Engineer

Statisk sitegenerator

⚡ et program der skriver hver side ud på forhånd

Hvad det er

En statisk sitegenerator er et byggeværktøj, der tager en mappe med kildefiler, som regel sider skrevet i Markdown og et sæt skabeloner, og gør dem til et færdigt websted, der kun består af filer: HTML, stylesheets, scripts og billeder, som en webserver sender præcis, som de er. Betegnelsen forkortes ofte til SSG, og »statisk« handler om, hvordan resultatet bliver leveret, nemlig de samme bytes til alle, der beder om en adresse, ikke om, hvordan en side opfører sig, når den først er nået frem til en browser.

Den står over for den dynamiske tilgang i et indholdsstyringssystem eller en webapplikation, hvor et program på serveren sætter hver side sammen, når nogen beder om den, som regel ved at slå op i en database og udfylde skabeloner. Statiske sitegeneratorer kører den samme slags skabeloner over den samme slags indhold, men én gang pr. ændring i stedet for én gang pr. besøg, og efterlader resultatet i en mappe, som en almindelig webserver, en lagerbucket eller et content delivery network kan levere uden at vide, hvordan siderne blev til.

Hugo er en af dem. Dokumentationen præsenterer den som en statisk sitegenerator skrevet i Go, bygget til fart og fleksibilitet, der laver et helt site på sekunder og ofte hurtigere. Den sædvanlige arbejdsgang er at skrive med Hugos indbyggede webserver kørende, så sitet bygges igen ved hver ændring, og de åbne sider i browseren genindlæses, og bagefter bygge sitet og udrulle det til en vært:

```
hugo server     # byg, servér, byg igen og genindlæs ved hver gemning
hugo --minify   # skriv det færdige site ind i public/
```

Brug begrebet i en beslutning

Brug »Hvad er en statisk sitegenerator?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Brugerflader, der indlæses hurtigt, virker på en telefon og opfører sig ens i alle de browsere, jeres kunder nu bruger.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har styret full-stack-udviklingen af GIS-kort — full stack-kortløsning, fra database til browser

Har styret full-stack-udviklingen af GIS-kort Kontakt os →

Hvorfor det betyder noget

Valget handler om, hvornår arbejdet bliver gjort. En generator flytter det fra forespørgslen til bygget, og næsten alt andet følger af det.

At sende en fil er noget af det billigste, en webserver gør, så et genereret site svarer hurtigt og kan køre på en meget lille maskine, og fordi alle, der besøger en adresse, får de samme bytes, er caching enkelt. Uden applikationskode eller database bag siderne er der langt mindre at sikre og opdatere. Resultatet er en mappe, så at skifte vært er et spørgsmål om at kopiere filer, og kilden ligger i versionsstyring, hvor hver ændring af hver side kan gennemgås og spores.

Omkostningerne kommer samme sted fra. Hver ændring, også en rettet tastefejl, kræver et nyt byg og en ny udrulning, så byggehastigheden betyder mere, jo flere sider der er. Alt, der afhænger af, hvem der spørger, som søgning, kommentarer, login eller en formular, der gemmer det, den modtager, kræver kode i browseren, en separat tjeneste eller en måde at blive afgjort på byggetidspunktet, for der kører intet program på serveren, der kan klare det. Tiden står stille mellem to byg: Hugo udelader som standard en side, hvis dato ligger i fremtiden, så en side, der er planlagt til mandag, dukker op med det første byg på eller efter mandag. Selv omdirigeringer skifter form. Uden et program til at sende en HTTP-omdirigering er Hugos standard for en flyttet side en lille HTML-fil på den gamle adresse, som sender browseren videre, en løsning, som dokumentationen kalder portabel på tværs af alle hostingudbydere.

Sådan ser det ud i praksis

Dette site er bygget med en. Kolofonen nævner Hugo som den statiske sitegenerator og giver den æren for skabelonerne, taksonomierne, alle sprogene og feedsene. Om det, der når frem til browseren, siger den, at HTML'en er færdigskrevet, inden nogen beder om den: hver side ligger som en fil på disken i

stedet for at blive sat sammen af en skabelon, hver gang nogen spørger. Selv udseendet ligger fast under bygget, for det lyse tema får trukket én af sine vægge, mens siden bygges, og sitet skifter beklædning ved hver udgivelse. Ingen hostet tjeneste bygger det: kvalitetsporten kører på den maskine, hvor sitet bliver skrevet, hver meddelelse fra et værktøj regnes for en fejl, og i oktober 2026 blev der bygget 2.268 sider. De bliver leveret fra en server med én kerne og 512 MB hukommelse, hvor webserveren deler mappen med de tre andre tjenester, der svarer for sitet, så en enkelt udrulning når frem til alle fire.

