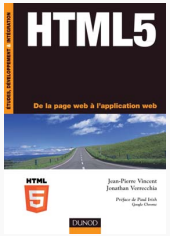
Performance Web
L’intégrateur ce héro

Jean-pierre Vincent – Expert Webperf indépendant

Hello !

Jean-Pierre Vincent

BrainCracking



- Consultant Webperf
- Formations Webperf
- Architecte JS, Développeur Web

- Co-organisateur de we ♥ speed

  We Love Speed

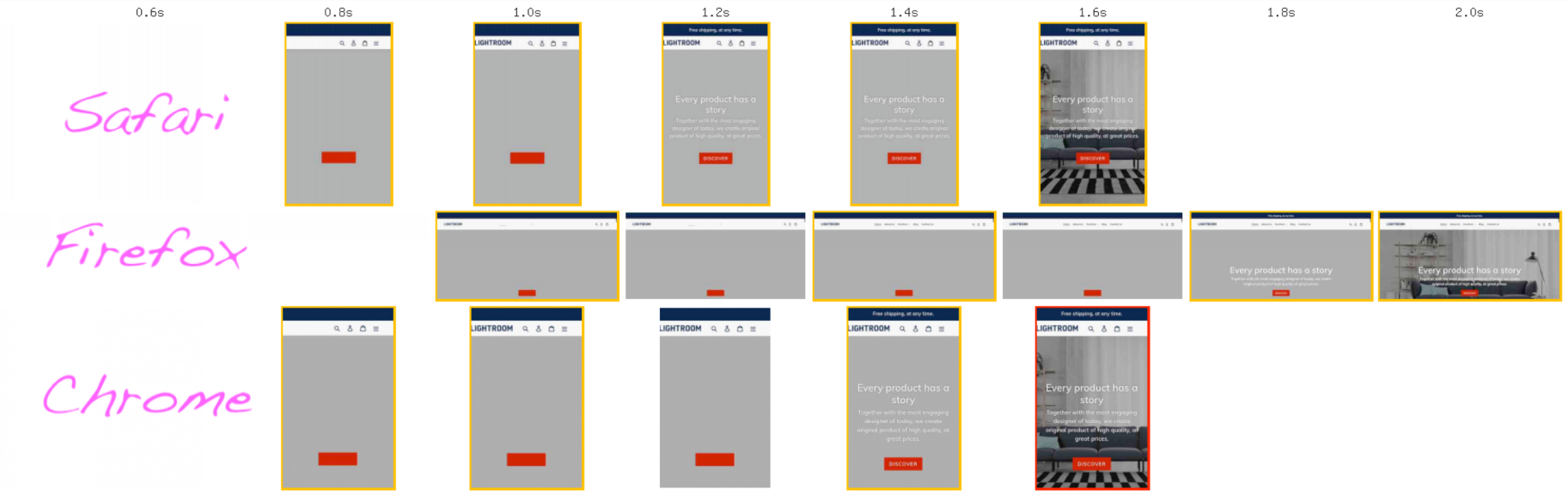
@theystolemynick

Afficher des trucs utiles (vite)

PROBLÈME DE BASE

Passer les scripts en asynchrone 💓

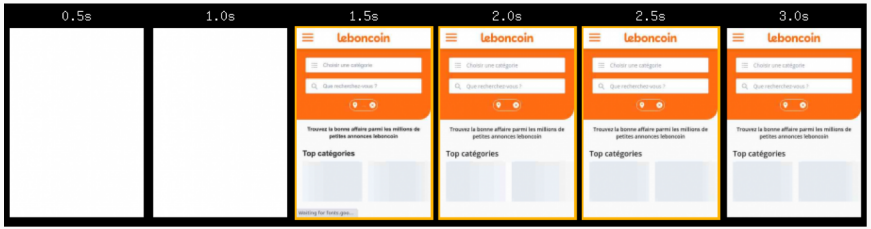
⚡Affichage rapide !

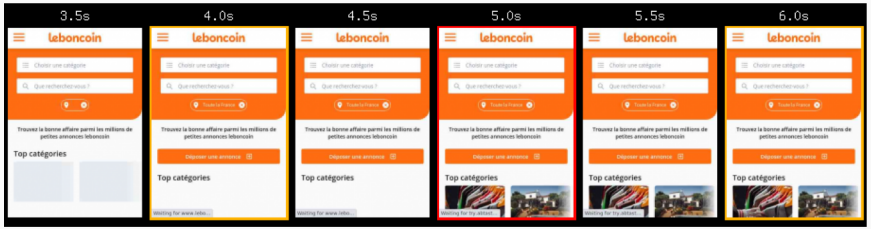


Et le début des problèmes 😅

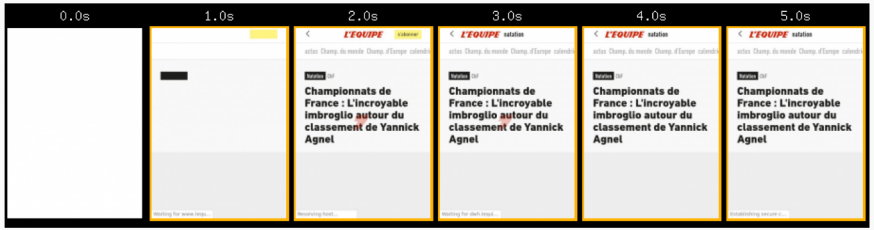
L’effet page non-réhydratée 

Certes en React …





Mais VueJS aussi 



Mais on a fait plein de progrès !

Pleins de solutions ! 🔧

OK, mais que voit l'utilisateur entre le retour serveur et l'arrivée de JS ?

