
(async () => {  
  if ("serviceWorker" in navigator) {  
    // we register our service worker  
    const registration = await navigator.serviceWorker.register('/sw.js');  
  
    // when our service worker is updated  
    registration.onupdatefound = () => {  
      // when our service worker is updated  
      registration.installing.onstatechange = function () {  
        console.log(`Service worker... ${this.state}`);  
      };  
    };  
  }  
})()  
.catch(e => console.log(`🤖 : ${e}`));
```

↙ We wait for our SW  
to be installed

```
self.addEventListener('install', async event => {  
  console.log('[SW] installing SW', event);  
  
  const myPWAdemoCache = await caches.open("my-pwa-demo");  
  
  await myPWAdemoCache.addAll(['/','index.html','script.js','style.css']);  
});
```

# Improve TTFB : Service Workers

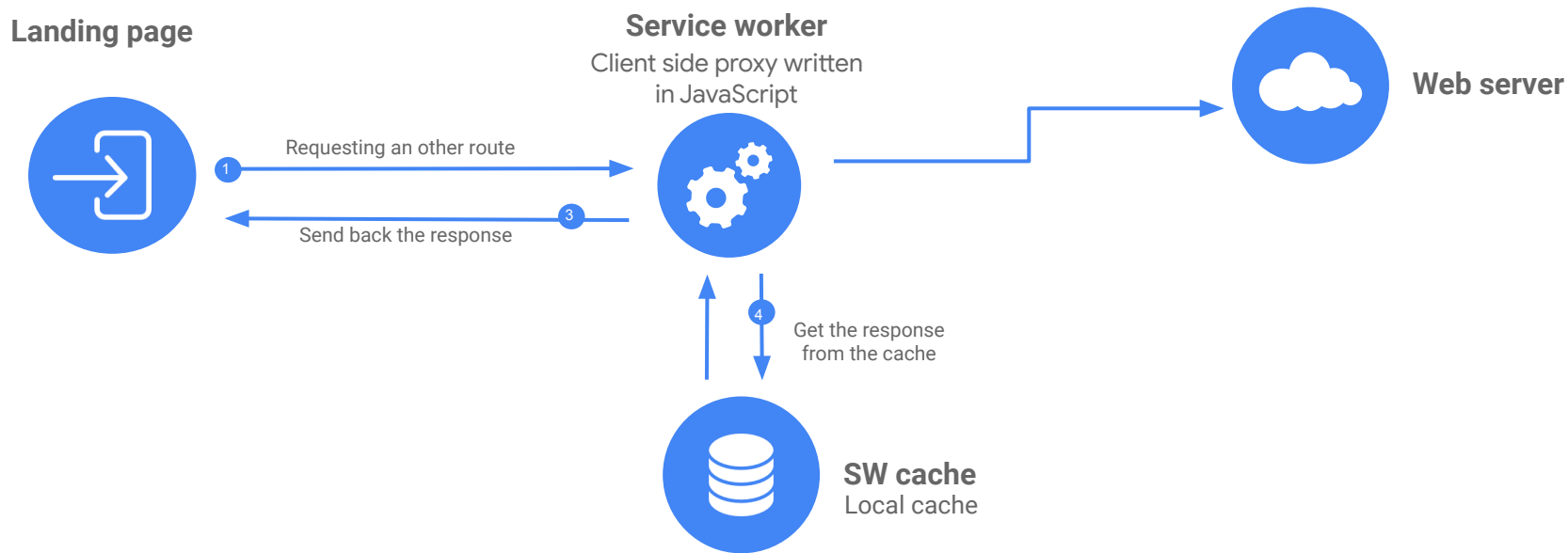


[sw-preload-on.glitch.me](https://sw-preload-on.glitch.me)

[sw-preload-off.glitch.me](https://sw-preload-off.glitch.me)




# Improve TTFB : Service Workers





*Faster subsequent page-loads by  
prefetching in-viewport links  
during idle time*



