WordCamp Lyon 2022

Embarcadere-Public
Embarcadere!

Optimiser la web performance d'un site WordPress

Eroan Boyer

40 min • Retour d'expérience - Performance

PLANNING

Qui suis-je ?

- ➔ Fondateur de l'Agence Web Performance
- ➔ Expert en web performance
- ➔ Développeur web front-end & WordPress
- ➔ Traducteur FR d’extensions WordPress
- ➔ Certifié Opquast Expert

@Eroan

Eroan Boyer

eboyer.com

agencewebperformance.fr

WordPress et la web performance

Un peu d'histoire et de contexte…

Les étapes marquantes

Naissance de WordPress (v0.7)

2003

2007

2009

Petites optimisations de la vitesse, surtout liées aux **requêtes** (v2.2, 2.8 et 3.2).

Introduction du **Responsive Web Design** et de la logique **mobile first** (v3.8)

2013

2019

Lancement de l'éditeur natif **Gutenberg** et de son système de blocs (v5.0).

Core Web Vitals

Création d'une **Performance Team** pour "coordonner les efforts pour améliorer la performance / vitesse"

2021

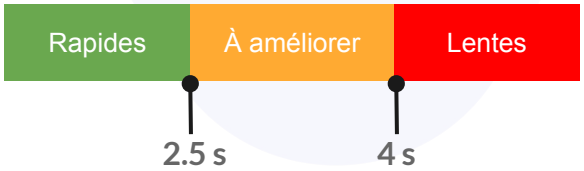
2022

Les Core Web Vitals

LCP

Largest Contentful Paint

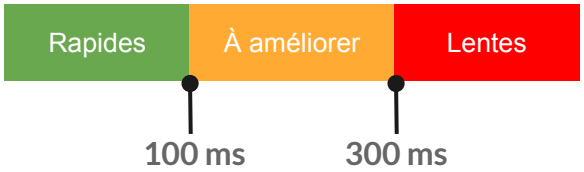
Mesure le temps nécessaire au **principal élément visible** de la page pour s'afficher. Il est assez **représentatif de la vitesse globale** d'une page.



FID

First Input Delay

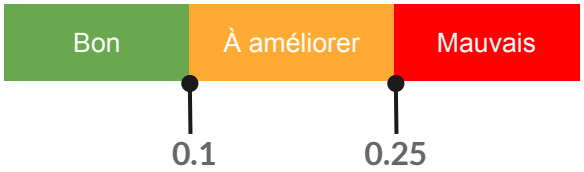
Mesure le temps nécessaire à une page pour être **interactive de façon fluide**. Il est directement lié à JavaScript.



CLS

Cumulative Layout Shift

Mesure la capacité d'une page à proposer une interface dont **les éléments restent stables visuellement** dans le temps.



TTFB

Time To First Byte

Mesure le temps nécessaire à un serveur pour répondre à la première requête du navigateur.

Pourquoi l'approche performance est-elle importante ?