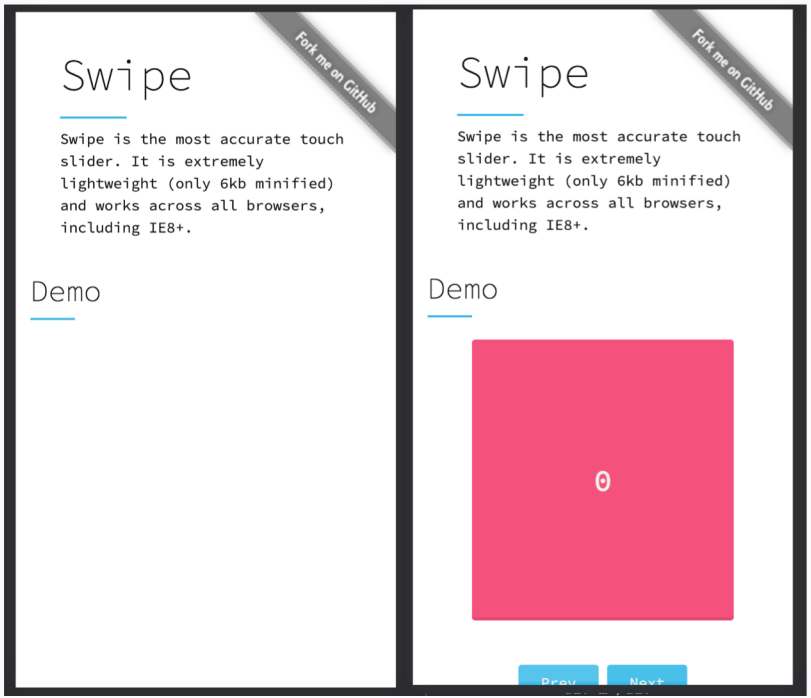
- Pas de texte
- Pas d'image
- Pas d'interactivité
- Des blocs qui bougent

Objectifs de l’intégrateur : progressive enhancement

- Le contenu d’abord ! 📚
- JS termine la page 🍒🎂
- UX agréable, l’attente est prévue depuis le design !
- Bonus : situations sans JS

Exemple : les plugins de slideshow

Le site de démo 🤦



Le test de la honte

L'intégrateur 🤦‍♀️

- Peut corriger le CSS
  - Ex: supprimer visibility:hidden par défaut au moins sur la 1ère slide
- Peut envisager des alternatives 0 JS
  - scroll-snap-type
  - Au moins temporairement

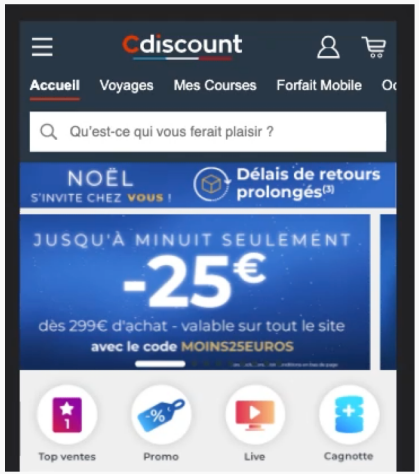
Alternatives 0 JS ? Vraiment ?

Soyons un peu réalistes

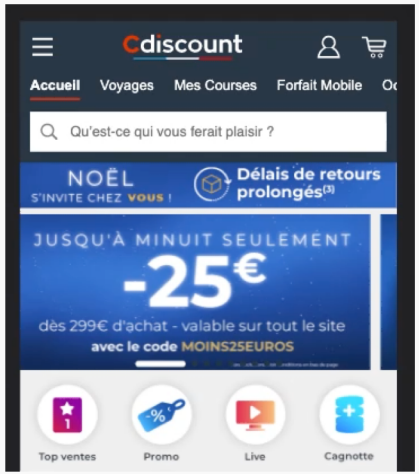
- Pour un carousel, il manquerait :
  - L'interface (bouton next, indicateur de position ...)
  - Le lazy-loading d'images
  - La rotation automatique
- L'alternative **nous sert à patienter**.

Peaufinez les détails

Avec JS



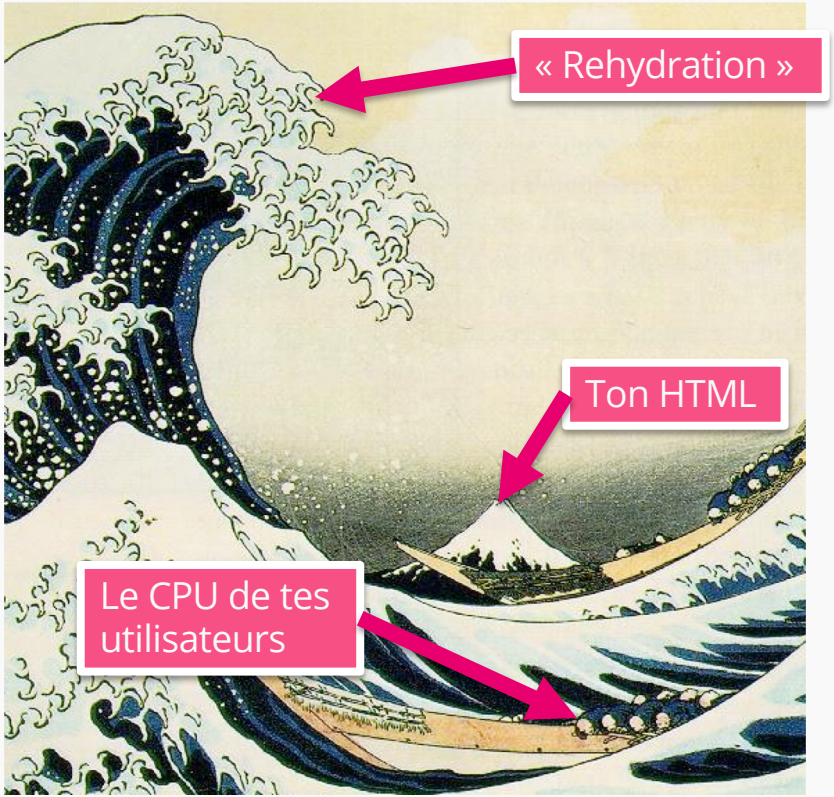
Sans JS



Quand l’intégration sauve le dev JS

ALLER PLUS LOIN

Réduire la charge du début de page



Hokusai : La Grande Vague de Kanagawa

- Effet **Big Bang** 💥 : chargement et exécution des JS à la création de la page
- Effet **Tsunami** 🌊 : la réhydradation des frameworks modernes
- Rappel : **l'exécution** se fait même si JS est en cache.

