Et pour le web, on fait comment ?
Développement web | Chapitre 3

Différentes bonnes pratiques, recensées dans le livre "**Écoconception web : les 115 bonnes pratiques**" permettent de réduire l'empreinte environnementale des services web. Ces bonnes pratiques doivent intégrer les **cycles de développement** pour se généraliser et devenir la norme à long terme. L'importance de l'impact du service peut aussi devenir une **spécification non-fonctionnelle** sanctionnant la **qualité globale du service**.

Utiliser la simple *quote* au lieu des guillemets

Si la chaîne de caractères ne contient pas de variable, ne pas utiliser les guillemets permet à PHP de ne pas chercher les **variables à substituer**, ce qui réduit la consommation de **cycles CPU**.

Stocker localement les données statiques

- Web Storage, cache, IndexedDB, Service Workers avec HTML5.
- Éviter les allers-retours inutiles avec le serveur **améliore la performance** du site.

Limiter le nombre de CSS et les compresser

Limiter le nombre de fichiers échangés permet de ne pas **surcharger le réseau** et de réduire le nombre de **requêtes HTTP.** Les compresser et les minifier systématiquement permet de réduire significativement leur taille.

Éviter la réécriture des getter / setter natifs

Implémenter des méthodes spécifiques métiers en utilisant les getter / setter standards permet de **limiter le temps** de **compilation** et d'**exécution** des méthodes.

Choisir le format de données adapté

Le type de données utilisées a un impact sur la consommation de mémoire et de cycles processeur. Pour les petites variables par exemple, il est intéressant d'utiliser des **TINYINT** (1 octet). Il est possible de **réduire par 8 sa consommation** en bande passante et en mémoire.

Ne pas modifier le DOM lorsqu'on le traverse

Cette action peut entraîner des situations où la boucle devient très gourmande en termes de **cycles CPU.** Ajouter des éléments au DOM en le traversant crée des boucles infinies.

Favoriser les pages statiques

Cela permet d'éviter énormément de **calculs PHP** au serveur à chaque requête. Chaque utilisateur se voit alors servir une **même page stockée dans un cache du serveur** et disponible à la demande.

Utiliser les forks applicatives orientées performance

- **Percona Server** au lieu de MySQL.
- **Pressflow** au lieu de Drupal.

Mettre en cache le code intermédiaire

Certains accélérateurs permettent de mettre en cache le **bytecode**, ce qui évite de recompiler à chaque fois à partir du code source.
Une **liste des accélérateurs** est disponible sur Wikipédia.

Gestion des variables

- Utiliser des **variables locales** en argument des routines pour éviter des vérifications couteuses.
- **Libérer de la mémoire** les variables qui ne sont plus nécessaires.

- Stocker les références des éléments du DOM dans une **variable JS** pour ne pas parcourir plusieurs fois tout le DOM.
- Ne pas appeler de fonction dans la **déclaration d'une boucle for**.

- Ne jamais faire de **requêtes SQL** dans une boucle.
- Utiliser des **variables statiques**.
- Supprimer tous les **warnings** et **notices**.
- Remplacer les $i++ par des **++$i**.