

**Getting your Team **Passionate**  
About Web Performance  
to Achieve **Performant Web Apps****



Hi! I am [Nicolas Goutay](#). I work at [Theodo](#), a web consultancy based in Paris, New York & London. I build [JS](#) & [Python](#) web applications.

I run the [JAMStack Paris](#) meetup and help organize the [WeLoveSpeed](#) conference.

You can find me online (Twitter & GitHub) on [@phacks](#).

W E B P E R F



In the open space, no one can hear you scream.

February 2018, **3** Theodo web apps were performant.



January 2019, **8** Theodo web apps were performant.



Today, **27** Theodo web apps are performant.



# The Culture of Web Performance



# The Culture of Web Performance

 Structure

 Tooling

# The Culture of Web Performance

 Structure

 Tooling

 Knowledge



Leveraging **Lean** methodologies



# What is Lean?

Lean is a **systematic method** to **maximize customer value** while **minimizing waste**.

Helped propel Toyota from a small company to the **world's largest automaker** in ~50 years.

Its roots trace back to manufacturing, but it can be applied to any industry, including **digital product crafting**.

# Identifying the Value Stream

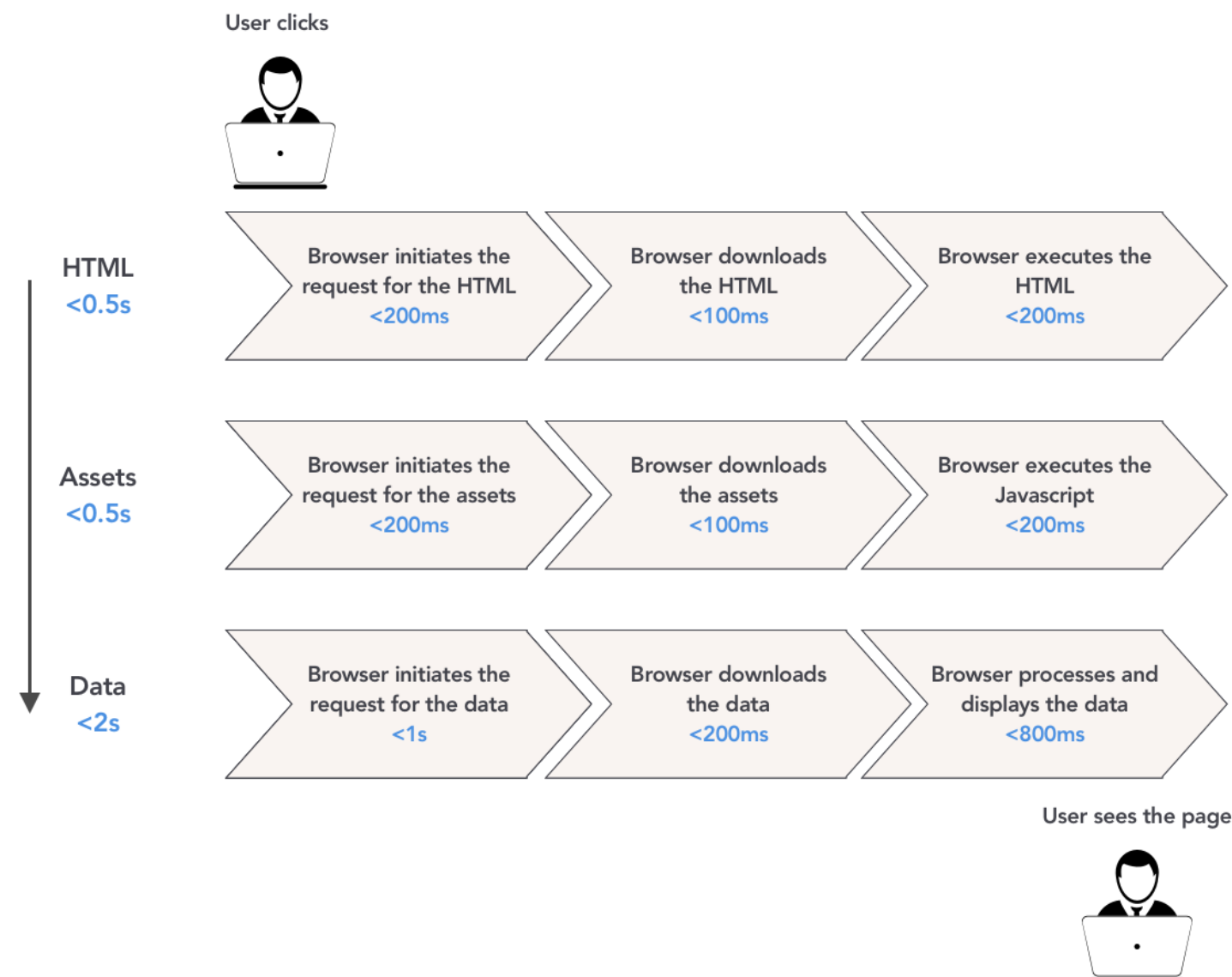
In Lean, the Value Stream is a representation of all the **value-added steps** in the process to go from the **raw materials** to the **finished product**.

It helps you **understand and visualize the whole process**, so that you can easily **see waste** and **fight it**.