```
if ("serviceWorker" in navigator) {  
  (async () => {  
    const registration = await navigator.serviceWorker.register('/sw.js');  
    registration.onupdatefound = () => {  
      console.log('Service worker...');  
      registration.installing.onstatechange = function () {  
        if (this.state === "activated") quicklink();  
      };  
    }  
  })();  
}
```

←  
When the Service Worker is  
activated we store in the cache  
the links which are in the viewport

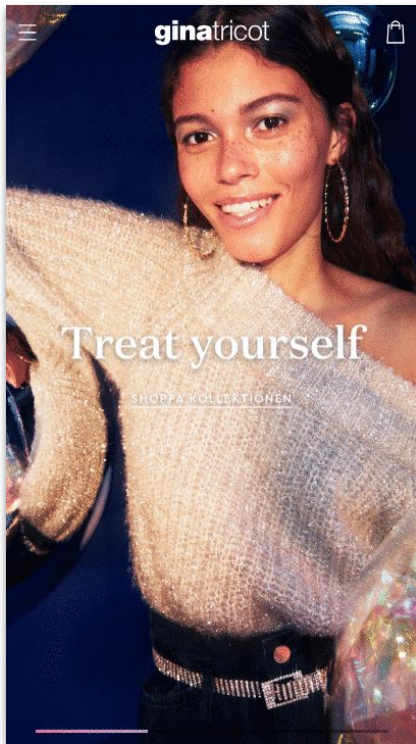
No slider





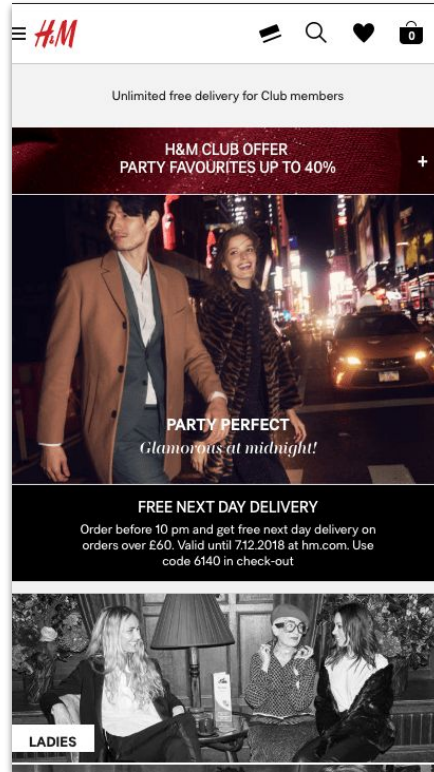
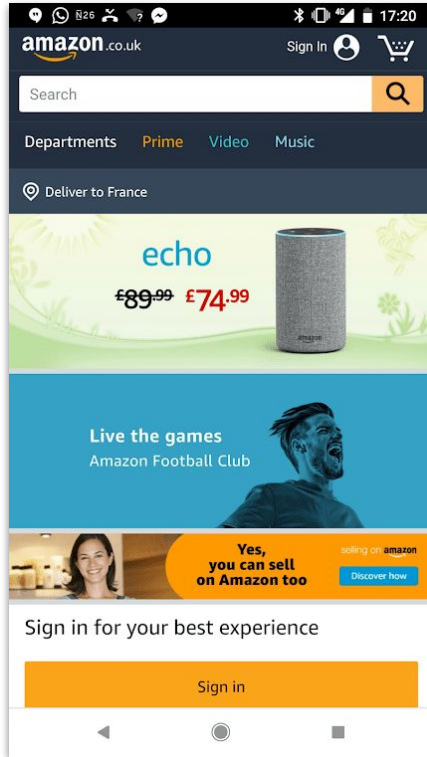
**Do not use carousel**  
on mobile

# Why sliders should be banned on mobile?



- ✗ Bad performance: speed killer
- ✗ First slide will get all the clicks - CTR
- ✗ Movement leads to distraction
- ✗ Move when you read
- ✗ Looks like ads
- ✗ Need multiple images and JavaScript

✓ Replace with static images



# Make the above the fold less JavaScript dependent



*A static image is loaded and the carousel loads in the background*

# CSS




In the Command Menu  
(Cmd + Shift + P / Ctrl + Shift + P)



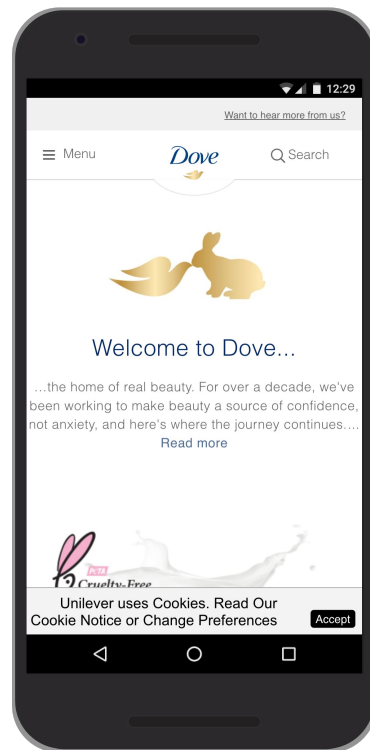
# Clean your CSS

*Up to 96% unused css code*

Type	Total B...	Unused Bytes	
CSS	2 245 604	2 162 482 96.3 %	

```
509 @media (min-width: 1199px) {  
510   .dl-horizontal dt {  
511     float: left;  
512     width: 160px;  
513     clear: left;  
514     text-align: right;  
515     overflow: hidden;  
516     text-overflow: ellipsis;  
517     white-space: nowrap;  
518   }
```

*Code not used on the landing page but still render blocking*





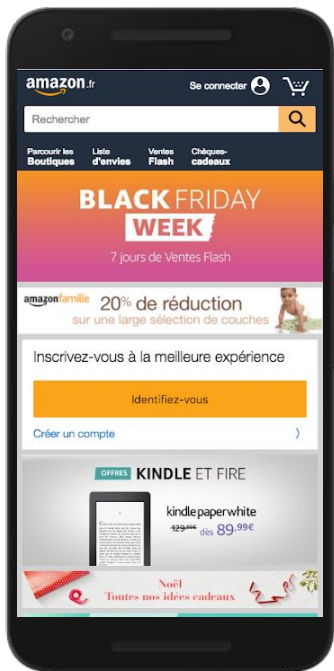
# Beware of nesting in CSS with SASS

*The Inception Rule:  
don't go more than  
four levels deep.*

```
.VehicleSelector .BazaarVoiceRating .bv-cleanslate .bv-shared .bv-inline-rating-container .bv-  
rating,.VehicleSelector .BazaarVoiceRating .bv-cleanslate .bv-shared .bv-inline-rating-container .bv-rating-  
label,.VehicleSelector .BazaarVoiceRating .bv-cleanslate .bv-shared .bv-inline-rating-container .bv-text-  
link {  
  color: #005a2b !important  
}  
  
