Statische Website-Generatoren

der goldene Mittelweg zwischen handgeschriebenen Webseiten und Web-CMS

Christian Pietsch
Universitätsbibliothek Bielefeld

Kolloquium Wissensinfrastruktur
21. Oktober 2016

CMS = Content Management System

- hier gemeint: WCMS = Web Content Management System
- d.h. eine Website, deren Inhalt sich per Browser ändern lässt
- bekannteste Open-Source-Beispiele: WordPress, Joomla!, TYPO3, Drupal
- „In den Jahren 2002–2015 wurden 143 Lücken in Drupal, 179 in TYPO3, 170 in Joomla! und 205 in Wordpress eingetragen.“

Quellen:

`https://de.wikipedia.org/wiki/Content-Management-System`

`http://www.heise.de/ix/meldung/Webauftritte-von-Unis-veraltet-unsicher-nicht-barrierefrei-3355069.html`

Abruf einer statischen Webseite

1. Browser: Ich hätte gern `http://www.site.de/a.html`
2. www.site.de:
   - 2.1 Webserver: OK, `/srv/www/a.html` ist vorhanden.
   - 2.2 Webserver: OK, ich darf diese Datei lesen.
   - 2.3 Webserver: Hallo Browser, im Anhang ist die Datei.

Abruf einer dynamischen Webseite

1. Browser: Ich hätte gern `http://www.site.de/a.cgi`
2. www.site.de:
   - 2.1 Webserver: OK, `/srv/www/a.cgi` ist vorhanden.
   - 2.2 Webserver: OK, ich darf diese Datei ausführen.
   - 2.3 Webserver: Hallo Betriebssystem, bitte diese Datei ausführen.
   - 2.4 Betriebssystem: `a.cgi` ist ein Perl-Skript. Hallo Perl?
   - 2.5 Perl: Ich brauche Daten aus der Datenbank.
   - 2.6 Datenbank: Hallo Perl, hier sind die Daten.
   - 2.7 Perl: Moment noch, ich generiere jetzt das HTML ...
   - 2.8 Perl: Hallo Betriebssystem, hier sind die Daten.
   - 2.9 Betriebssystem: Hallo Webserver, hier sind die Daten.
   - 2.10 Webserver: Hallo Browser, im Anhang ist die Datei.

Abruf einer dynamischen Webseite

1. Browser: Ich hätte gern `http://www.site.de/a.cgi`
2. www.site.de:
   - 2.1 Webserver: OK, `/srv/www/a.cgi` ist vorhanden.
   - 2.2 Webserver: OK, ich darf diese Datei ausführen.
   - 2.3 Webserver: Hallo Betriebssystem, bitte diese Datei ausführen.
   - 2.4 Betriebssystem: `a.cgi` ist ein Perl-Skript. Hallo Perl?
   - 2.5 Perl: Ich brauche Daten aus der Datenbank.
   - 2.6 Datenbank: Hallo Perl, hier sind die Daten.
   - 2.7 **Perl: Moment noch, ich generiere jetzt das HTML ...**
   - 2.8 Perl: Hallo Betriebssystem, hier sind die Daten.
   - 2.9 Betriebssystem: Hallo Webserver, hier sind die Daten.
   - 2.10 Webserver: Hallo Browser, im Anhang ist die Datei.

Wie man früher Websites baute

- Man begann mit einer einzelnen HTML-Datei (= Webseite).
- Brauchte man weitere Seiten, kopierte man die erste Seite und passte am Anfang der Datei den Titel an und im mittleren Bereich den Inhalt.
- Um den Überblick nicht zu verlieren, baute man eine Navigation ein. In jede einzelne Seite. Kommt eine Seite hinzu, die in der Navigation auftauchen soll, musste man jede einzelne HTML-Datei ändern.
- Wenn das Design geändert werden sollte, musste man wieder jede einzelne Datei ändern. (Später konnte man das Design in eine CSS-Datei auslagern.)