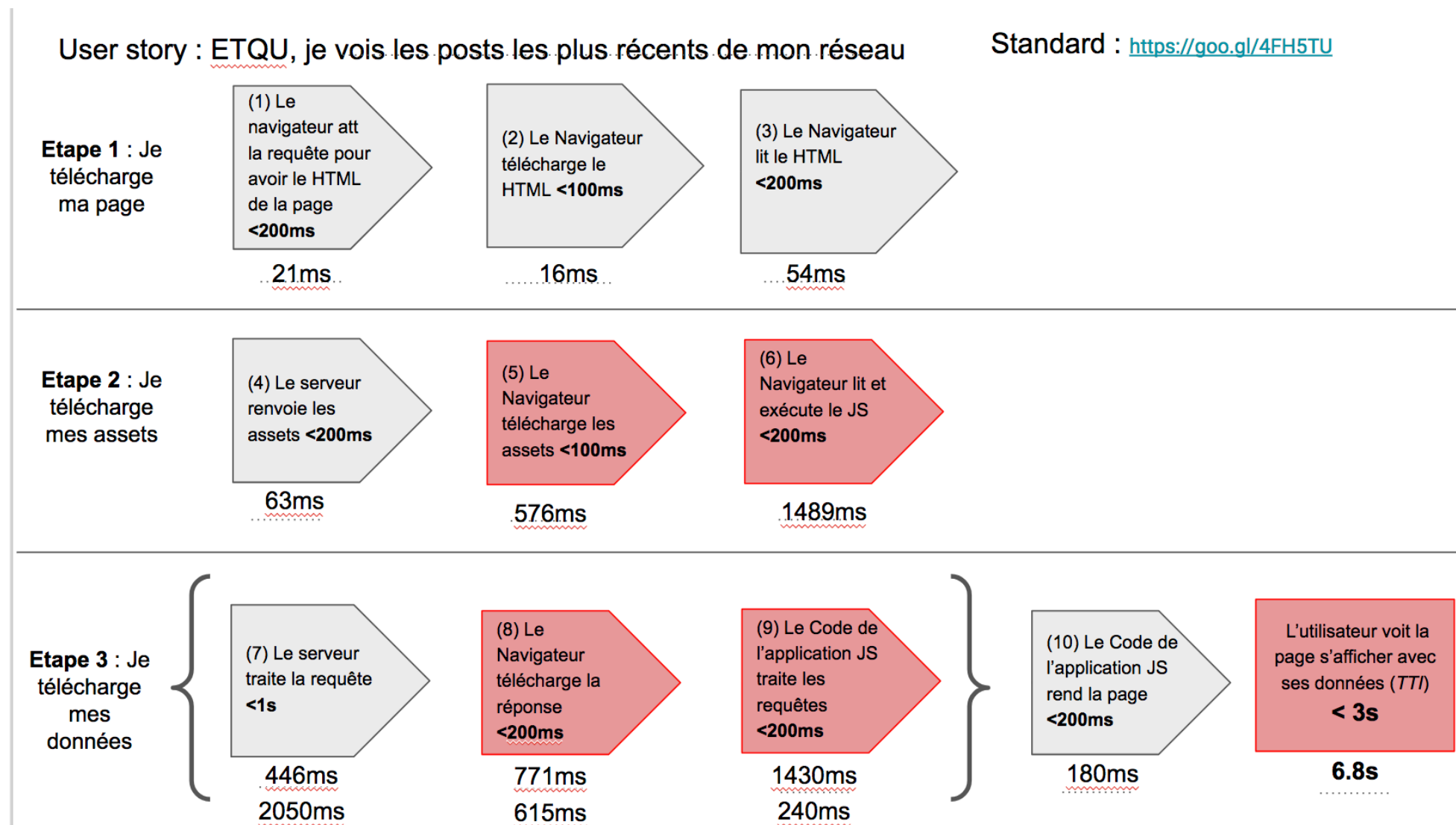
In Web Perf, the Value Stream would be **all the steps** for a website to go **from the initial user request** to it being **interactive**.

# Identifying the Value Stream

We mapped the Value Stream, from backend to frontend and back again, and set out to determine **maximum durations at every step**.



# Identifying the Value Stream



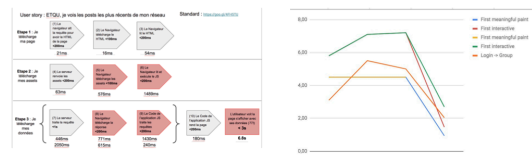
# Identifying the Value Stream

**Ffyn** — 6.8s ⇒ 2.5s ✓ — **Projet terminé**

ThibautC

"C'est instantané, ça fait pas pareil en démonstration quand tu montres une application ou demandes d'attendre et une où c'est instantané"

Richard Jones, PO Ffyn



#### Résumé des actions

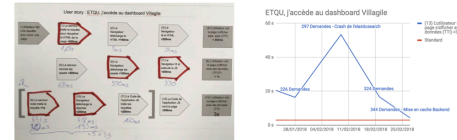
- Supprimer le fichier `reset.css` Gain : 0.1s
- Décaler dans le temps les appels à Piwik et à HotJar. Gain : 0.5s
- Suppression de la double boucle réseau. Gain : 0.5s
- Arrêter d'utiliser des fonctions anonymes. Gain : ?
- Utiliser `reset` pour memoiser les résultats des sélecteurs. Gain : ?

#### Next Steps

- Supprimer les appels `OPTION` inutiles (cf. standard)
- Le backend a l'air d'avoir des temps de réponse extrêmement variables (~100ms-2s) pour une même ressource. Vérifier que ça ne vient pas du réseau de Theodo. Si c'est pas le cas, demander conseil à l'infogérant.
- Enquêter sur les renders inutiles sur la page des posts

**Suez Villagile** — 3.5s ✗ — 26/10

KévinJ



#### Résumé des actions

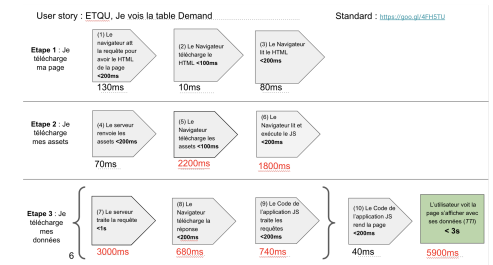
- [31/12] Ne plus charger les analyses retro à l'ouverture de l'application Gain: 14s
- [09/01] Ne plus charger les demandeurs & les interventions à l'ouverture de l'application Gain : 14s
- [23/01] Avoir une prod avec de vrai data
- [26/01] Optimiser la fonction de recherche Gain : Recherche en moins de 1s
- [30/01] Réparer ElasticSearch Gain : 20s
- [19/02] Retirer du chargement de la page la récupération des données géographiques (SIO) Gain : 800ms
- [28/02] Mettre en place un cache côté backend pour conserver les appels au serveur de données Gain attendu : 8s, Gain réel : 12s
- [16/03] Priorisation de la mise en place de Gzip Gain attendu : 1s KO
- [23/03] Réduction de la taille du bundle en supprimant les libs pas utilisées : Gain attendu : 0.5s KO
- Délayer le chargement des configs, donc le HTML part plus vite. Gain : 1s
- Supprimer un call API sur les contours des villes (mis à jour une fois par an)
- Preloader les polices : [https://developer.mozilla.org/fr/docs/Web/HTML/Pr%C3%A9charger\\_du\\_contenu](https://developer.mozilla.org/fr/docs/Web/HTML/Pr%C3%A9charger_du_contenu) KO

#### Next Steps

- Mettre en place React Virtualized pour alléger la construction du DOM
- Alléger l'import de moment : <https://github.com/moment/moment/issues/1170>
- Alléger l'import de lodash : utiliser 'import { map } from lodash/map' par exemple
- Supprimer / différer le chargement de JQuery
- Minifier le javascript hors-Webpack
- Supprimer le call API pour avoir le contour des communes
- Améliorer la cache policy