.VehicleSelector .BazaarVoiceRating .bv-cleanslate .bv-shared .bv-inline-rating-container .bv-rating-stars-  
off {  
  color: #cdd1d1 !important  
}
```



# Split critical - non-critical

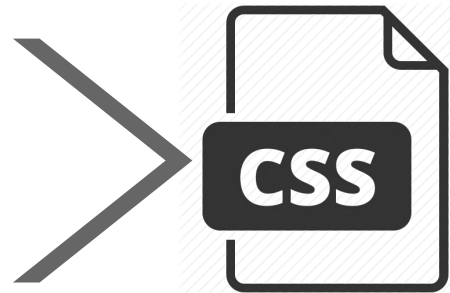


Critical CSS

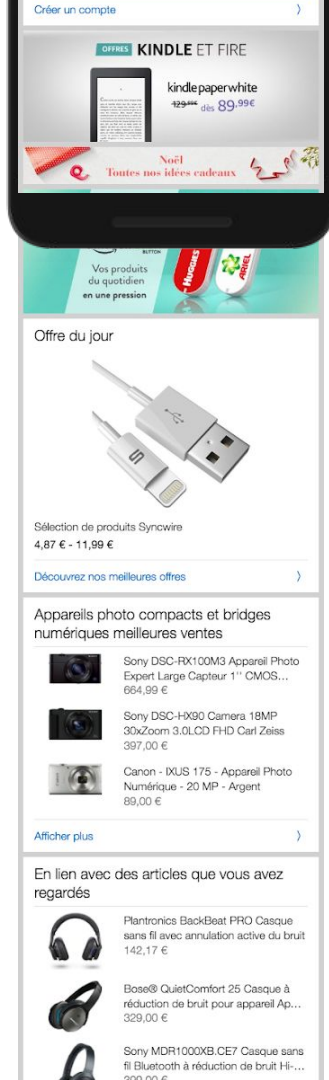
```
const critical = require('critical');

critical.generate({
  src: "https://amazon.fr",
  dest: "critical.css",
  width: 412,
  height: 732
})
```

extraction (node.js)



critical.css



Non-critical CSS

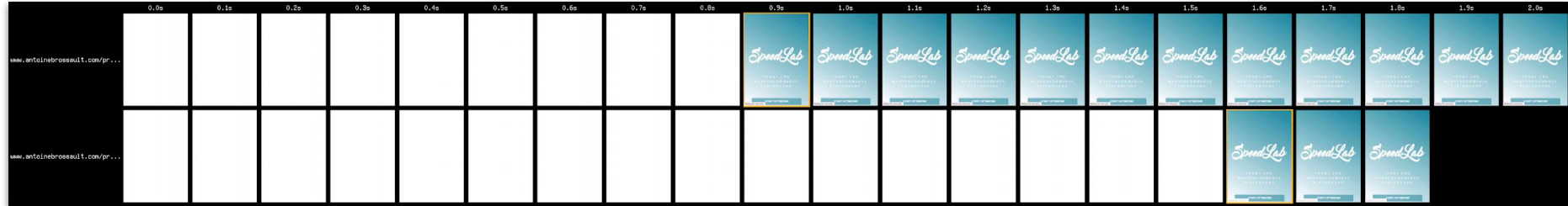
# Split critical - non-critical