Réduire l'effet Big Bang, sans optimiser JS ? 🙃

- Chercher les opportunités dans l'interface
- Vos composants sont ils **tous** utiles là maintenant tout de suite ? (non)

Ex: Datepicker

- Pourquoi charger et exécuter tant que l'utilisateur n'a pas cliqué dedans ?

- Demo time !
  - Utilisation de import()

À retenir

- Du point de vue performance :
  - JS est un problème (même s'il apporte ses propres solutions)
  - HTML et CSS sont les premières solutions à évaluer
- L'intégration permet
  - À minima un affichage temporaire satisfaisant
  - Parfois de remplacer totalement JS

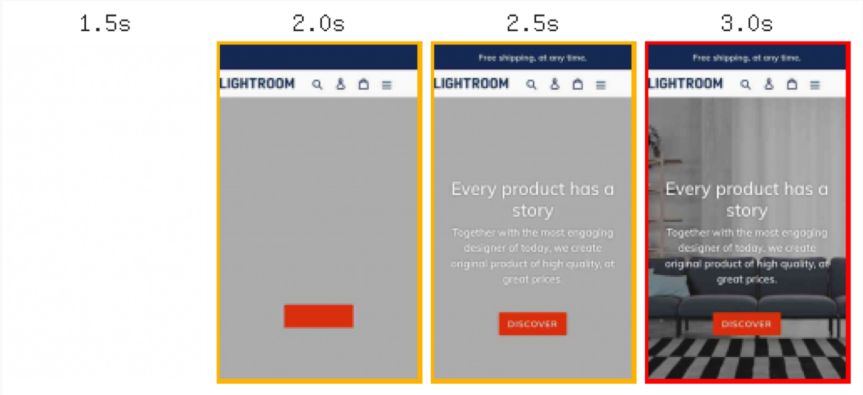
Non, sérieux c’est compliqué

AFFICHER UNE IMAGE

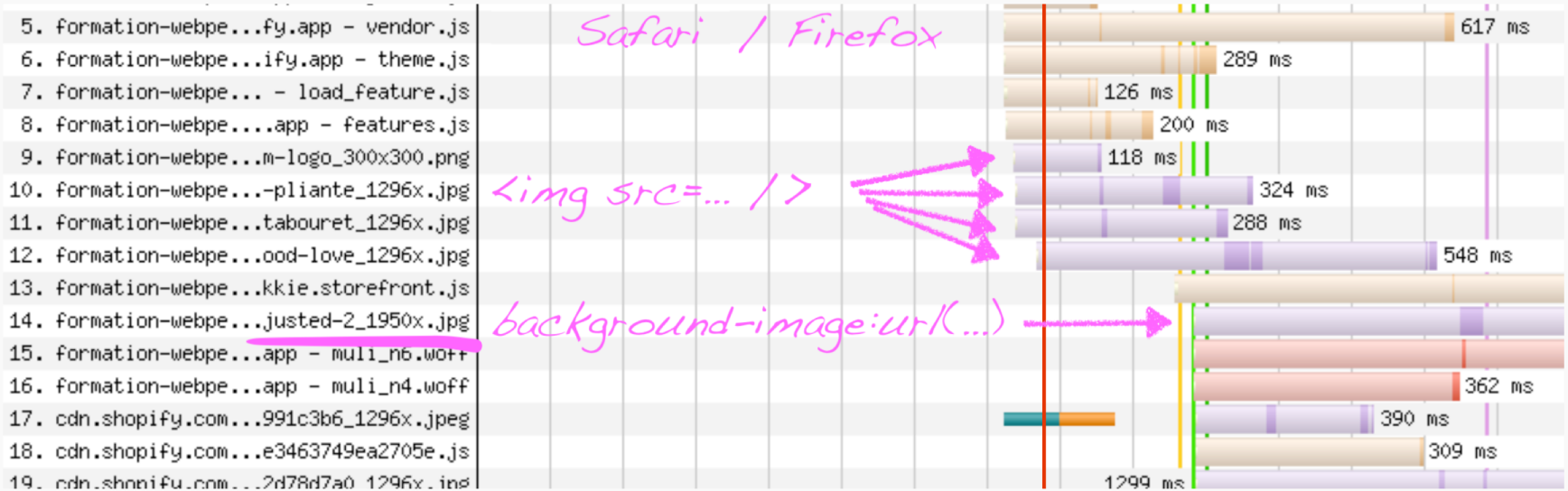
L’image principale

- But principal : affichage rapide de la proposition commerciale

- Secondaire : la métrique Largest Contentful Paint



L’image principale en background-image ? 🙄



test iPhone ou Firefox : les images de background sont dépriorisées

Fix pour Firefox / Safari

```
<img src=Hero.jpg
        style=display:none; />

<img src=Hero.jpg
        style=object-fit:cover; />

...
```