**OCF Casa (page modèle)** — 5.9s ⇒ 2.4s ✓ — 22/10

FlorianG



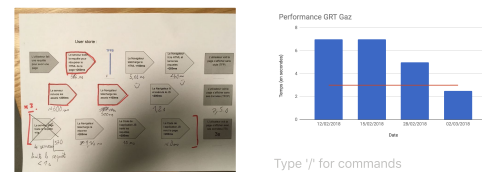
#### Résumé des actions

- Mise en place de Gzip. Gain : 1s
- Identification des librairies inutilisées et priorisation avec *Bundlephobia*
- Suppression de la librairie des drapeaux pour des emojis KO c'était moche
  - Chargement dynamique des SVGs : gain de 700Ko
- Identification du flux de performance backend :
  - Aujourd'hui on est plus à 6s sur le serveur traite la requête, ~1s pour le download (10Mo de données !!!)
- Quick fix : Mise en place de Gzip

**GRT Gaz** — 2.5s ✓

AlexAdrienA

Condition réseau : Fast 3G



#### Résumé des actions

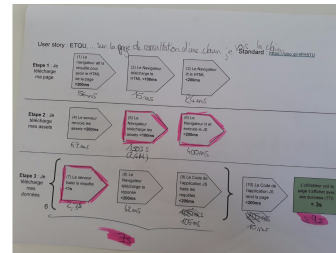
- [12/02] Changer les headers sur nginx pour activer le cache coté client : KO car le reverse proxy GRT Gaz rajoute les headers "no-store, no-cache"
- [15/02] Mettre en place un service worker pour cacher les assets front + html : KO car la librairie utiliser pour générer des services workers avec angular "@angular/service-worker" ne transmet pas les headers d'authentification nécessaires pour passer le reverse proxy
- [28/02] Enquête pour supprimer les 2 boucles réseaux inutiles (3 fois le même call à Prismic) Gain attendu : 2s ⇒ KO car l'équipe a viré Prismic du projet, du coup plus de boucles réseau
- [02/03] Retrait du header `Cache-Control: Private` avec un des archis GRT Gaz. Gain : 2.5s
- Objectif de TTI de 2s (OK) sur desktop et 10s sur mobile OK

**Sphere** — 9s ⇒ 5s ✗ — **Projet terminé**

AlbéricT

"Les outils actuels de nos potentiels clients sont lents, et c'est une opportunité pour nous : si notre produit est rapide on pourra convertir plus de clients"

Nicolas Gavenard, PO Sphere



#### Résumé des actions

- Paralléliser les requêtes faites au backend Gain : 3s
- Simplification des requêtes GraphQL
- Mise en place de Tideways (monitoring de performances PHP) sur Heroku
- Activer Gzip d'ici le 05/09 ⇒ Gain : 1s

#### Next Steps

- Cleaner l'import de Moment et de Lodash KO
- Mettre en place du Tree Shaking KO
- Investigation sur les performances de GraphQL KO

# Identifying the Value Stream

👁️ Forces to [look at the whole process](#), not only frontend/backend

🕒 Setting standard times meant that [teams asked for help much sooner](#), and it was easier and faster to help them

💰 Great format to [communicate with stakeholders](#) and get buy-in, as it makes “tech stuff” much more tangible

# Jidoka: catch problems at the earliest

🛑 In Toyota factories, whenever a **defect** is spotted on the assembly line, the operator **stops the whole line** to try and fix the defect instead of passing it down the line.

🔧 Then, the process is refined to make sure that it **prevents that defect from happening again**.

# Jidoka: catch problems at the earliest

**In the IDE:** We use the Import Cost VSCode extension to detect heavy libraries when we add them to our code.

```
import React from 'react'; 8K (gzipped: 3.3K)
import { connect } from 'redux'; 2.3K (gzipped: 895)
import { Link } from 'react-router'; 48.4K (gzipped: 14K)
import Button from '@material-ui/core/Button'; 105.6K (gzipped: 27.6K)
```



# Jidoka: catch problems at the earliest

**In the command line:** We spot avoidable library duplication in bundles using yarn.lock information

```
#!/bin/bash

duplicates=$(node ./node_modules/yarn-tools/index.js list-duplicates yarn.lock);

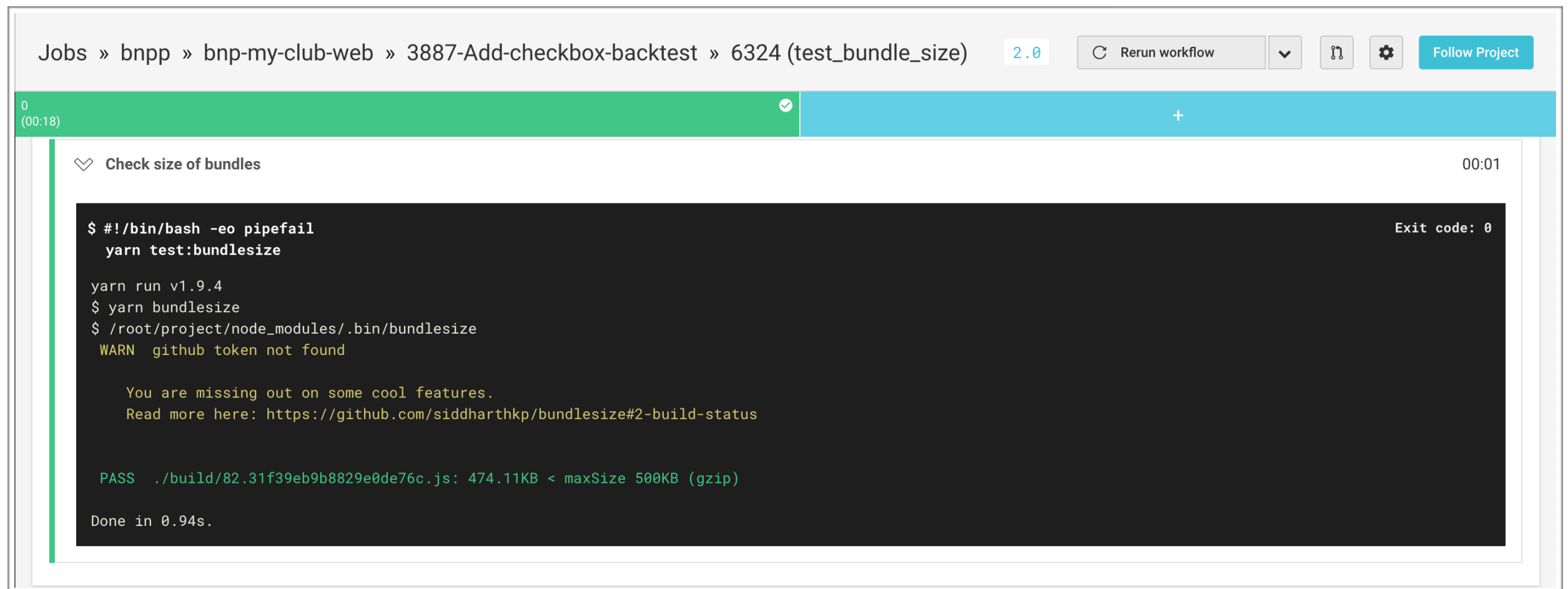
duplicates_number=$(echo "$duplicates" | grep "Package" | wc -l);

if [ "$duplicates_number" -gt "0" ]; then
    echo "Found $duplicates_number unnecessary duplicate dependencies, please run 'yarn duplicates:fix'";
    echo "$duplicates";
    exit 1;
else
    echo "No unnecessary duplicate dependencies found, good to go";
fi
```