```
<script>loadCSS("css/app.css");</script>
<script>loadCSS("css/sprite.css");</script>
```

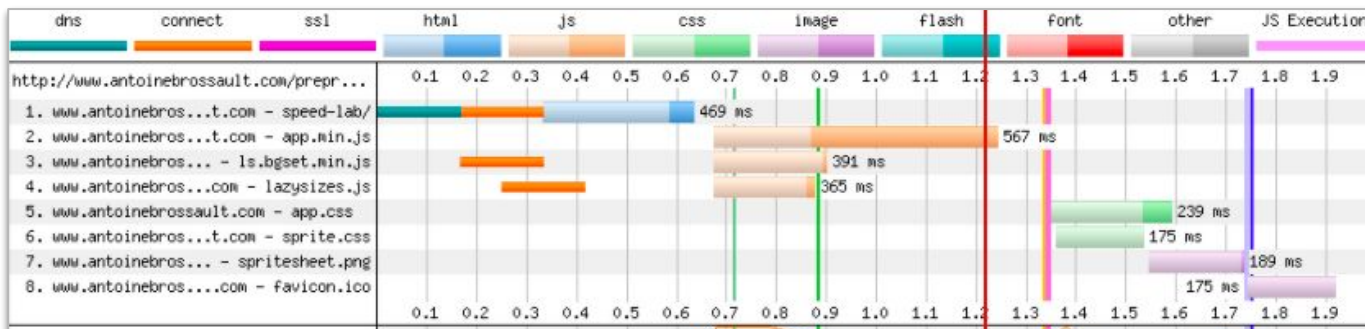
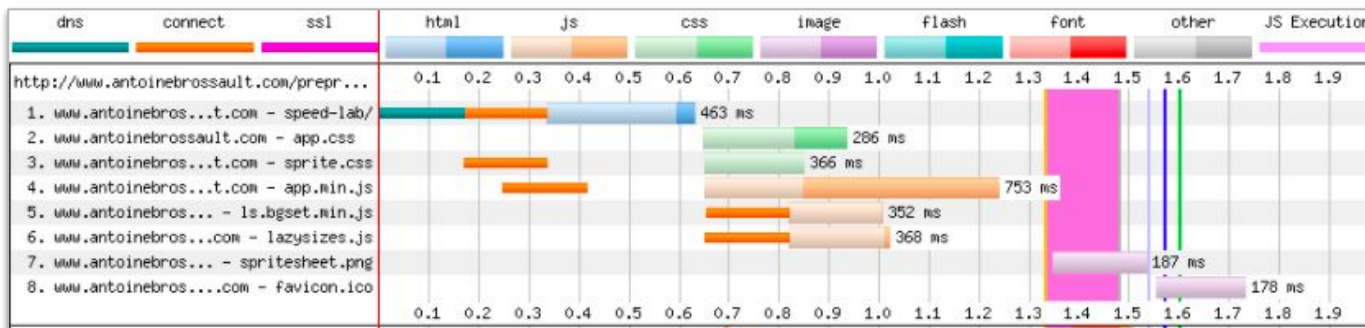
github.com/addyosmani/critical



# Split critical - non-critical



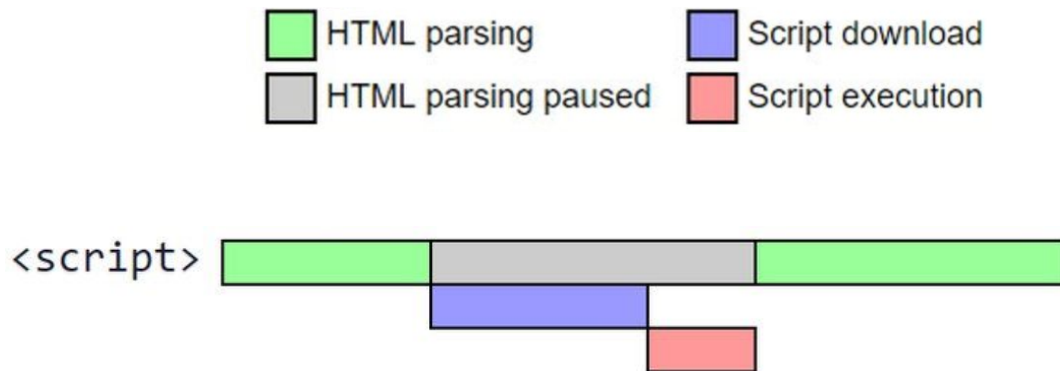
# Split critical - non-critical



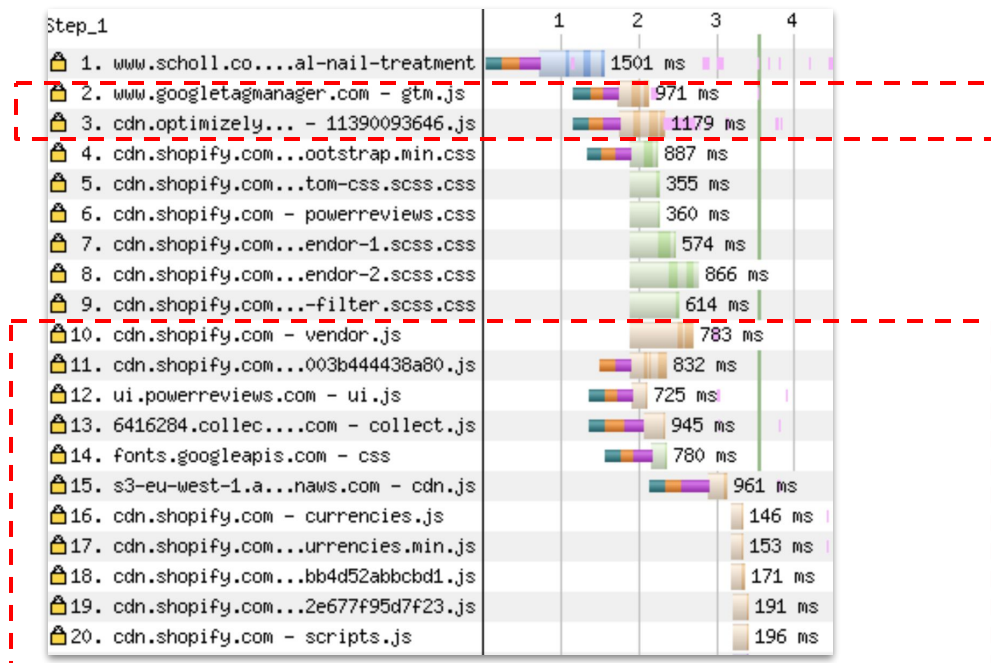
# JavaScript



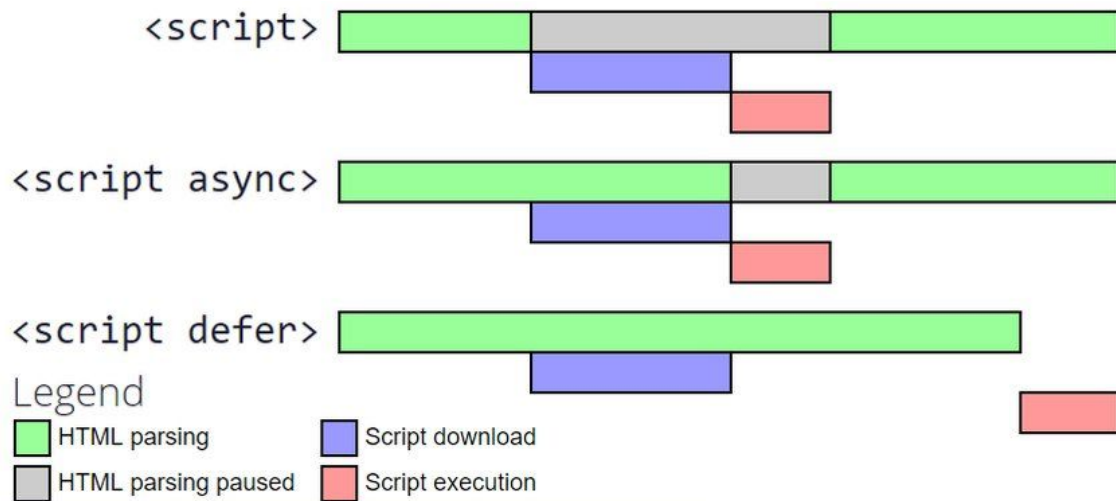
# Avoid render-blocking scripts



# Avoid render-blocking scripts



# Optimize JavaScript loading





# Optimize JavaScript loading

## **Use defer and async**

**Defer** : if loading order matters

**Async** : if the script is standalone

Remove the AB testing script  
when you are not  
doing mobile testing

# No AB-Testing on mobile ?



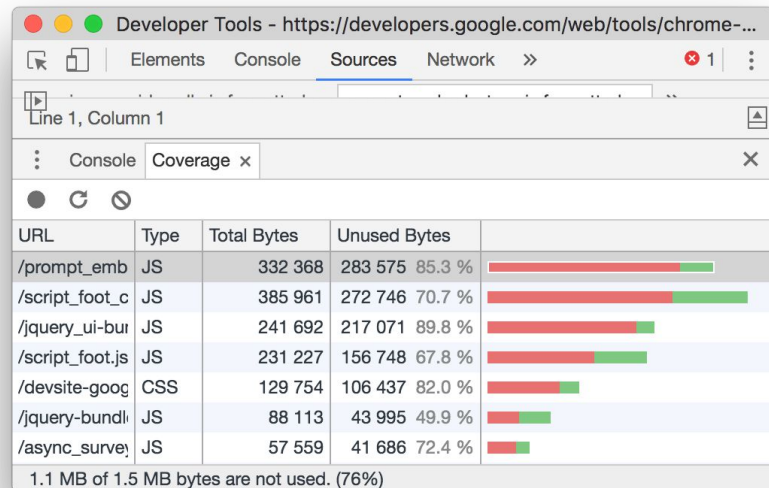
```
$detect = new Mobile_Detect;  
  