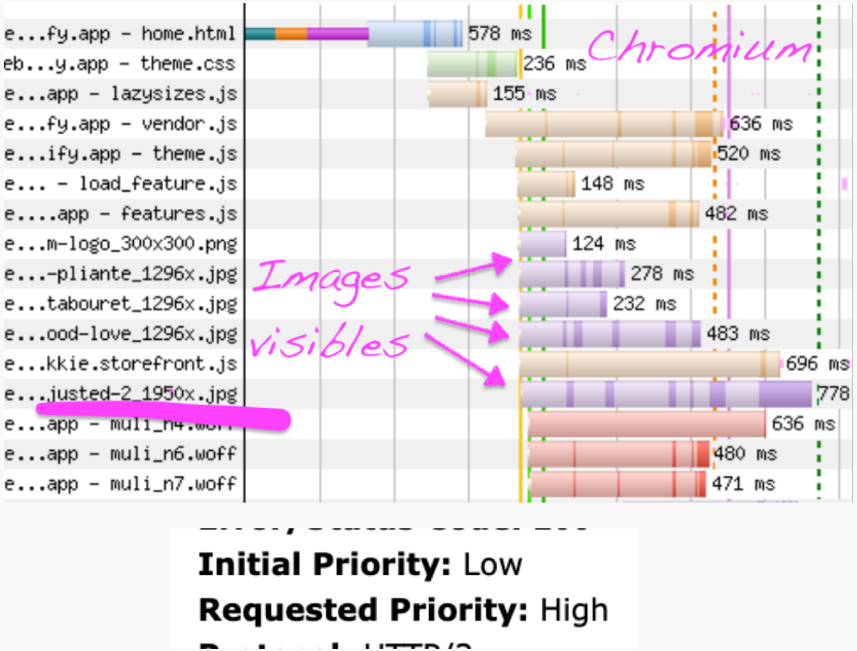
Le chargement démarre dès que la balise est trouvée !

⬅ Petit hack rapide

⬅ Meilleure option

- (hors IE 11 🤦)

Comportement Chromium avec les images

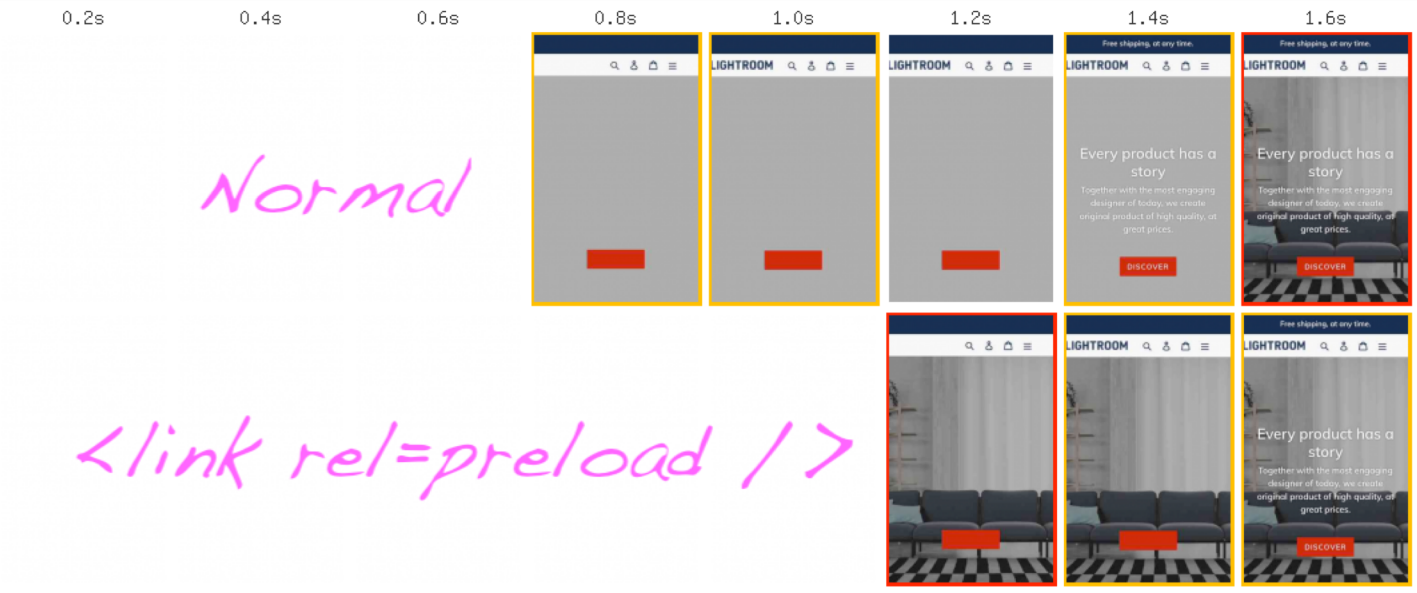


- L'origine (HTML, CSS) ne compte pas
- Priorisation basée sur la visibilité réelle de l'image
- **Mais** ... TOUTES les images attendent CSS

Accélérer une image visible : quelles options ?

- 👉🚄 Preload
- 🧚 Réduire le poids (challenge : sans compression)
- 🥇 Déprioriser tout le reste