*Runs automatically on **yarn** or **yarn add***

# Jidoka: catch problems at the earliest

**In the CI:** We check on each PR if the bundle size of our web site is under a certain threshold (here, 500Kb).



The screenshot shows a GitHub Actions workflow run for the repository 'bnpp-my-club-web'. The workflow is named '3887-Add-checkbox-backtest' and the specific job is '6324 (test\_bundle\_size)'. The job status is '2.0' and it is marked as 'Rerun workflow'. The job is part of a sequence of jobs: 'bnpp' » 'bnp-my-club-web' » '3887-Add-checkbox-backtest' » '6324 (test\_bundle\_size)'. The job is currently running, as indicated by the green bar and the checkmark icon. The job title is 'Check size of bundles' and it has a duration of '00:01'. The job output shows the following commands and results:

```
$ #!/bin/bash -eo pipefail
yarn test:bundleSize

yarn run v1.9.4
$ yarn bundleSize
$ /root/project/node_modules/.bin/bundleSize
WARN  github token not found

You are missing out on some cool features.
Read more here: https://github.com/siddharthkp/bundleSize#2-build-status

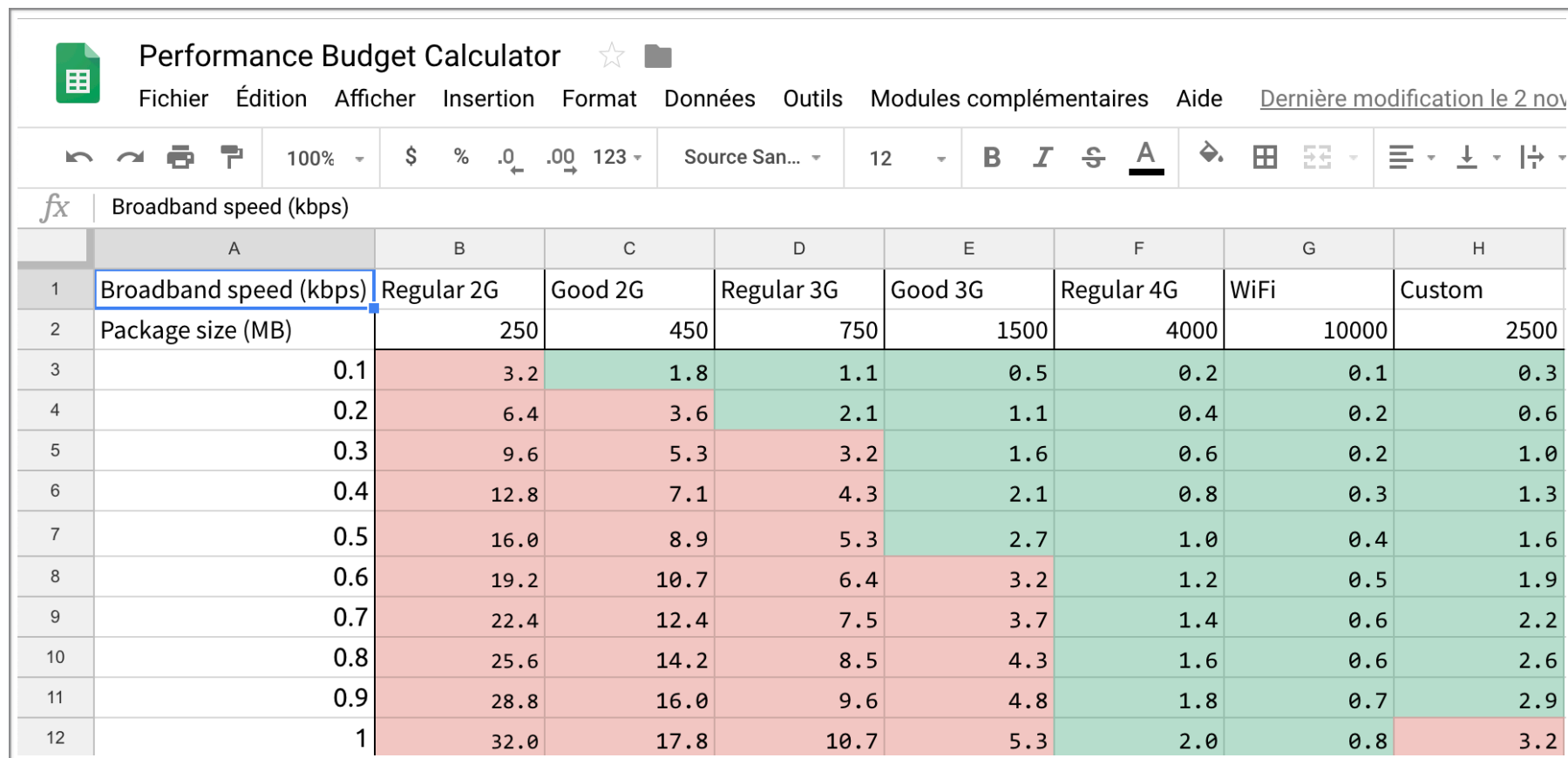
PASS  ./build/82.31f39eb9b8829e0de76c.js: 474.11KB < maxSize 500KB (gzip)

Done in 0.94s.
```

The output also shows the exit code: 'Exit code: 0'.