if($detect->isMobile()){  
    // do stuff for mobile devices  
}else {  
    // do stuff for non mobile  
}
```

Your should keep JavaScript  
payloads at minimum

*In the Command Menu  
(Cmd + Shift + P / Ctrl + Shift + P)*



Find unused CSS and JS  
with the **Coverage** tab



## Analyze your bundle to find opportunities



# Be sure your app leverages Treeshaking



You have to use a bundler ( Webpack, Rollup, Parcel )



You have to use the ES6 syntax for import export `import {camelCase} from 'lodash'`



If you use Babel Webpack's built-in tree shaking works on ES6 module syntax only. If you're using Babel's defaults settings, Babel will compile ES6 modules to CommonJS modules, leaving nothing for Webpack to work with.

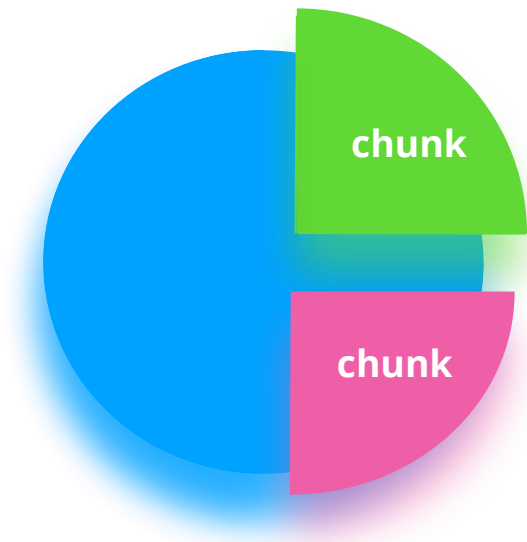
# Be sure your app leverages Treeshaking

```
module: {  
  rules: [  
    {  
      test: /\.js?$/,  
      exclude: /node_modules/,  
      loader: 'babel-loader',  
      query: {  
        // it's important to set modules: false  
        // otherwise ES2017 import / export are converted to require() (common js syntax)  
        presets: [ ["env", { modules: false } ], "stage-0" ]  
      }  
    }  
  ],  
}
```



# Reduce JavaScript payload : How to fix ?

Split your bundle





```
import('module')  
.then(module => module.default)  
.then(module => doSomethingCool(module))
```

✓ *you will save ~6 sec*



```
const $form = document.querySelector('form');

if ($form) {

  // we dynamically import the module
  (async () => {

    const validator = await import('validator');

    // use validator

  })();

}
```



```
export default new Router({
  mode: 'history',
  base: process.env.BASE_URL,
  routes: [
    {
      path: '/',
      name: 'dashboard',
      component: Dashboard
    },
    {
      path: '/team',
      name: 'team',
      // route level code-splitting
      // this generates a separate chunk (Team.[hash].js) for this route
      // which is lazy-loaded when the route is visited.
      component: () => import(/* webpackChunkName: "Team" */ './views/Team.vue')
    }
  ]
})
```

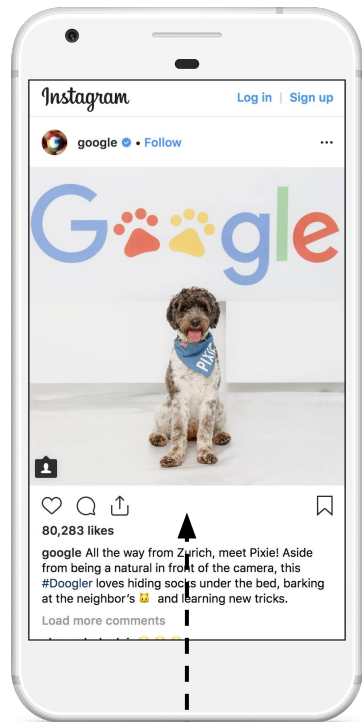
# Images



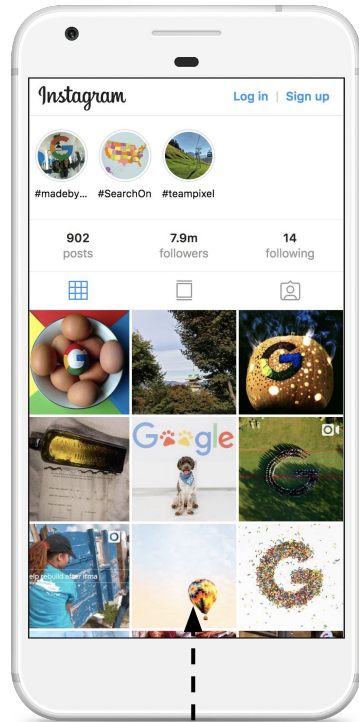
Load images to the correct size

# Serve Multiple Image Sizes

- For most sites, serving **3-5 different sizes** of an image works best.



3 sizes



5 sizes

# Image Resizing Tooling

## npm packages:

### Sharp

- + **Fastest** image resizing tool
- Installation requires compiling C++/C