Expérience Utilisateur

- Performance Marketing
- ⬆️ Conversions
- ⬇️ Rebonds

Amélioration SEO

- ⬆️ de visibilité
- ⬆️ du Crawl
- ⬆️ de fréquentation

Efficience / Résilience

- Évolutivité
- ⬇️ de la dette technique
- ⬆️ de la durée de vie

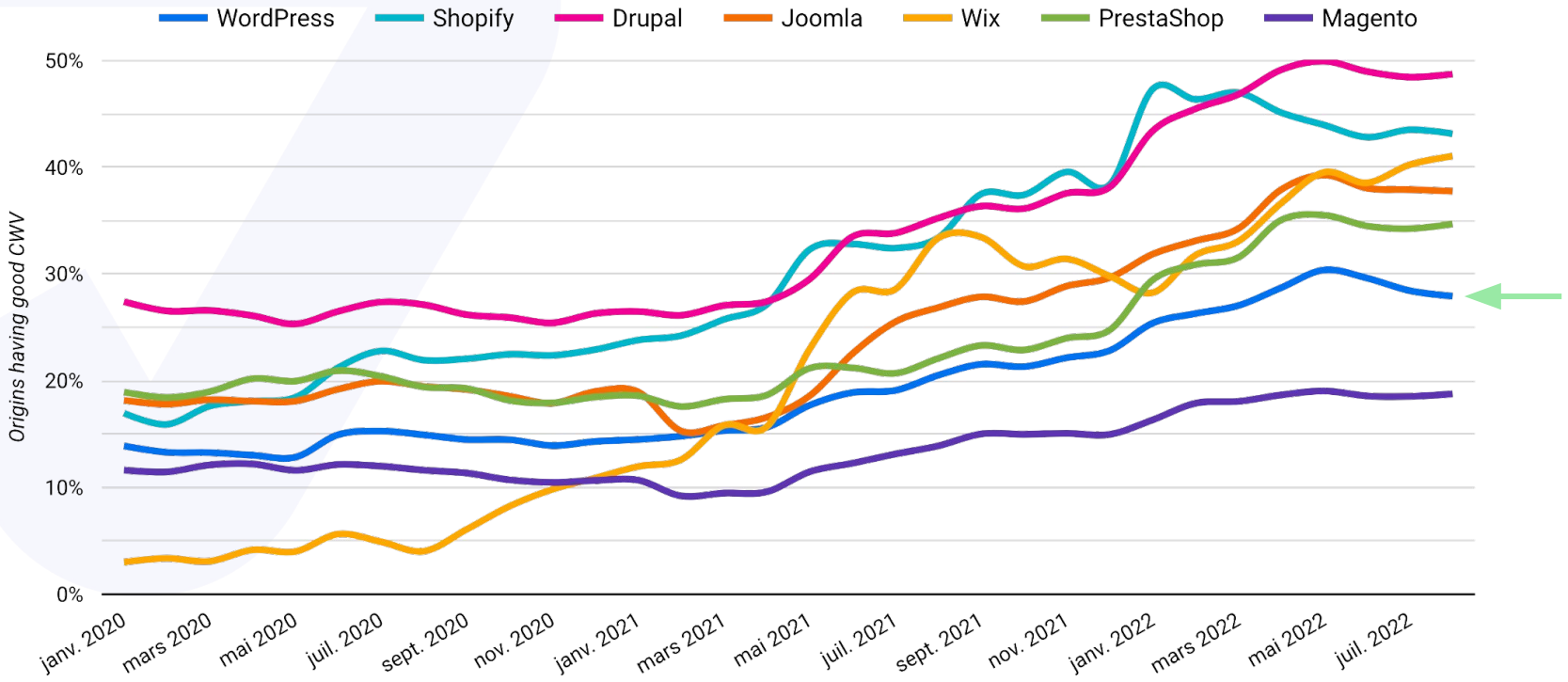
Environnement 🌱

- Décarbonation
- Éco-conception

WordPress est-il « un CMS performant » ?