# Jidoka: catch problems at the earliest

**Finding the right threshold:** Find your users' average network speed, verify them & use this handmade, organic & gluten-free [Performance Budget Calculator](#)



The screenshot shows a Google Sheets interface titled "Performance Budget Calculator". The menu bar includes "Fichier", "Édition", "Afficher", "Insertion", "Format", "Données", "Outils", "Modules complémentaires", "Aide", and a link for "Dernière modification le 2 nov". The toolbar shows various editing and formatting icons. The spreadsheet has columns A through H. Column A is labeled "Broadband speed (kbps)". Column B is "Regular 2G", C is "Good 2G", D is "Regular 3G", E is "Good 3G", F is "Regular 4G", G is "WiFi", and H is "Custom". The data rows show values for different bandwidth thresholds (0.1 to 1.0 kbps) and their corresponding performance metrics for each network type. The values are color-coded: red for "Regular" and green for "Good".

	A	B	C	D	E	F	G	H
1	Broadband speed (kbps)	Regular 2G	Good 2G	Regular 3G	Good 3G	Regular 4G	WiFi	Custom
2	Package size (MB)	250	450	750	1500	4000	10000	2500
3	0.1	3.2	1.8	1.1	0.5	0.2	0.1	0.3
4	0.2	6.4	3.6	2.1	1.1	0.4	0.2	0.6
5	0.3	9.6	5.3	3.2	1.6	0.6	0.2	1.0
6	0.4	12.8	7.1	4.3	2.1	0.8	0.3	1.3
7	0.5	16.0	8.9	5.3	2.7	1.0	0.4	1.6
8	0.6	19.2	10.7	6.4	3.2	1.2	0.5	1.9
9	0.7	22.4	12.4	7.5	3.7	1.4	0.6	2.2
10	0.8	25.6	14.2	8.5	4.3	1.6	0.6	2.6
11	0.9	28.8	16.0	9.6	4.8	1.8	0.7	2.9
12	1	32.0	17.8	10.7	5.3	2.0	0.8	3.2

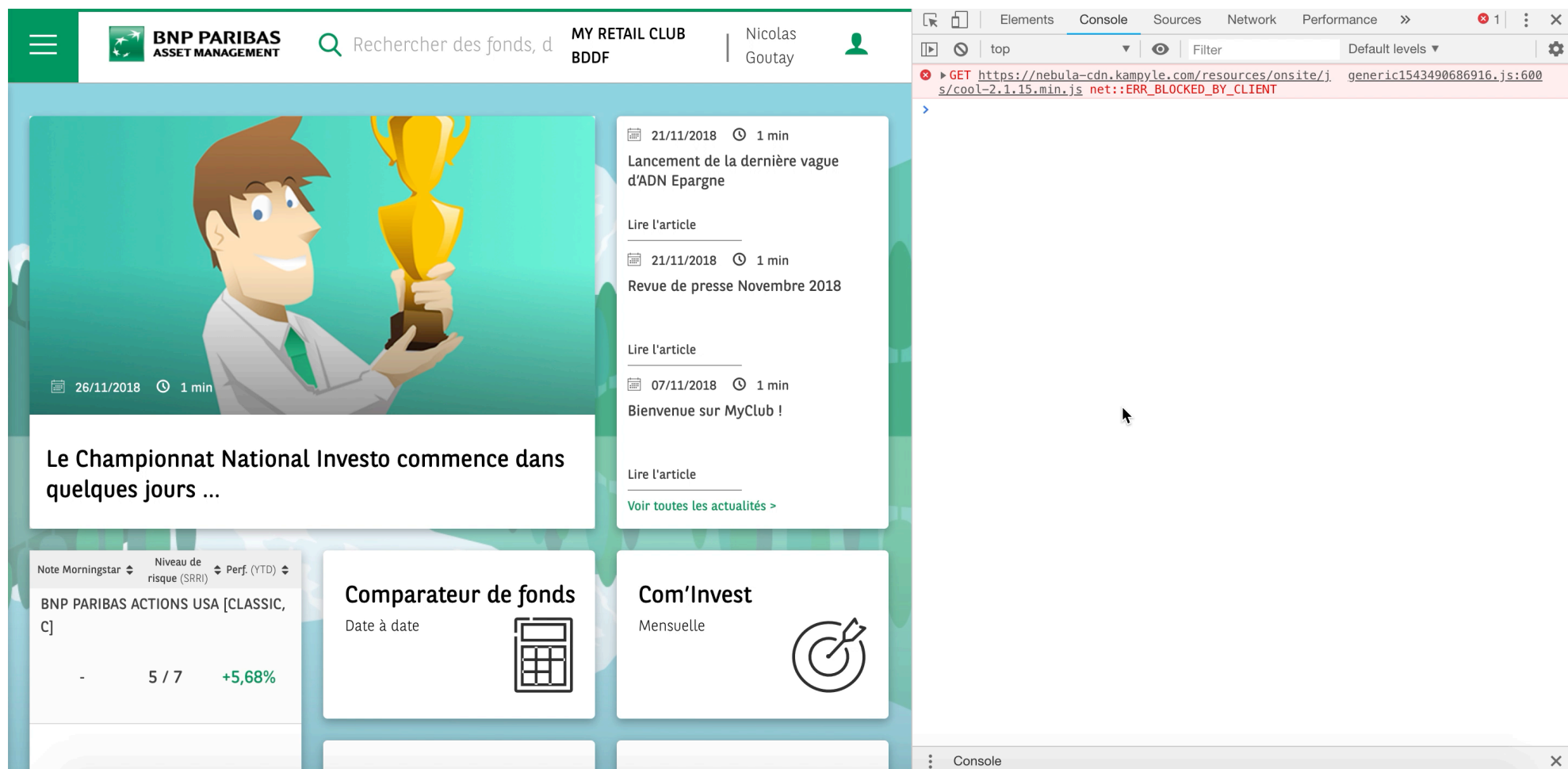


## The Tools of the Trade

# Little-known Chrome Dev Tools for Web Perf

🎨 **Coverage:** Real-time CSS & JS code coverage.

