Engineer

Générateur de sites statiques

un programme qui écrit chaque page à l'avance

Ce que c’est

Un générateur de sites statiques est un outil de génération qui prend un dossier de fichiers sources, généralement des pages écrites en Markdown et un ensemble de modèles, puis les transforme en un site terminé, composé uniquement de fichiers : du HTML, des feuilles de style, des scripts et des images qu’un serveur web envoie tels quels. Le terme est souvent abrégé en SSG, et « statique » décrit la manière dont le résultat est servi, avec les mêmes octets pour toute personne qui demande une adresse, pas le comportement de la page une fois arrivée dans le navigateur.

Il s’oppose à l’approche dynamique d’un système de gestion de contenu ou d’une application web, où un programme sur le serveur assemble chaque page à la demande, généralement en interrogeant une base de données et en remplissant des modèles. Un générateur applique le même genre de modèles au même genre de contenu, mais une fois par modification plutôt qu’une fois par visite. Il laisse le résultat dans un dossier qu’un serveur web ordinaire, un compartiment de stockage ou un réseau de diffusion de contenu peut servir sans savoir comment les pages ont été fabriquées.

Hugo est l’un de ces générateurs. Sa documentation le présente comme « un générateur de sites statiques écrit en Go, optimisé pour la vitesse et conçu pour la souplesse », qui « génère un site complet en quelques secondes, souvent moins ». Le cycle habituel consiste à écrire pendant que son serveur web intégré tourne, régénère le site à chaque modification et recharge les pages ouvertes dans le navigateur, puis à générer le site et à le déployer chez un hébergeur :

```
hugo server     # build, serve, rebuild and reload on every save
hugo --minify   # write the finished site into public/
```

Utilisez le concept pour prendre une décision

Utilisez « Qu'est-ce qu'un générateur de sites statiques ? » pour éclairer une décision réelle. Nous pouvons vous aider à relier ce concept au système que vous construisez ou achetez. Des interfaces rapides, utilisables sur téléphone et cohérentes dans tous les navigateurs de vos clients.

Découvrir nos prestations Contactez-nous →

Vérifiez les hypothèses derrière le terme

Appuyez-vous sur l'explication et les réalisations liées pour formuler les questions de votre projet. Nous pouvons examiner les hypothèses avec vous avant le choix d'un outil. Développement cartographique GIS full-stack — cartographie full-stack, de la base au navigateur

Développement cartographique GIS full-stack Contactez-nous →

Pourquoi c’est important

Le choix porte sur le moment où le travail se fait. Un générateur le déplace du traitement de la requête à la génération du site, et presque tout le reste en découle.

Envoyer un fichier est l’une des opérations les moins coûteuses pour un serveur web. Un site généré répond donc vite et peut fonctionner sur une très petite machine. Comme toute personne demandant une adresse reçoit les mêmes octets, la mise en cache est simple. Sans code applicatif ni base de données en fonctionnement derrière les pages, il y a bien moins à sécuriser et à mettre à jour. Le résultat est un dossier : changer d’hébergeur revient à copier des fichiers. Les fichiers sources sont conservés dans un système de gestion de versions, où chaque modification de chaque page peut être examinée et retracée.

Les coûts viennent du même choix. Chaque modification, même la correction d’une coquille, impose une nouvelle génération et un déploiement : la vitesse de génération compte donc davantage à mesure que le nombre de pages augmente. Tout ce qui dépend de la personne qui fait la demande, comme la recherche, les commentaires, l’authentification ou un formulaire qui conserve ce qu’il reçoit, exige du code dans le navigateur, un service séparé ou une solution calculée à la génération, puisqu’aucun programme sur le serveur ne s’en charge. Le temps s’arrête entre deux générations : par défaut, Hugo omet une page dont la date est future. Une page prévue pour lundi apparaît donc lors de la première génération effectuée lundi ou après. Même les redirections changent de forme. Faute de programme pour envoyer une redirection HTTP, Hugo crée par défaut, pour une page déplacée, un petit fichier HTML à l’ancienne adresse qui redirige le navigateur. Sa documentation décrit cette approche comme portable chez tous les hébergeurs.

À quoi cela ressemble en pratique

Ce site est construit avec un tel outil. La page des crédits nomme Hugo comme générateur de sites statiques et lui attribue les modèles, les taxonomies, toutes les langues et les flux. À propos de ce qui arrive au navigateur, elle précise que le HTML est écrit intégralement avant toute demande : chaque page est un fichier sur disque, pas un modèle assemblé à chaque requête. Même l'apparence est décidée à la génération : le fond du thème clair est tiré au sort pendant la génération du site, et le site change de tenue à chaque publication. Le contrôle qualité obligatoire s'exécute localement, sur l'ordinateur où le site est modifié, sans service de génération hébergé dans la chaîne. Chaque diagnostic y compte comme une erreur, et, en octobre 2026, le site générait 2 268 pages. Elles sont servies par un serveur à un cœur doté de 512 Mo de mémoire, où le serveur web et les trois autres services qui répondent pour le site lisent le même dossier de fichiers : un seul déploiement met ainsi les quatre à jour.