Analysons ensemble quelques chiffres…

WordPress vs. autres CMS

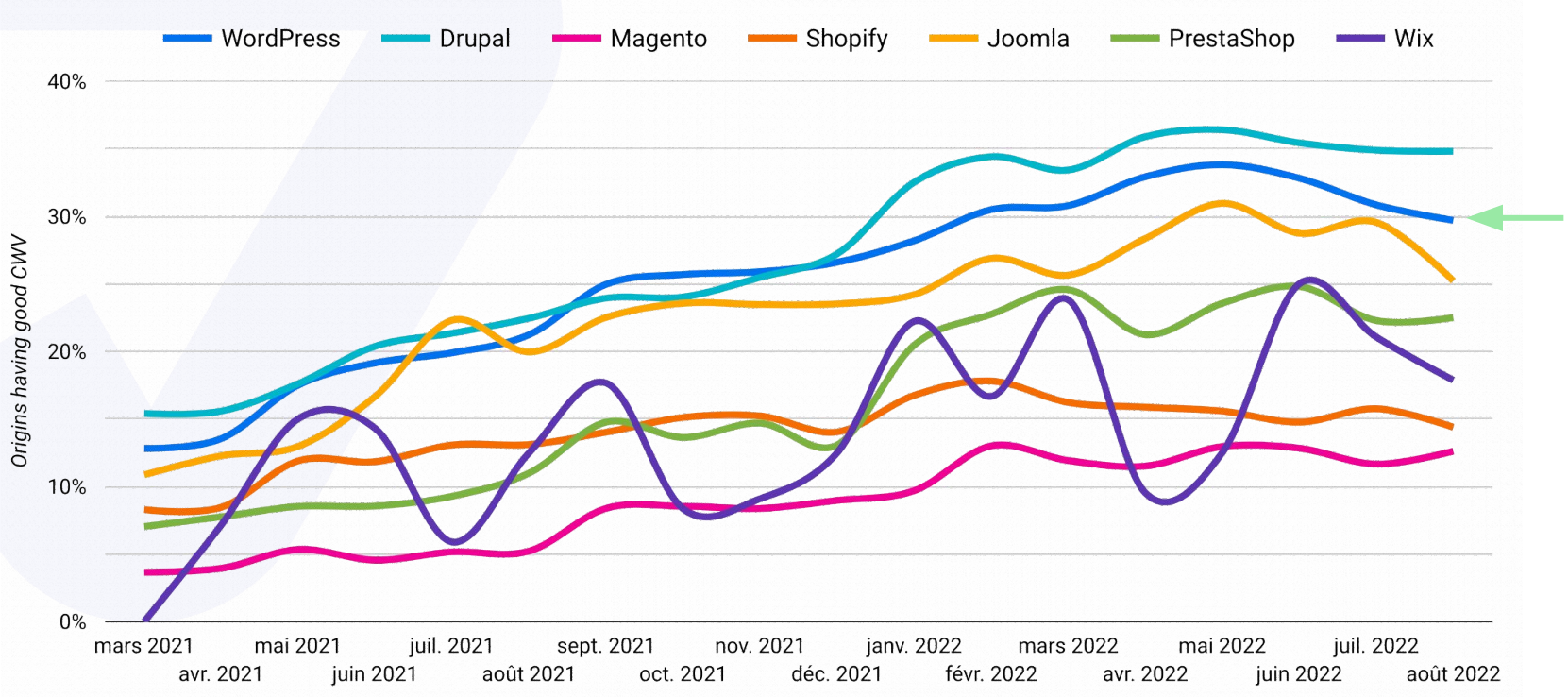


Source : Données CrUX mobile, **100 millions de sites**, toutes origines géographiques (lien).

Quelles conclusions en tirer ?

Gare aux conclusions hâtives : cela ne signifie pas que WordPress n'est pas performant, mais plutôt que ses utilisateurs n'ont **pas de sensibilité particulière** aux problématiques de web performance.

WordPress vs. autres CMS



Source : Données CrUX mobile, **100 000 top sites**, toutes origines géographiques (lien).

On peut donc avoir un site WordPress performant !

Voyons ensemble comment les choix du thème, du constructeur de page et des extensions influencent la web performance…

L'hébergement

Les indispensables

Une config CPU & RAM bien dimensionnée ;

Un SLA aligné sur les objectifs (> 99.5%) ;


Compression : **Brotli** plutôt que Gzip (+20%) ;

Version HTTP : **h2** minimum, h3 si possible ;

Version TLS : **1.2** minimum, **1.3** si possible ;

Version de php : 7.4 minimum, **8.0** idéalement. (de 1 à 47% de gain de performance)

Pour aller plus loin

➔ Serveur web : **Nginx** ou **LiteSpeed** plutôt qu'Apache ;

➔ Cache objet **Redis** (transients, options…) ;

➔ Un **cdn** peut combler les lacunes d'un hébergement peu performant.

Moins on en met, plus c'est rapide !

Dans l'écosystème WordPress, l'approche Web Performance consiste dans un premier temps à **se tourner vers les bons outils** et à **ne pas en multiplier le nombre**.

La démarche est complexe car :

- ➔ Cela sous-entend de connaître ou d'avoir **testé les différentes solutions** . Le **questionnement** et la mise en perspective sont essentiels : gare aux faux comparatifs en ligne ;
- ➔ Pour un non développeur, **installer un outil** pour résoudre une problématique est facile ;
- ➔ Les affinités des parties prenantes successives créent souvent un passif, qui devient rapidement une forme de **dette technique**.

Entre 15 et 20 extensions

Le thème

Les pièges à éviter

Choisir un thème pour son **look**.

Choisir un thème pour des **fonctionnalités** très spécifiques.