Porteføljeindlægget om at bygge dette site i Hugo dokumenterer implementeringsstadiet fra august 2026 og giver begrundelsen. Tre sprog, ingen særskilt faktureret applikationstjeneste og ingen besøgende overladt til tredjeparter udelukkede næsten alle indholdsstyringssystemer, fordi en kørende backend skal sikres og holdes opdateret, koster penge og skal forklares på en privatlivsside. Sitet havde dengang 108 skabelonfiler, heraf 53 partials, udgav hver side i fire repræsentationer fra ét indholdstræ og byggede det samme træ syv gange mere til fokuserede subdomæner ved at pege generatoren på en anden indholdsmappe. Indlægget beskriver afvejningen ved at tilføje funktioner under bygget og ikke i browseren: bortset fra den dokumenterede JavaScript-undtagelse for et afgrænset porteføljefilter måtte interaktive funktioner klares under bygget eller udelades, og derfor blev et søgeindeks regnet igennem og lagt til side frem for at blive udgivet.

Hvor det går galt

En slettet side bliver ved med at være online. En side, der er fjernet fra kilden, bliver stadig leveret på sin gamle adresse, og søgemaskinerne bliver ved med at finde den. Hugo rydder ikke sin outputmappe før et byg: eksisterende filer bliver overskrevet, men aldrig slettet, så `public/` kan beholde sider fra tidligere byg, også kladder og udløbne sider, og alt, der kopierer mappen til en server, kopierer dem med. Hugos `resources/` -cache har en fætter til problemet: dette sites designregler advarer om, at den kan levere et forældet CSS-bundt, når kun kilde-CSS'en er ændret. Byg ind i en ren mappe, med Hugos flag `--cleanDestinationDir` eller et sletningstrin, du selv skriver; dette sites byg fjerner både `public/` og `resources/`, inden det går i gang.

En færdig side dukker aldrig op. Et indlæg bliver skrevet, committet og udrullet, bygget melder succes, og sitet har ingen side til det. Som standard udelader Hugo indhold, der er markeret som kladde, indhold, hvis dato eller udgivelsesdato ligger i fremtiden, og indhold, hvis udløbsdato er passeret, og

intet af det regnes for en fejl. En dato, der er tastet én dag frem, er nok. Tjek de fire felter først, gem `--buildDrafts`, `--buildFuture` og `--buildExpired` til forhåndsvisninger, og sørg for, at en planlagt side får et byg efter sin dato.

Alle sider påstår at være ændret på én gang. Sitemappet giver næsten hver adresse det samme ændringstidspunkt, hvilket fortæller en crawler, at hele sitet er ændret, og intet om, hvilke sider der faktisk er. En generator må hente hver sides ændringsdato et sted fra, og filens eget tidsstempel er den forkerte kilde, når et trin omskriver filer, hvis indhold forbliver det samme. Dette sites konfiguration noterer, at synkroniseringstrinnet omskriver alle 189 indholdsfiler ved hver kørsel, så 172 af de 176 adresser i sitemappet delte ét tidsstempel. Tag datoen fra versionsstyringen i stedet: Hugo læser datoen for en fils seneste Git-revision, når `enableGitInfo` er slået til, og en ny synkronisering, der giver identiske bytes, laver intet commit, så en synkronisering efter en måneds redigeringer rørte ved 8 af 186 filer.

Filer, som ingen ville udgive, ryger med ud. Enhver, der gætter navnet, kan hente en fil, der oplister andre filer. En generator udgiver alt, der ligger i de mapper, den læser, minus det, en udelukkelsesregel fjerner, og en regel, der ikke når alle mapper, efterlader resten åbent. Dette sites konfiguration noterer, at den sitedækkende ignoreliste ikke nåede ind i et modulmount: bygget udgav alle seks `.DS_Store`-filer, som Finder i macOS havde efterladt under icons/, images/ og webmanifest/, på hver vært, og hver af dem oplistede navnene ved siden af sig, så den i images/ navngav en mappe, sitet holder tilbage. Et filter, der manglede ét mellemrum efter sit `!`, udelukkede engang alle billeder i et mount i stedet. Udeluk på hvert mount, der udgiver, og tjek derefter det byggede output for filer, der ikke hører hjemme der; to af dette sites tjek stopper et byg, der leverer en af dem.

Læs mere

- Hugo: Introduction — hvad Hugo er, hvad den bruges til, og hvordan arbejdsgangen fra skrivning over byg til udrulning forløber.
- Hugo: Basic usage — at bygge og servere et site, hvorfor `public/` beholder gamle filer, og hvilket indhold et byg udelader.
- Hugo: Configure front matter — hvordan Hugo vælger en sides datoer, herunder tokenerne `:git` og `:fileModTime`.
- Hugo: Configure modules — mounts, det `files`-filter, hvert af dem tager, og reglen om, at et defineret mount fjerner standardmountet.
- Hugo: URL management — aliasser og de omdirigeringsfiler, en statisk vært leverer i stedet for en serveromdirigering.

Ordliste

Vælg ud fra jeres faktiske vilkår

Beskriv arbejdsgangen, de eksisterende grænseflader og de brugere, der har brug for ændringen. Vi kan definere en testbar leverance og en overdragelse, der gør teamet i stand til at vedligeholde løsningen.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er en statisk sitegenerator?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

welcome@engineer.company +45 50 10 61 84 København, Danmark

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

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

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/ssg/