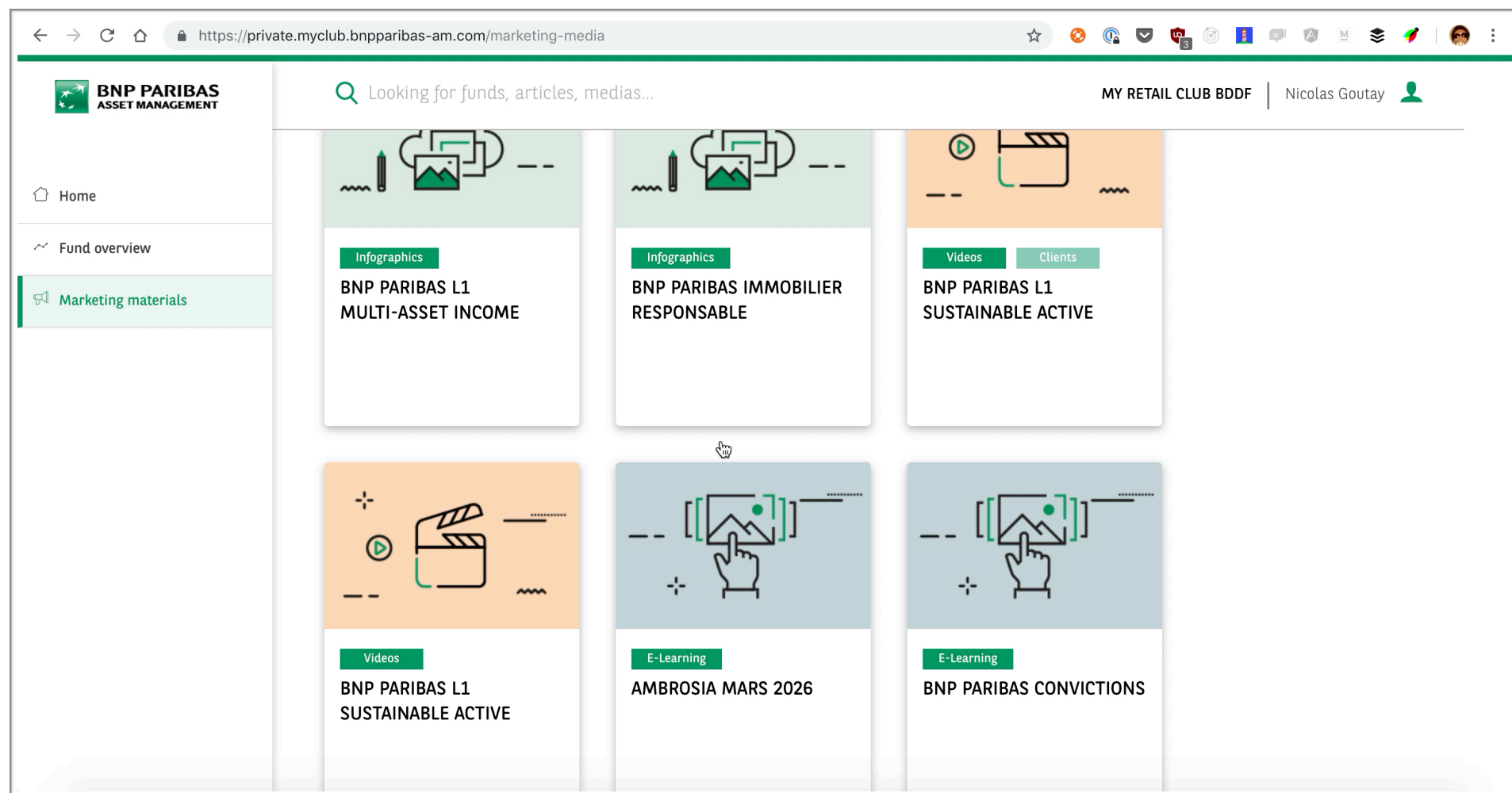
💡 *Helpful for:* detecting unused libraries or dead code.



# Little-known Chrome Dev Tools for Web Perf

🎨 **Paint Flashing:** Highlight in green the parts of the UI that has been painted by Chrome.

💡 *Helpful for:* detecting useless renders slowing down the UI.



# Why Did You Render?

**? why-did-you-render:** Displays a warning in the Chrome console when a React makes unnecessary updates


💡 *Helpful for:* detecting useless renders slowing down the UI.

```
⚠️ ClassDemo.props: Changes are in functions only. Possibly avoidable re-render?  
  Functions before: ▶Object {fn: function something()}  
  Functions after: ▶Object {fn: function something()}  
⚠️ ClassDemo.state: Value is the same (equal by reference). Avoidable re-render!  
  Value: null
```


# Keeping it small


 **bundlesize**: CI-friendly bundle size checker.

💡 *Helpful for*: detecting if your JS goes over a certain threshold



✓ All checks have passed  
2 successful checks [Hide all checks](#)

✓  **bundlesize** — ./dist.js: 3.6Kb < threshold 4Kb (0.2Kb larger than master, careful!)

✓  **continuous-integration/travis-ci/push** — The Travis CI build passed [Details](#)

✓ This branch has no conflicts with the base branch  
Merging can be performed automatically.