```
const sharp = require('sharp');
const fs = require('fs');
const directory = './images';

fs.readdirSync(directory).forEach(file => {
  sharp(`${directory}/${file}`)
    .resize(200, 100) // width, height
    .toFile(`${directory}/${file}-small.jpg`);
});
```



# Image Resizing Tooling

## npm packages:

### Jimp

+ Installation does **not**  
require compiling C++/C

```
const Jimp = require('jimp');
const fs = require('fs');
const directory = './images';
const imgs = fs.readdirSync(directory);

(async () => {
  for (let img of imgs) {
    const newImg = await Jimp.read(`${directory}/${img}`);
    await newImg.resize(200, 100);
    await newImg.write(`${directory}/${img}.min.jpg`);
  }
})();
```

# Serve Multiple Image Sizes

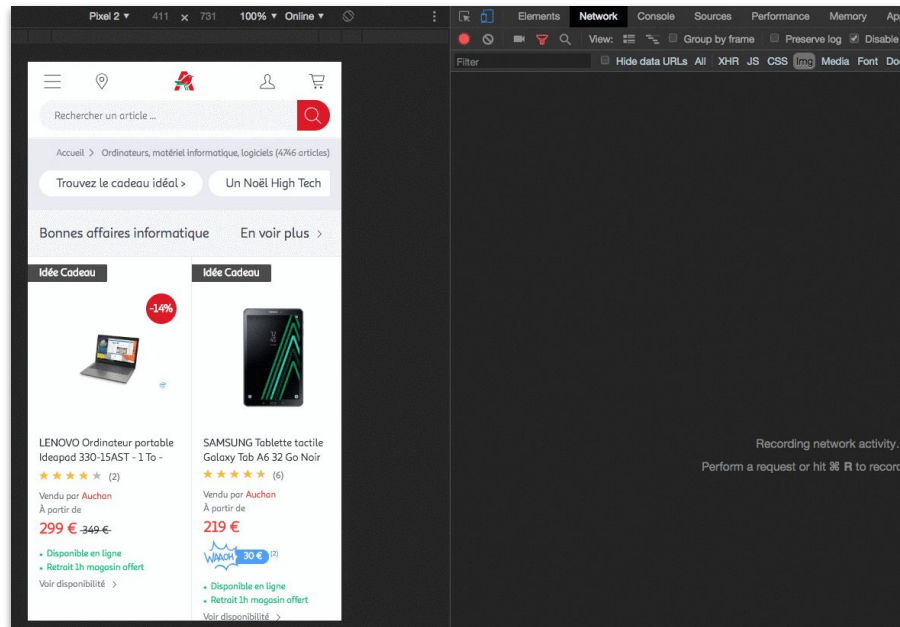
```

```

Lazyload your images

# Use Lazy Loading

Lazy loading allow you to trigger the image download before the user need it.

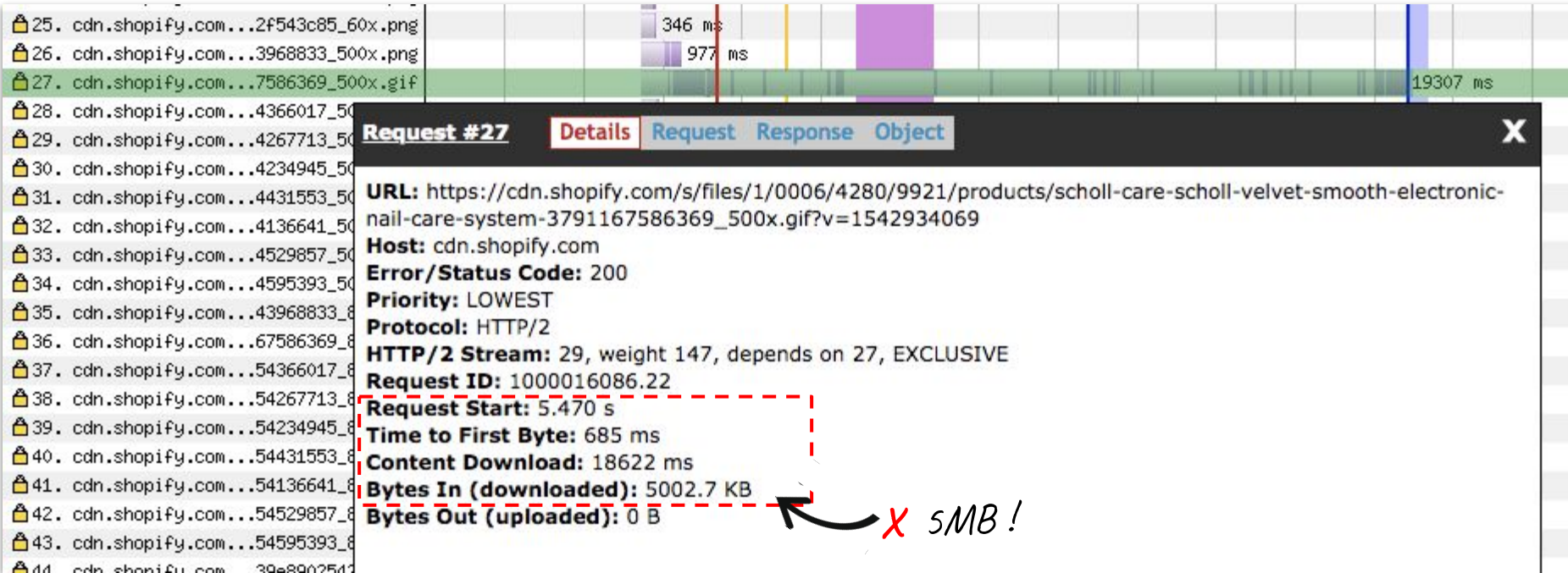


# Use Lazy Loading