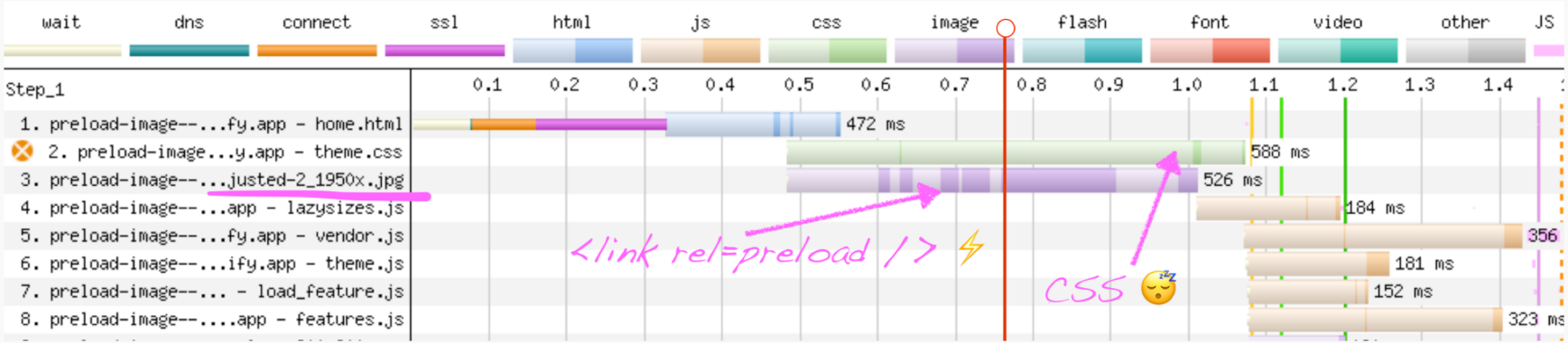
Preload de contenu important



<**link rel=preload** href=Hero.jpg as=image />

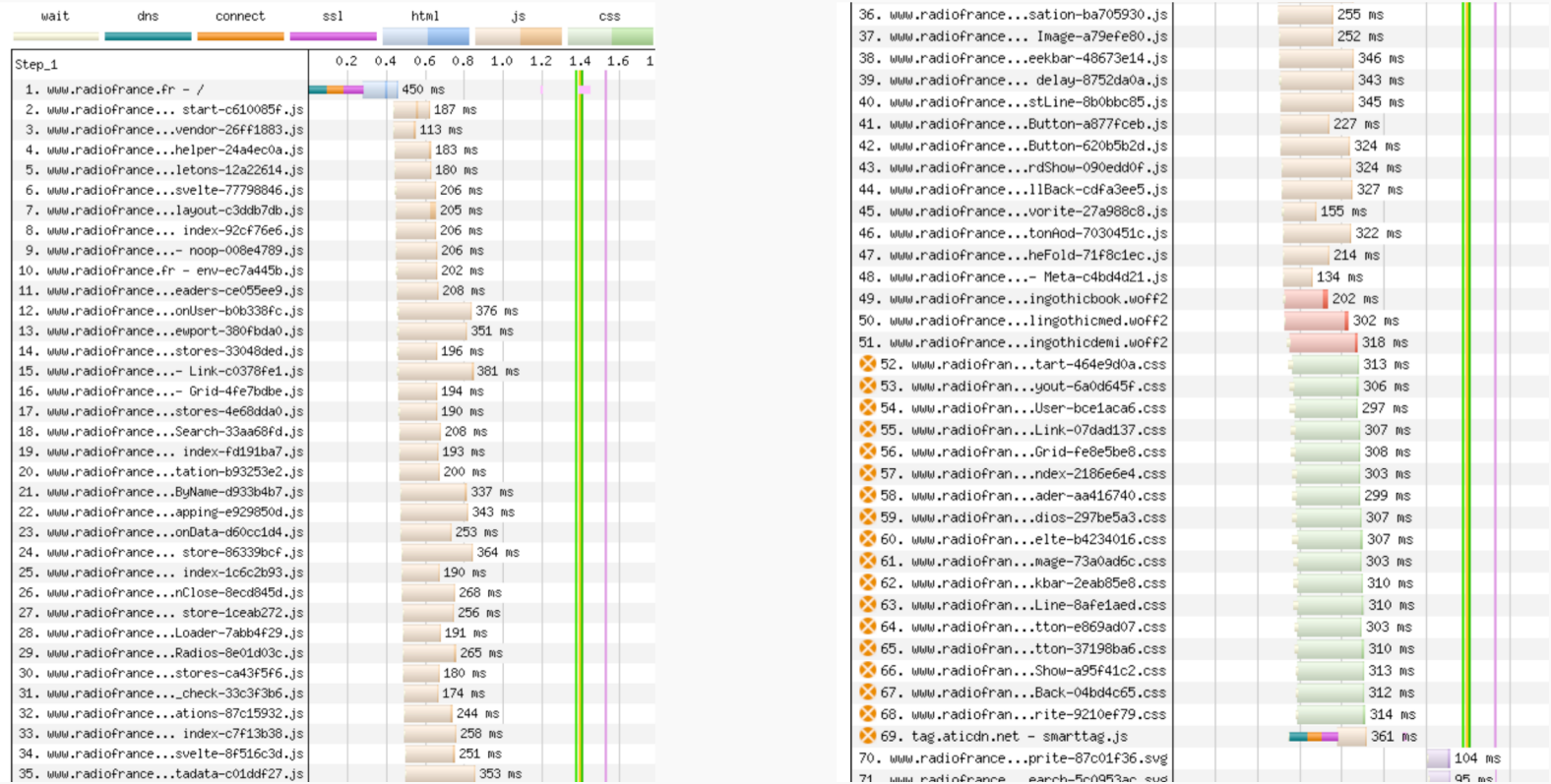
Similaire sur Firefox, Chromium, et Safari

Preload est à utiliser avec modération



Ici l'image est trop lourde pour qu'il n'y ait pas de conséquence négative.