Wie man mit einem CMS Websites baut

- EinE WebdesignerIn legt (hoffentlich) einmalig Templates und CSS-Dateien an, in denen die Inhaltstypen und deren Aussehen festgelegt werden.
- Webredakteure loggen sich auf der Website ein und legen neue Inhalte an, indem sie Webformulare ausfüllen und in einer Datenbank speichern.

Fast alle CMS liefern ihre Seiten dynamisch aus (Ausnahme: Roxen).

Nachteile von CMS

1. Templates für ein CMS zu bauen, ist eine hochspezialisierte Aufgabe, die meist ein Outsourcing erfordert.
2. Eine Datenbank ist erforderlich. Diese ist manchmal nicht erreichbar.
3. hohe Hardware-Anforderungen
4. langsam
5. bricht unter hoher Last zusammen (skaliert schlecht)
6. Wer das Login einer Webredakteurs hat, kann den Inhalt ändern.
7. Wer Sicherheitslücken in der CMS-Software ausnutzt, kann noch mehr Schaden anrichten. CMS-Plugins sind oft unsicher programmiert.
8. Umstieg auf neuere Versionen der CMS-Software oft schwierig
9. Inhalte kaum portabel, schlecht archivierbar

Neue Entwicklungen

- HTML ist mächtiger geworden (HTML5)
- Javascript hat sich weiterentwickelt
- leichtgewichtige Alternativen zu HTML und XML haben sich etabliert, v.a. Markdown und YAML/JSON
- Es gibt den hochwertigen Konverter pandoc (Markdown $\leftrightarrow$ HTML $\leftrightarrow$ Office $\leftrightarrow$ Wiki-Formate $\leftrightarrow$ EPUB $\leftrightarrow$ LaTeX).
- Es gibt leistungsfähige Generatoren für statische Websites (engl. "static site generators", SSG), z.B. Jekyll, Middleman (Ruby), Hugo (Go), Brunch, Hexo (Javascript), Ikiwiki (Perl), Sculpin (PHP), Cactus, Hyde, MkDocs, Nikola, Pelican, Sphinx (Python).
- Moderne SSGs erlauben Templating (oft Liquid oder Jinja) und erzeugen saubere Metadaten.

Quelle: `https://www.smashingmagazine.com/2015/11/modern-static-website-generators-next-big-thing/`

Vorteile statischer Website-Generatoren

1. Die Nachteile von CMS fallen alle weg.
2. schnell
3. Inhalt ist in Textdateien und darum alterungsbeständig, versionierbar, portabel
4. sicher
5. Server einfach zu konfigurieren
6. Server kann simpel sein (z.B. nginx, lighttpd)
7. weniger Software-Abhängigkeiten auf dem Server
8. lastbeständig und robust
9. flexibel

Quellen:

`https://davidwalsh.name/introduction-static-site-generators`,

`https://www.sitepoint.com/7-reasons-use-static-site-generator/`

Nachteile statischer Website-Generatoren

Durch die neuen Entwicklungen und die zunehmenden Probleme mit dynamischen CMS werden statische Websites wieder attraktiv, aber Einschränkungen bleiben:

1. keine Echtzeit-Inhalte (außer per Javascript im Browser)
2. keine Verwaltung per Browser (Falls wir das brauchen, lässt sich das z.B. mit GitLab einrichten. Roxen kann das!)

Quelle:

`https://davidwalsh.name/introduction-static-site-generators`

Welche statischen Website-Generatoren gibt es?

chique Liste: `https://www.staticgen.com/`

lange Liste (z.Z. 445): `https://staticsitegenerators.net/`

Wer nutzt statische Website-Generatoren?

Jekyll: Fundraising-Website für Barack Obamas Wahlkampf, healthcare.gov, GitHub Pages, GitLab Pages, Amazon (AWS), StackOverflow (Blog), ...

Middleman: Vox Media

Quelle:

`https://davidwalsh.name/introduction-static-site-generators`