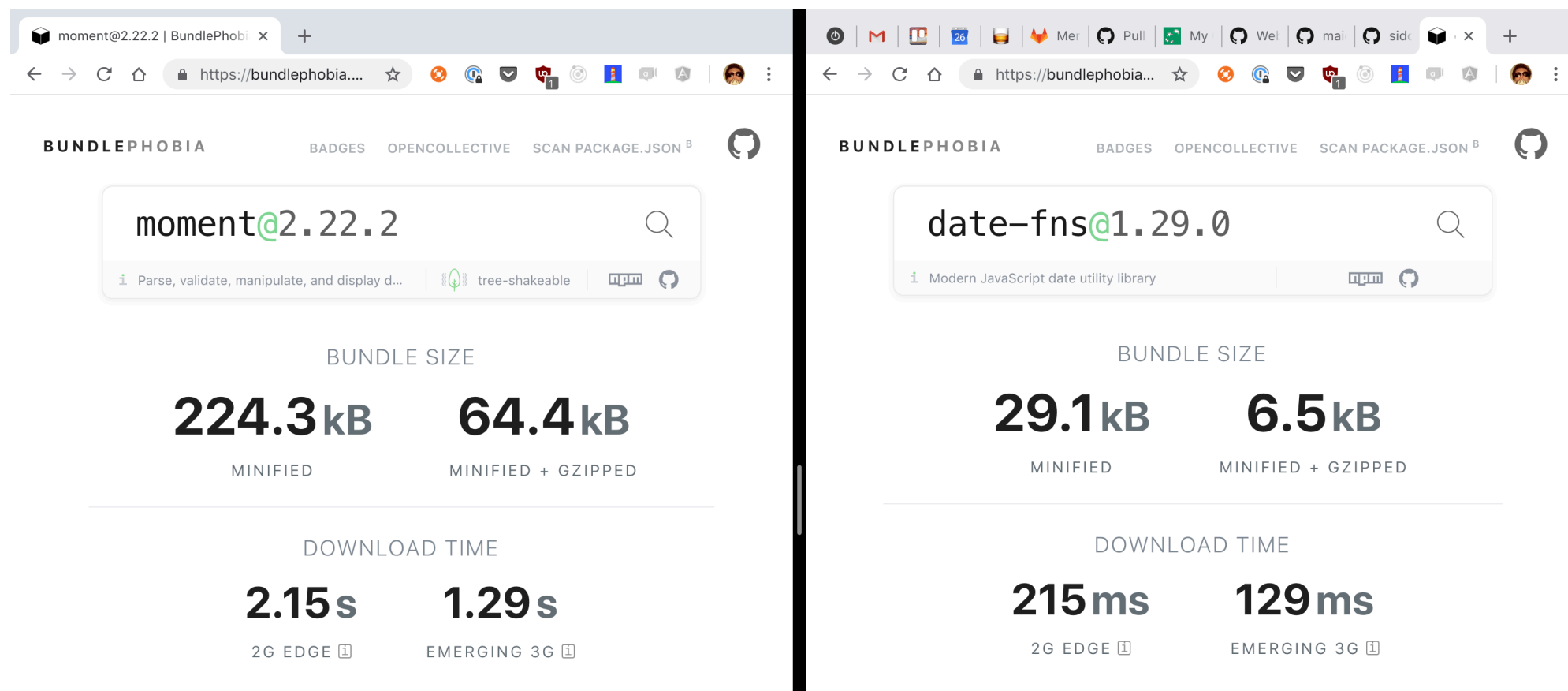
Merge pull request ▼



# Keeping it small


😱 **Bundlephobia**: Find the cost of adding a NPM package to your bundle.

💡 *Helpful for*: Deciding between two libraries with similar functionalities.



# Keeping an eye on performance


 **Falco:** Theodo's Open Source WebPageTest runner

 **FALCO**

**ChooseMyCompany**

[Launch audits manually](#)

Audits environment

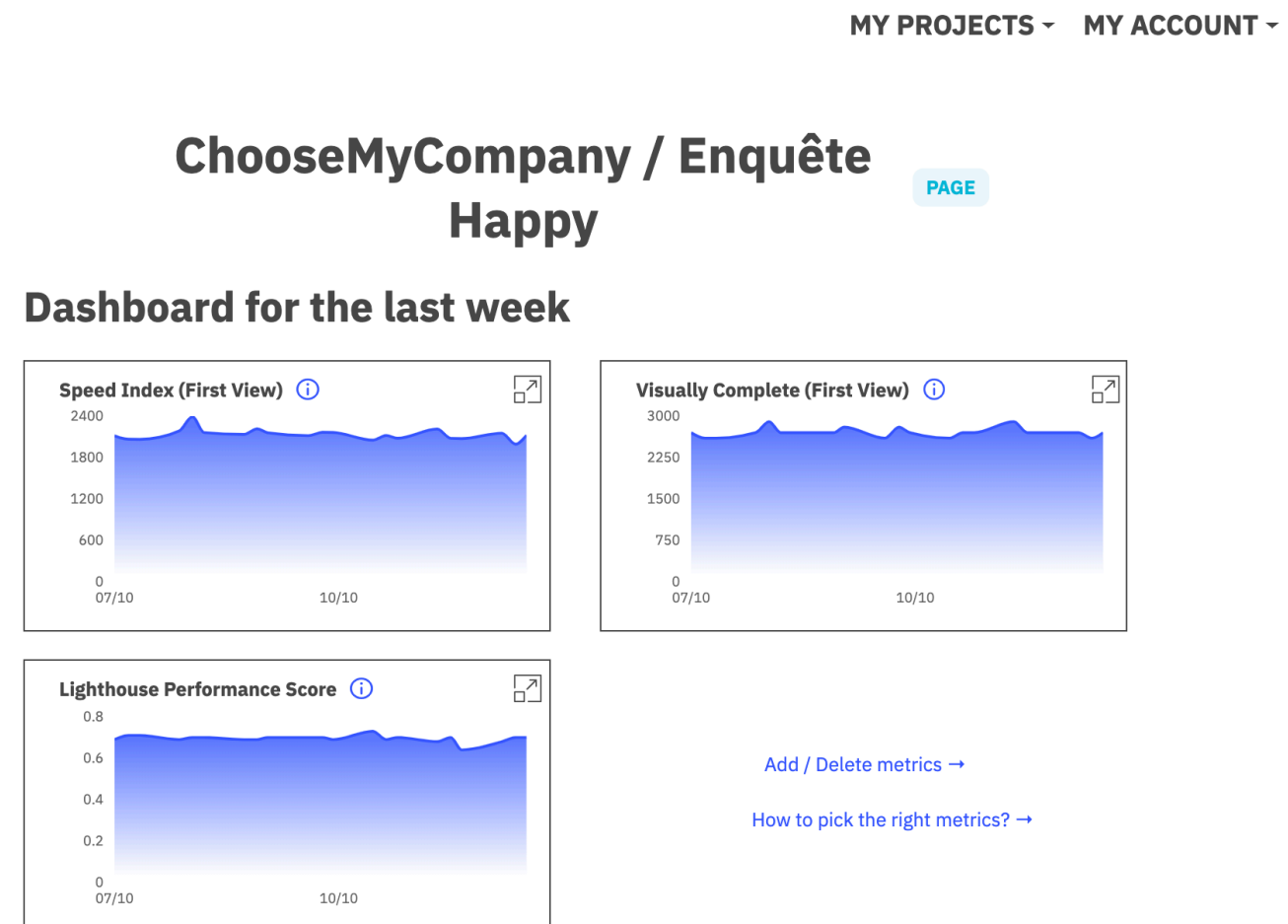
Chrome | Cable (Dulles) | 

**Audits**

**Enquête Happy** [PAGE](#) >

**Home publique** [PAGE](#) >

**Espace Employeur** [SCRIPT](#) >



# Knowledge

**Knowledge is power, Sharing is better**

# Share your discoveries!



**Weekly Perf:** Every week, we have a 1h informal meeting about performance, where we share new libraries, articles, and tackle specific problems on a project.