Preload JavaScript par défaut ? 😐



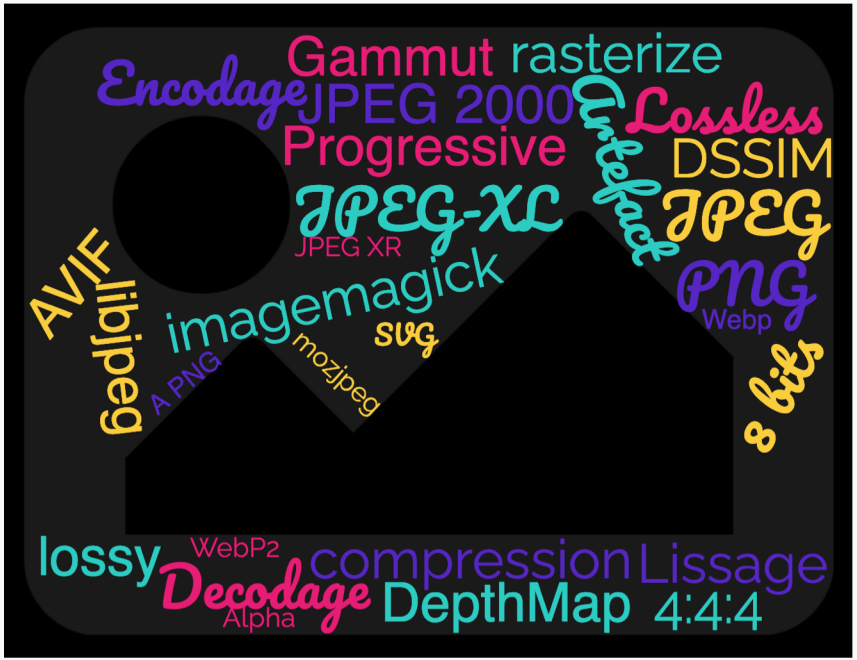
[Test stack Svelte + Vite]

@theystolemynick

Accélérer une image visibles : quelles options ?

- ✅🚄 Preload
- 👉🧚 Réduire le poids (challenge : sans compression)
- 🥇 Déprioriser tout le reste

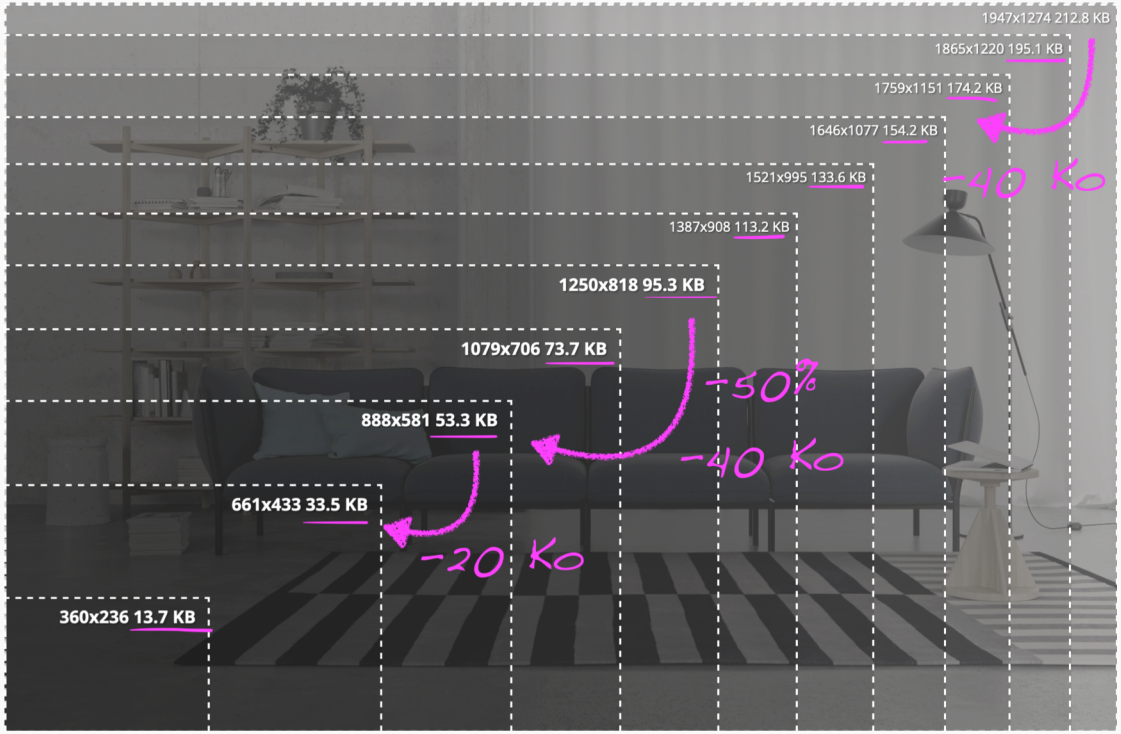
Il suffit de compresser l'image ! 🤥



- Nouveaux formats ou super algos : gain moyen de **30% du poids** par rapport à une image compressée à la va-vite.

- Mais il y a plus efficace !

Technique brevetée JMIPP™ 🤠



Je Mets l'Image Plus Petite.

Images responsive

```
<img class="img" alt="Chaise"
 src="product_1296x.jpg"
 srcset="
  product_360x.jpg    360w,
  product_540x.jpg    540w,
  .........
  product_1728x.jpg  1728w,
  product_2048x.jpg  2048w"
 sizes="45vw" />
```

- On liste ses options au navigateurs dans **srcset**.
- Il multiplie
  - le viewport par
  - le Device Pixel Ratio par
  - la valeur de **sizes**

et sélectionne la plus proche

Démo simple

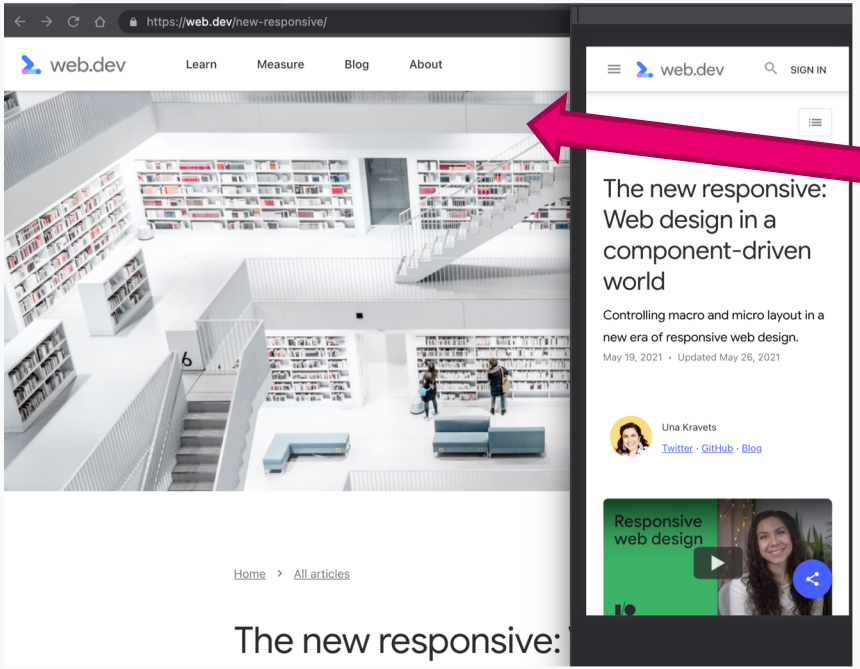
Résumé images responsive (en HTML pur)