L'étude de cas sur la construction de ce site avec Hugo documente l'étape de réalisation d'août 2026 et en explique les raisons. Trois langues, aucun service applicatif facturé séparément et aucun visiteur confié à un tiers ont éliminé presque tous les systèmes de gestion de contenu : un backend en fonctionnement doit être sécurisé, corrigé, payé et expliqué sur une page de confidentialité. Le site comptait alors 108 fichiers de modèles, dont 53 modèles partiels, publiait chaque page en quatre représentations à partir d'une seule arborescence de contenu et construisait encore sept fois cette arborescence pour des sous-domaines thématiques, en indiquant au générateur un autre dossier de contenu. Le récit expose le compromis lié à l'ajout de fonctionnalités à la génération plutôt que dans le navigateur : hormis l'exception JavaScript documentée et limitée à un filtre du portfolio, les fonctionnalités interactives devaient être résolues à la génération ou omises, c'est pourquoi le coût d'un index de recherche a été estimé et le projet mis de côté plutôt que livré.

Où les problèmes surviennent

Une page supprimée reste en ligne. Une page retirée des sources continue d'être servie à son ancienne adresse, et les moteurs de recherche continuent de la trouver. Hugo ne vide pas son dossier de sortie avant une génération : les fichiers existants sont écrasés, jamais supprimés. `public/` peut donc conserver des pages de générations précédentes, notamment des brouillons et des pages expirées, et tout outil qui copie ce dossier sur un serveur les copie aussi. Le cache `resources/` présente un problème voisin : les règles de conception de ce site avertissent qu'il peut servir un ancien ensemble CSS lorsque seules les sources

CSS ont changé. Générer dans un dossier propre, avec l'option `--cleanDestinationDir` de Hugo ou une étape de suppression dédiée ; la génération de ce site commence par supprimer `public/` et `resources/` .

Une page terminée n'apparaît jamais. Un article est écrit, enregistré dans Git et déployé, la génération annonce un succès, mais le site n'en contient aucune page. Par défaut, Hugo omet le contenu marqué comme brouillon, celui dont la date ou la date de publication est future et celui dont la date d'expiration est dépassée. Aucun de ces cas ne compte comme une erreur. Une date saisie avec un jour d'avance suffit. Vérifier d'abord ces quatre champs, réserver `--buildDrafts` , `--buildFuture` et `--buildExpired` aux prévisualisations et prévoir, pour une page programmée, une génération après sa date.

Toutes les pages paraissent avoir changé en même temps. Le sitemap attribue presque partout la même date de dernière modification : il dit au robot que tout le site a changé, sans indiquer quelles pages ont réellement changé. Un générateur doit tirer cette date de quelque part, et l'horodatage du fichier est une mauvaise source lorsqu'une étape réécrit des fichiers dont le contenu ne change pas. La configuration de ce site indique que sa synchronisation réécrit les 189 fichiers de contenu à chaque exécution : 172 des 176 adresses du sitemap partageaient ainsi le même horodatage. Prendre plutôt la date dans l'historique des versions : Hugo lit la date de la dernière révision Git d'un fichier lorsque `enableGitInfo` est activé, et une nouvelle synchronisation produisant des octets identiques ne crée aucun commit. Après un mois de modifications, une resynchronisation n'a donc touché que 8 fichiers sur 186.

Des fichiers que personne ne voulait publier deviennent accessibles. Quiconque en devine le nom peut télécharger un fichier qui en énumère d'autres. Un générateur publie tout ce qui se trouve dans les dossiers qu'il lit, sauf ce qu'une règle d'exclusion retire ; si cette règle n'atteint pas tous les dossiers, le reste est exposé. La configuration de ce site indique que sa liste globale d'exclusions n'atteignait pas un montage de module : chaque génération publiait sur tous les hôtes les six fichiers `.DS_Store` laissés par le Finder de macOS dans icons/, images/ et webmanifest/. Chacun énumérait les noms des éléments du même dossier, et celui d'images/ révélait donc un dossier que le site garde privé. Un filtre privé d'une seule espace après son `!` a, une autre fois, exclu toutes les images d'un montage. Exclure sur chaque montage qui publie, puis chercher dans la sortie générée les fichiers qui n'y ont pas leur place ; deux contrôles de ce site font échouer une génération qui en sert un.

Pour aller plus loin

- Hugo : Introduction — ce qu'est Hugo, ses usages et son cycle d'écriture, de génération et de déploiement.
- Hugo : Basic usage — générer et servir un site, pourquoi `public/` conserve d'anciens fichiers et quels contenus une génération omet.
- Hugo : Configure front matter — comment Hugo choisit les dates d'une page, notamment avec les marqueurs `:git` et `:fileModTime`.
- Hugo : Configure modules — les montages, le filtre `files` propre à chacun et la règle selon laquelle définir un montage supprime celui par défaut.
- Hugo : URL management — les alias et les fichiers de redirection qu'un hôte statique sert à la place d'une redirection du serveur.

Glossaire

Choisissez selon vos contraintes réelles

Présentez le processus de travail, les interfaces existantes et les utilisateurs qui ont besoin du changement. Nous pouvons définir une livraison testable et une transmission permettant à votre équipe d'assurer la maintenance des éléments livrés.

Consulter les tarifs Contactez-nous →

Rendez la terminologie concrète

Demandez comment « Qu'est-ce qu'un générateur de sites statiques ? » s'applique à votre organisation. Un exemple concret tiré de votre système actuel suffit pour engager la discussion.

Découvrir nos prestations

welcome@engineer.company +45 50 10 61 84 Copenhague, Danemark

Emportez cette page avec vous

Fiche de contact

Créé avec soin et amour par D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Droit d'auteur © 2025 – aujourd'hui · Engineer ApS · Tous droits réservés.

Imprimé depuis https://engineer.company/fr/glossary/ssg/