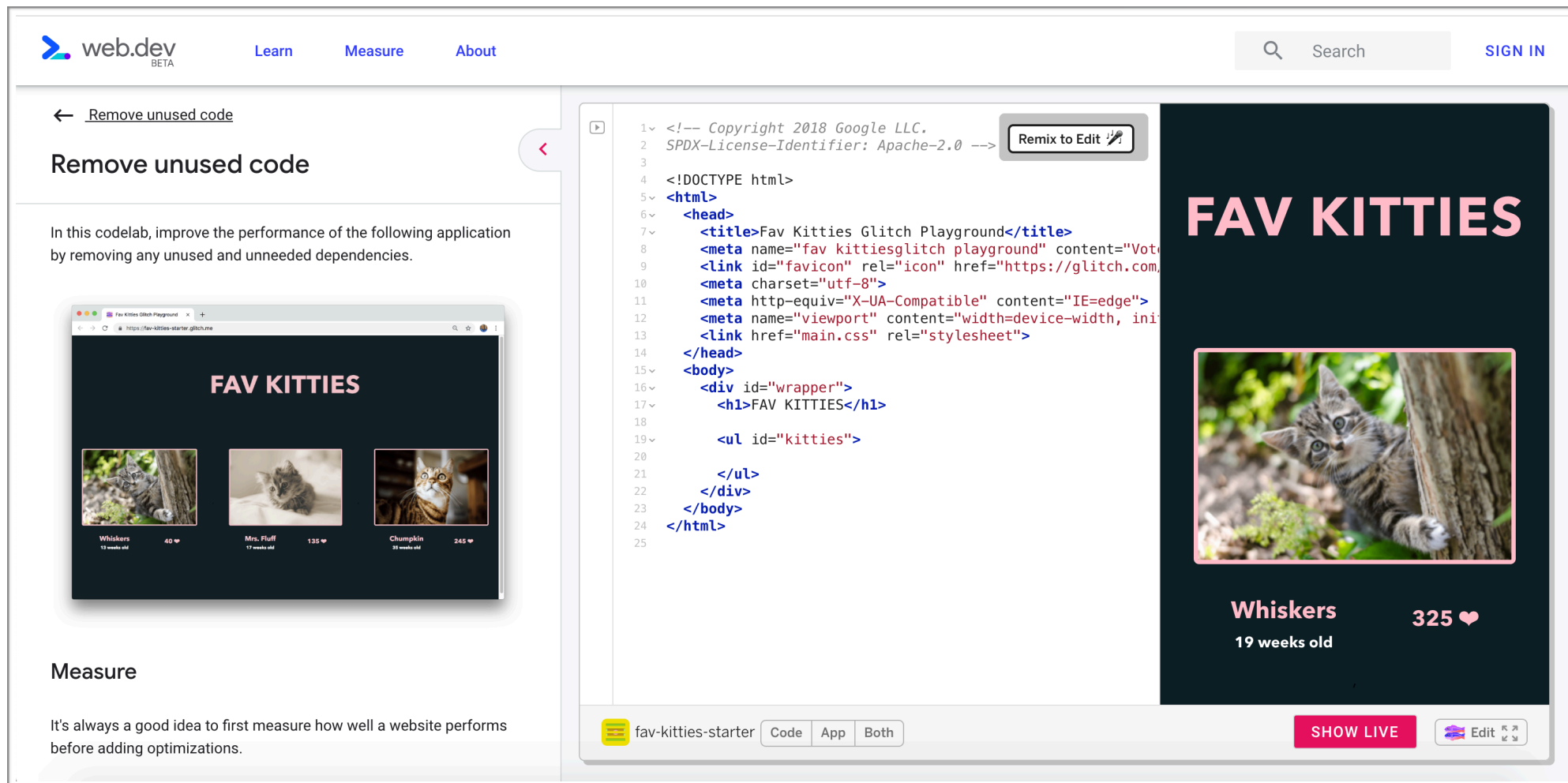
**Dedicated channel:** On your company's social network (Slack, Workplace...), set up a channel dedicated to all things Web Perf.



**Dedicated training :** I help run a 4-week WebPerformance cursus, 1 hour every Monday, so that all developers can learn the basics and apply them to their projects

# Best resources to get started

 **web.dev**: Online Lighthouse audit; lots of [tutorials](#) to implement performance best practices with [interactive exercises](#)



The screenshot displays the 'Remove unused code' exercise on the web.dev platform. The interface is divided into three main sections:

- Remove unused code:** This section contains instructions: "In this codelab, improve the performance of the following application by removing any unused and unneeded dependencies." Below the text is a preview of the 'Fav Kitties' application, which shows three kittens: Whiskers (12 weeks old, 40 hearts), Mrs. Fluff (17 weeks old, 135 hearts), and Chumpkin (30 weeks old, 245 hearts).
- Code Editor:** This section shows the HTML code for the application. The code includes a title 'Fav Kitties Glitch Playground', a meta tag for the favicon, a link to the main CSS file, and a list of kittens. The code is as follows:

```
1 <!-- Copyright 2018 Google LLC.
2 SPDX-License-Identifier: Apache-2.0 -->
3
4 <!DOCTYPE html>
5 <html>
6 <head>
7 <title>Fav Kitties Glitch Playground</title>
8 <meta name="fav kittiesglitch playground" content="Vote
9 <link id="favicon" rel="icon" href="https://glitch.com
10 <meta charset="utf-8">
11 <meta http-equiv="X-UA-Compatible" content="IE=edge">
12 <meta name="viewport" content="width=device-width, ini
13 <link href="main.css" rel="stylesheet">
14 </head>
15 <body>
16 <div id="wrapper">
17 <h1>FAV KITTIES</h1>
18
19 <ul id="kitties">
20
21 </ul>
22 </div>
23 </body>
24 </html>
25
```
- Measure:** This section contains the text: "It's always a good idea to first measure how well a website performs before adding optimizations."

The interface also includes a 'Search' bar, a 'SIGN IN' button, and a 'Remix to Edit' button. At the bottom, there are buttons for 'Code', 'App', and 'Both', and a 'SHOW LIVE' button.

# Best resources to get started

📖 **HPBN**: High Performance Browser Networking — a [free e-book](#) teaching deep understanding of [how browsers & Web protocols work](#), so that [you can get the most of them](#)

★★★★½ (463): [GoodReads](#) ↗ [Amazon](#) ↗ [O'Reilly](#) ↗

“ *This book is required reading for anyone who cares about web performance; it's already established as the go-to reference on the topic.*

—Mark Nottingham (IETF HTTPBis Chair)



**Thanks!**

# Slides

Available at: <https://noti.st/phacks>

## Sources

My article in Planet Lean: <https://planet-lean.com/doubling-performant-apps-using-kaizen/>

Import Cost VSCode extension: <https://github.com/wix/import-cost>

Yarn tools to list duplicates: <https://gist.github.com/phacks/6878465820605e6c2946f034f70f662c>

Performance Budget Calculator: <https://docs.google.com/spreadsheets/d/1X7RTp0cQbuSTA1ND07K-Ln5V4i3iwmWA00Cndgs4ulw/edit?usp=sharing>

Bundlesize: <https://github.com/siddharthkp/bundlesize>

Bundlephobia: <https://bundlephobia.com>

Why Did You Render?: <https://github.com/welldone-software/why-did-you-render>

Web.dev: <https://web.dev/learn>

HPBN: <https://hpbn.co/>

Falco: <https://getfal.co/>