```

```

Do not use GIFs  
use video instead



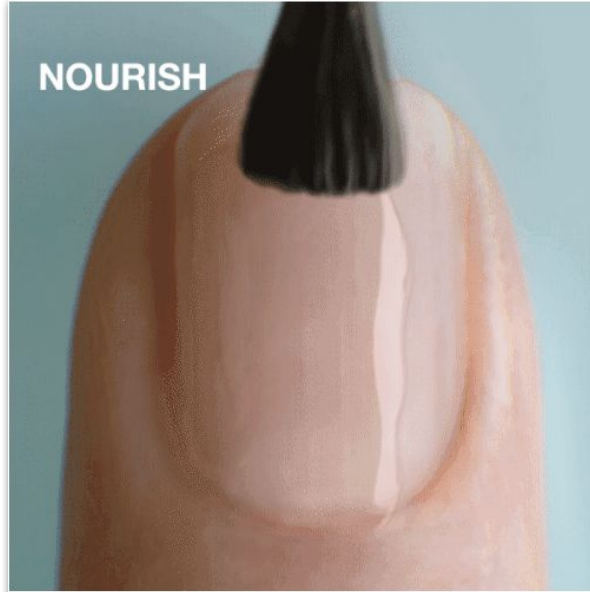


```
ffmpeg -i original.gif video.mp4
```





```
<video autoplay loop muted playsinline>  
  <source src="/video.mp4" type="video/mp4">  
</video>
```



gif : 5MB VS mp4 403KB



gif : 124KB VS mp4 14KB

# Compress your images

Lossless

Lossy

Image

Lossless

Im

*At a minimum, sites should be using  
lossless compression.*

Lossy

age

*Most sites would benefit from more  
aggressive (i.e. lossy) compression.*

# Lossy Compression



# Lossy Compression



# Lossy Compression

For most images, **80-85 quality will reduce file size by 30-40%** with minimal effect on image quality.



# Imagemin



# Imagemin

## Popular Imagemin Plugins

	JPEG	PNG	GIF	SVG	WebP
Lossless	imagemin- <b>jpegtran</b>	imagemin- <b>optipng</b>	imagemin- <b>gifsicle</b>	imagemin- <b>svgo</b>	imagemin- <b>webp</b>
Lossy	imagemin- <b>mozjpeg</b>	imagemin- <b>pngquant</b>	imagemin- <b>giflossy</b>		

```
imagemin(['source/*.jpg'], 'destination', {  
  plugins: [  
    imageminMozjpeg({quality: 80})  
  ]});
```

*Example Usage*

# Image compression issues



*x Original 155KB*

*Compressed with MozJPEG 75%  
87% smaller ( 20.6KB )*

# Preload important images



```
<link rel="preload" as="image" href="/myImportantImage.jpg" >
```



*Preload the image to  
trigger an early download*

# Fonts



# Improve font loading

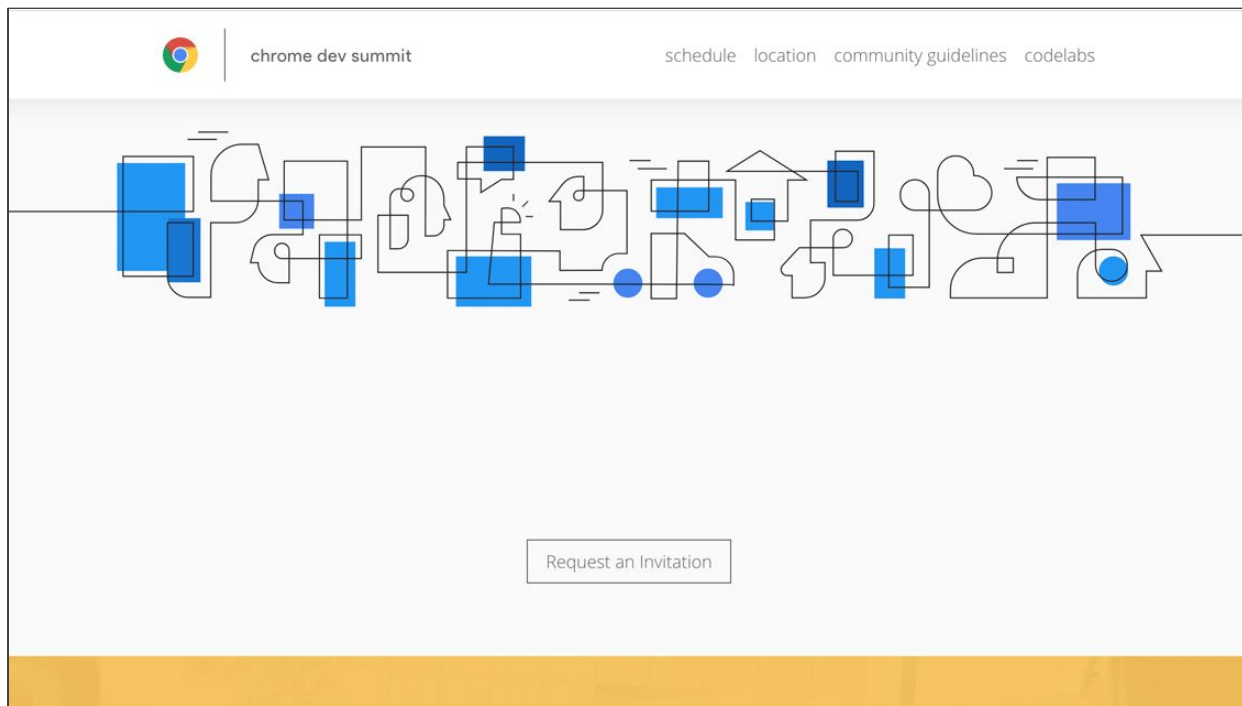
Generate a woff2