- Super efficace, mangez-en
- ☢️ Attention aux DPR > 2
- Autres bénéfices : compatible lazy-loading, référencement, rétro-compatible

Accélérer une image visible : quelles options ?

- ✅🚄 Preload
- ✅🧚 Réduire le poids (challenge : sans compression)
- 👉🥇 Déprioriser tout le reste

Design responsive : problème ou opportunité ? ⚖️



- Le design desktop prévoit une grande image ? Vérifiez qu'elle n'est **pas** chargée sur mobile !

- Démo
  - web.dev/new-responsive/

Design responsive : l'opportunité

```
<picture>
 <source
   media="(min-width: 481px)"
   srcset="bandeau_desktop.jpg"
/>
 <img class=hero  alt="Achète" />
</picture>
```

- Éviter un téléchargement dans certaines résolutions
- Démo simple bandeau large
- NB : `IMG` reste nécessaire pour porter les attributs. Mais `src` est vide

Un attribut et c’est fini ?

LE LAZY-LOADING

La simplicité incarnée 😇

```
<img    src=chatons.jpg
        loading=lazy    />
```

Mais … 😈

- Support absent :
  - IE 11 (mais l'image s'affiche)
  - Safari : 2022
- Pièges d'intégration :
  - Pas sur toutes les images !
  - Prévoir l'espace !

Pas sur toutes les images !

- WordPress l'a appliqué par défaut à grande échelle ➔ Boum

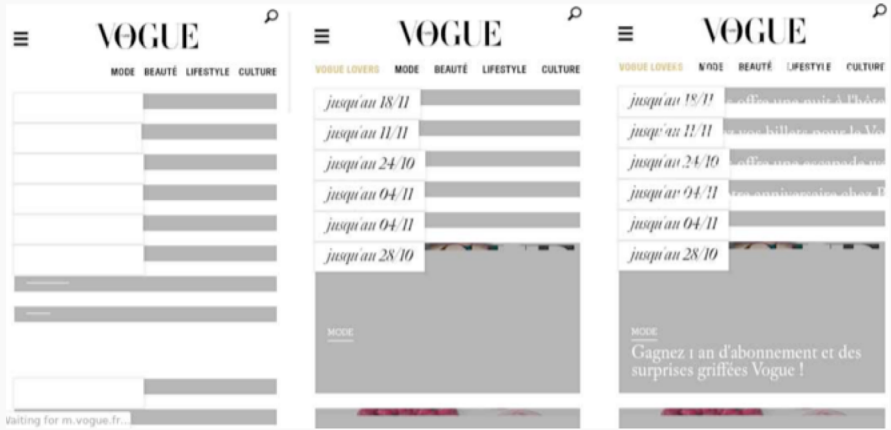
Images importantes :

- **Ne pas** mettre l'attribut `loading`,
  - Éventuellement la valeur `eager` (défaut)

```
<img   src=important.jpg
       loading=eager          />

<img   src=secondaire.jpg
       loading=lazy   />
```


Ça bouge 😵‍💫



Lazy loading ou pas, le layout se met à jour à chaque chargement d'image :s

Solutions pour fixer les images en responsive

En HTML

```
<img
   src=chatons.jpg
   width=16
   height=9
/>
```

Note : le navigateur n'a besoin que du ratio L/H

En CSS

```
@media (orientation: portrait) {
.hero img { aspect-ratio: 1 / 1; }
}
@media (orientation: landscape) {
.hero img { aspect-ratio: 16 / 9; }
}
```


Résumé : déprioriser tout le reste 🥇

- 🔪 Surveiller le réseau, à toutes les résolutions
- Le lazy-loading c’est le bien ! Mais intelligemment 🧠
- 👩‍🎨 Pensez au design d’attente