Acheter un thème **“usine-à-gaz”** sur une boutique de thèmes

La démarche performance

Se tourner vers un thème **généraliste et évolutif** comme Astra, GeneratePress, Blocksy… Ils intègrent des mécanismes de crochets avancés.

Privilégier un thème **populaire** du répertoire WordPress officiel : support, mises à jour, documentation, communauté…

Utiliser des **extensions** pour des fonctionnalités très spécifiques.

Les extensions

Les pièges à éviter

Multiplier les extensions qui chargent du **CSS** et du **JavaScript** sur le site (ex : Contact Form 7).

Installer une extension pour le moindre **besoin**.

La démarche performance

Privilégier les extensions **populaires** du répertoire WordPress officiel.

Se tourner vers les extensions les plus **spécifiques** possibles : éviter les couteaux Suisses tout-en-un.

Lire la documentation et **paramétrer** l'extension en fonction des besoins.

Activer les **mises à jour** automatiques ou mettre en place une routine.

Constructeur de Page, ou “Page Builder”

Les pièges à éviter

Adopter un Constructeur de Page juste **parce que c’est celui** du thème acheté.

Installer de multiples **extensions** pour le constructeur de page.

Rédiger ses **articles de** blog avec un Constructeur de Page.

La démarche performance

Adopter l’éditeur **Gutenberg**, avec si besoin des librairies de blocs comme GenerateBlocks, Spectra, Ultimate Blocks…

Adopter **Divi** ou **Elementor** avec leurs thèmes respectifs Divi et Hello par **choix**.

Comment améliorer la performance d'un site WordPress ?

Grâce aux fonctionnalités d'extensions spécialisées.

Mettre en place un système de cache

Objectifs

➔ Réduire le **temps d'affichage** des pages publiques pour les visiteurs et les robots d'indexation (optimisation du crawl budget) ;

➔ Soulager le **CPU** du serveur en diminuant le volume de scripts exécutés par php et les requêtes SQL.

Les métriques impactées

➔ **TTFB** (Time To First Byte) : le temps de réponse de votre serveur est drastiquement réduit car il n'a plus qu'à servir des pages déjà générées auparavant.


W3 Total Cache


WP Super Cache


WP Rocket

Supprimer les ressources non utilisées

Objectifs

- ➔ Ne charger que les ressources CSS et JavaScript **utilisées** dans la page courante ;
- ➔ Ne charger que les webfonts utilisées, tout particulièrement les **icon-fonts** ;
- ➔ Supprimer les éventuelles ressources chargées en **doublon** (jQuery, GTM, CSS…).

Les métriques impactées

- ➔ **TBT** (Total Blocking Time) et **FID** (First Input Delay) : réduire le volume de JavaScript soulage le CPU, assurant une meilleure UX ;
- ➔ **FCP** (First Contentful Paint) et **LCP** (Largest Contentful Paint) : réduire le volume de CSS accélère le rendu de la page.

Plugin Load Filter

Asset CleanUp

Passer les JavaScript en “defer” ou report d’exécution

Objectifs

- ➔ Privilégier à tout prix **le rendu** de la page ;
- ➔ Exécuter en priorité les JavaScript **propres**, et seulement dans un second temps les JavaScript **tiers** de type tracking.

Les métriques impactées

- ➔ **LCP** (Largest Contentful Paint) : en évitant les blocages du rendu et en supprimant certains JavaScript secondaires, on accélère l’affichage des pages.
- ➔ **TBT** (Total Blocking Time) : en reportant l’exécution d’une partie des scripts, on limite les risques de Blocking Time et de Long Tasks.

Perfmatters Autoptimize WP Rocket

Combiner et minifier les CSS

Objectifs

- ➔ Ne charger que ce qui est **utile** pour l'utilisateur : on supprime les commentaires, SourceMaps… ;
- ➔ Profiter au maximum de la compression **Gzip** ou **Brotli**, dont l'efficacité augmente avec le poids : 10 fichiers de 1 ko combinés ne pèsent que 8,5/9 ko une fois compressés ;
- ➔ Réduire le poids des **CSS tiers**, qu'on ne maîtrise par nature pas.

Les métriques impactées

- ➔ **FCP** (First Contentful Paint) et **LCP** (Largest Contentful Paint) : réduire le volume de CSS accélère le rendu de la page.