```
@font-face {  
  font-family: "Open Sans";  
  src: url("/fonts/OpenSans-Regular-webfont.woff2") format("woff2"),  
        url("/fonts/OpenSans-Regular-webfont.woff") format("woff"),  
        url("/fonts/OpenSans-Regular-webfont.otf") format("truetype");  
  font-display: swap;  
}
```

Avoid FOIT



# “Flash of Invisible Text” (FOIT)



# Improve font loading

Preload and use font-display swap

```
<link rel="preload" as="font" href="/font" crossorigin>
```

← Preload the font

```
@font-face {  
  font-family: 'SedgwickAve-Regular';  
  font-style: normal;  
  font-weight: 400;  
  src: url(/font) format('woff');  
  font-display: swap;  
}  
h2{  
  font-family: 'SedgwickAve-Regular', sans-serif;  
}
```

← Use font-display : swap

# Improve font loading

 *font-display : properties*

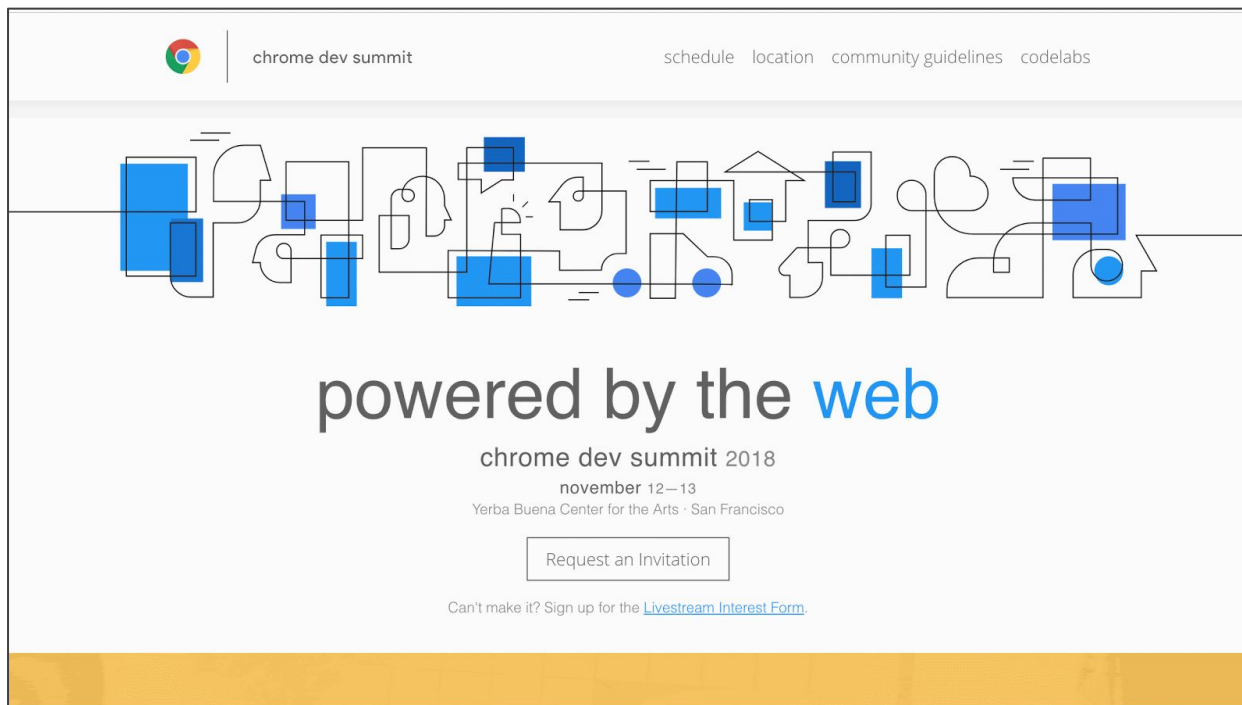
**Very slow** font (4s server delay)

**0.0 s**

**Medium slow** font (2s server delay)

**Fast** font (0s server delay)

# “Flash of Unstyled Text” (FOUT)



# Thank you



# Speed checklist

## Webperformance made simple

[speed-up my site](#)

### What to check and how to fix the issues

Let's fix issues with the "stack overflow" way to do it

[All](#)[JavaScript](#)[Images](#)[Backend](#)[Server](#)[Font](#)

Priority

Type

Difficulty

Title

1



HARD

Optimizing the first response :  
Time to first byte

TTFB measures the duration from the user or  
client making an HTTP request to the first ...

[see more](#)

ne-speedrace@google.com

Twitter : @antoineBr