Mais que fait la police

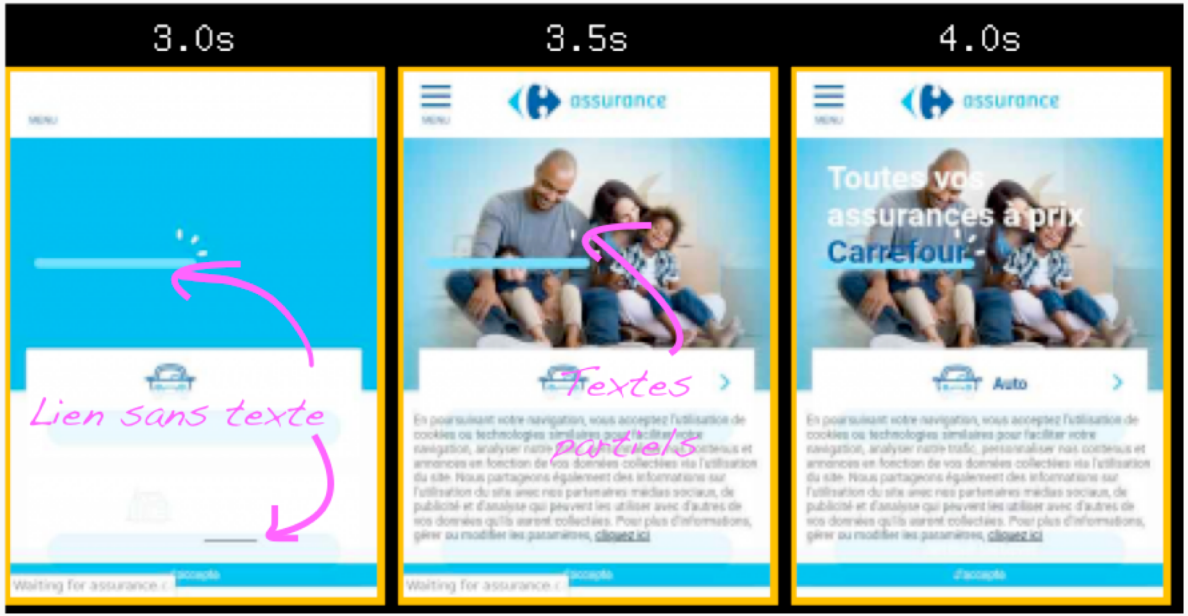
LES POLICES DE CARACTÈRES

Que faire en attendant la police ? 🚓

Chrome, Firefox : **texte invisible** pendant **3 s** max

iOS : **page blanche**

IE 9+ : **swap**



Déclaration optimale (IE9+)

```
@font-face {
  font-family: 'Police';
  font-weight: normal;  font-style: normal;
  src:local('Police Name Regular'),
      local('PoliceName-Regular'),
      url('police.woff2') format('woff2'),
      url('police.woff') format('woff');
  font-display: swap;
  unicode-range(U+0020-U+007E, U+00A0-U+00FF, U+20AC);
}
…
.element { font-family: 'Police', sans-serif ; }
```

Tenter sa chance

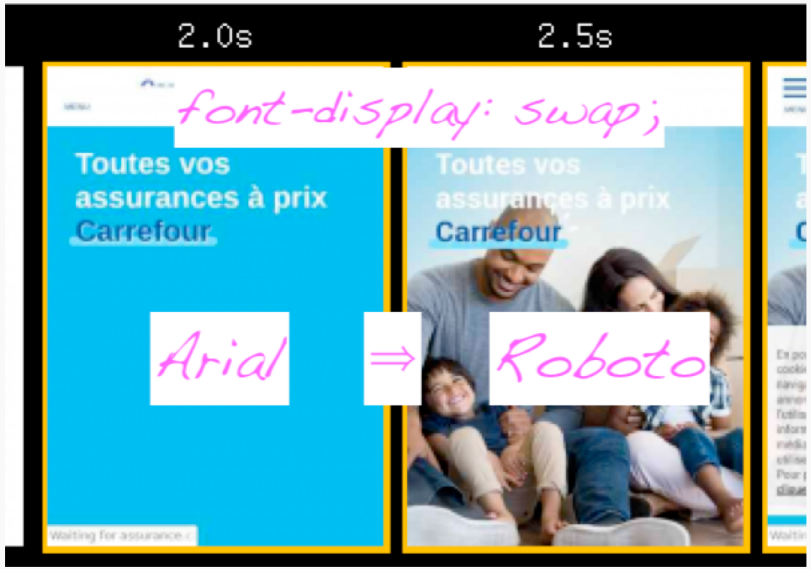
Woff 2 en premier

Stratégie d'attente.
Swap = affichage immédiat

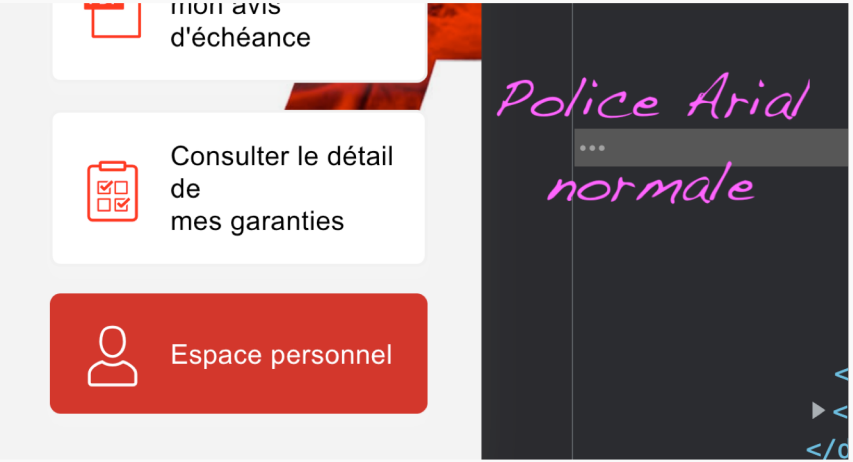
Les glyphes de ce fichier

Police de repli

Quand le swap marche bien 🎉



Quand le swap ne marche pas 🤐



Maîtriser le swap : size-adjust + local()

```
@font-face {
  font-family: 'Police-repli';
  src: local('Arial');
  size-adjust: 106.8%;
}
.element {
  font-family: 'Police', 'Police-repli',
sans-serif;
}
```

Police universelle

Correction des dimensions

Nouvelle fallback

Conclusion Polices

- ⚡ Choisir une stratégie d’affichage (swap)
- 🏋️ Enlever l’inutile
- 👷 Corriger les défauts

Résumé

- Investissez dans vos outils pour connaître le déroulé de l'affichage : DevTools, WebPagetest, les services payants …
- Compétences d'intégration : vital dans toute équipe visant la qualité

Questions ? 🙋

BrainCracking

- Article : 24joursdeweb.fr, jour 6
- Slides et vidéo : suivez @_welovespeed ou @theystolemynick
- Discuter
  - Slack EN : webperformance.herokuapp.com
  - Slack FR : welovespeed.herokuapp.com

- Formation, audit, accompagnement ? jp@braincracking.fr