Perfmatters Autoptimize WP Rocket

Lazy-loader les images et iframes hors du viewport

Objectifs

- ➔ Ne charger que ce qui est **visible** par l'utilisateur, et donc au-dessus de la ligne de flottaison.
- ➔ Libérer de la bande passante pour les ressources nécessaires au **rendu initial** des pages.

Les métriques impactées

- ➔ **LCP** (Largest Contentful Paint) : en réduisant le nombre et le poids des images téléchargées, la bande passante libérée devient disponible pour des ressources plus importantes.
- ➔ **TBT** (Total Blocking Time) : en ne chargeant plus les iframes externes dès le départ, les JavaScript qu'elles intègrent ne s'exécutent pas dans la foulée de celles de la page (un embed YouTube = 2,5 Mo de scripts non compressés à exécuter).

Perfmatters Autoptimize WP Rocket

Passer les Google Fonts en local

Objectifs

- ➔ Éviter les **connexions** à des serveurs tiers, toujours très coûteuses (dns, connexion, ssl…) ;
- ➔ Profiter de la parallélisation http2 ;
- ➔ S'ouvrir la porte à des **optimisations** complémentaires (preload, fonte de fallback…) ;
- ➔ S'affranchir de l'écosystème de **Google**.

Les métriques impactées

- ➔ **LCP** (Largest Contentful Paint) : en supprimant les latences liées à la connexion distante, on accélère les temps de rendu ;
- ➔ **CLS** (Cumulative Layout Shift) : en chargeant les fontes plus tôt, on réduit les risques de Layout Shifts liés aux textes.

Local Google Fonts

Perfmatters

Précharger les ressources importantes

Objectifs

➔ S’assurer que les ressources **nécessaires** au bon fonctionnement d’une page sont disponibles le plus tôt possible : webfonts (en woff2) et image de LCP typiquement.

Les métriques impactées

➔ **LCP** (Largest Contentful Paint) : en pré-chargeant une image qui constitue le LCP ou la webfont utilisée pour le paragraphe de LCP, on améliore de façon significative l’indicateur.


Pre* Party Resource Hints


Perfmatters


WP Rocket

Optimiser le poids des images

Objectifs

➔ Réduire au maximum le **poids** des images :

- un choix de format adapté : jpeg pour les photos, png voire svg pour les logos et icônes ;
- des dimensions cohérentes : on téléverse des images aux dimensions souhaitées ;
- un enregistrement “pour le web” dans les outils d’édition et/ou une optimisation via un outil.

Les métriques impactées

➔ **LCP** (Largest Contentful Paint) : en réduisant le poids des images, on améliore significativement la vitesse d’affichage de la page.


Imagify


ShortPixel Image Optimizer


Smush

Servir les images dans des formats modernes

Objectifs

➔ Réduire au maximum le **poids** des images. Cela ne doit pas être systématique : seules les alternatives Webp et/ou Avif plus légères doivent être générées. À défaut, la démarche est parfois **contre-productive**.

```php
add_filter('imagify_keep_large_webp', '__return_false');
```

Les métriques impactées

➔ **LCP** (Largest Contentful Paint) : en réduisant le poids des images de façon importante (jusqu'à -70%), on améliore significativement la vitesse d'affichage de la page.

Imagify ShortPixel Image Optimizer Smush

Ajouter des dimensions aux images en html

Objectifs

- ➔ S’assurer que l’interface d’une page est **stable visuellement** tout au long de son utilisation.
- ➔ Offrir une Expérience Utilisateur de qualité.

Les métriques impactées

- ➔ **CLS** (Cumulative Layout Shift) : en permettant au navigateur de “réserver” l’espace occupé par les images (grâce à un calcul de ratio d’affichage), on supprime tout risque de mouvement horizontal ou vertical durant leur chargement.

Sensibiliser les parties prenantes à la web performance

La meilleure optimisation durable, c'est finalement de sensibiliser les différents intervenants aux problématiques de webperf !

Mise en place de process et routines

Monitoring de la performance

Création d'une équipe transverse

WordCamp Lyon 2022

Embarcadere-Public
Embarcadere!

Merci à tous

Des questions ?

PLANNING