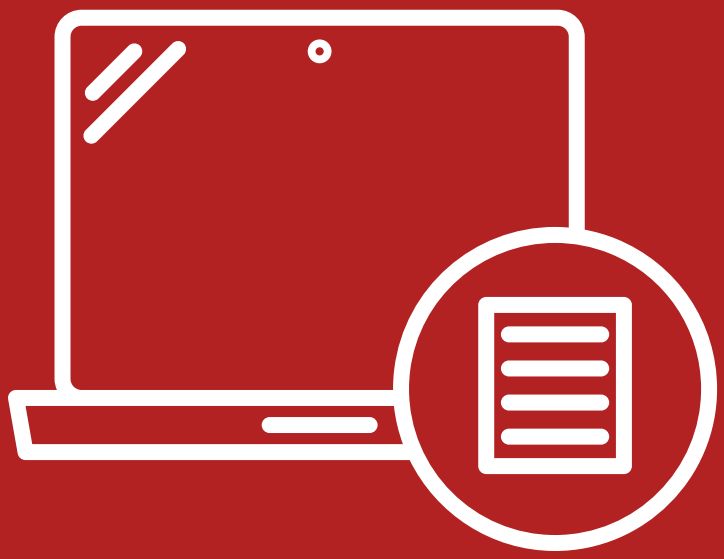
CMSstash

Flat File CMS

Schlanke Webseiten ohne Datenbank erstellen. Mit einer umfassenden Markt-Übersicht und vielen Einzel-Reviews.



Ausgabe 1 / Stand 2022

Weitere CMS-Reports finden Sie auf CMSstash.de

© Sebastian Schürmanns

Auflage 1, Februar 2022
Autor: Sebastian Schürmanns
Anschrift: Sparrstraße 28, 13353 Berlin

Webseite: CMSstash.de
Twitter: twitter.com/CMSstash

Diese Publikation wurde mit dem CMS Typemill erstellt. Unterstützung bei Publishing-Projekten mit Typemill bietet Trendschau Digital.

Inhaltsverzeichnis

1. Auf der Suche nach einem CMS 6
1.1 Wie die Suche gelingt 6
1.2 Wie CMSstash helfen kann 6
1.3 Warum die CMS-Auswahl so wichtig ist 7

2. Flat File CMS im Überblick 9
2.1 Was ist ein Flat-File-CMS? 9
2.2 Vorteile und Nachteile eines Flat-File-CMS 9
2.3 Flat-File-CMS: Eine Übersicht 11
2.4 Generelle Empfehlungen 12
2.5 Kirby: Der Pionier unter den Flat-File-CMS 13
2.6 Statamic: Ein Enterprise Flat-File-CMS 14
2.7 Grav: Das populäre Open Source System 16
2.8 Bludit für Generalisten 17
2.9 Typemill für Publisher 18
2.10 Automad für Ästheten 19
2.11 Yellow für Minimalisten 20
2.12 Pico für Puristen 21
2.13 HTMLy für Blogger 21
2.14 Flextype für Headless-Entwickler 22
2.15 Weitere Optionen 23

3. CMS-Reviews 25
3.1 Ein subjektiver Zugang 25
3.2 Der Aufbau der Reviews 25
3.3 Wie Reviews entstehen 26

4. Automad: Ein Flat-File-CMS für Ästheten 27
4.1 Automad für Autoren 27
4.2 Automad für Entwickler 30
4.3 Automad für Nicht-Entwickler 33
4.4 Kosten 34
4.5 Einsatzsfelder für Automad 34

5. Bludit: Einfaches CMS für schlanke Webseiten 35
5.1 Bludit für Autoren 36

5.2 Bludit für Entwickler 37
5.3 Bludit für Nicht-Entwickler 39
5.4 Kosten 39
5.5 Einsatzfelder für Bludit 40

6. GRAV: Das Open Source Flat-File-CMS 41
6.1 Grav für Content Manager 41
6.2 Grav für Entwickler 43
6.3 Grav für Nicht-Entwickler 46
6.4 Kosten 46
6.5 Einsatzfelder für Grav? 47

7. HTMLy: Ein Flat-File-CMS für Blogger 48
7.1 HTMLy für Autoren 48
7.2 HTMLy für Entwickler 50
7.3 HTMLy für Nicht-Entwickler 52
7.4 Kosten 52
7.5 Einsatzfelder für HTMLy 53

8. Kirby: Der Flat-File-Pionier 54
8.1 Kirby für Autoren 55
8.2 Kirby für Entwickler 57
8.3 Kirby für Nicht-Entwickler 60
8.4 Kosten 60
8.5 Einsatzfelder für Kirby 61

9. Statamic: Flat-File-CMS on Steroids 62
9.1 Statamic für Content Manager 63
9.2 Statamic für Entwickler 65
9.3 Statamic für Nicht-Entwickler 70
9.4 Roadmap 70
9.5 Kosten 70
9.6 Einsatzfelder für Statamic 71

10. Typemill - Ein CMS für Micro-Publisher 73
10.1 Typemill für Autoren 74
10.2 Typemill für Entwickler 76
10.3 Typemill für Nicht-Entwickler 78

10.4 Kosten 79
10.5 Einsatzfelder für Typemill 79

11. Publikationen von CMSstash 80
11.1 Die CMS-Reports 80
11.2 Über den Autor 80

1. Auf der Suche nach einem CMS

Fast alle Menschen kommen täglich mit Web Content Management Systemen (CMS) in Berührung. Die meisten merken davon wenig, weil das Content Management System seine Arbeit unbemerkt im Hintergrund verrichtet, beispielsweise beim Besuch einer Nachrichtenseite. Viele Menschen arbeiten jedoch auch aktiv mit Content Management Systemen, zum Beispiel wenn sie die Corporate Website des Arbeitgebers aktualisieren oder ihre private Webseite pflegen.

Doch auch wenn Content Management Systeme in unserer digitalen Welt allgegenwärtig sind, kommen selbst Profis in Verlegenheit, wenn sie nach einem passenden CMS gefragt werden. Zu groß ist das Angebot an Content Management Systemen und zu vielfältig sind die Szenarien für ihren Einsatz.

1.1 Wie die Suche gelingt

Weil das Angebot an Content Mangement Systemen so unübersichtlich ist und die eigenen Anforderungen zu Beginn meist verschwommen bleiben, wird man im ersten Schritt die Meinung von Freunden, Kollegen, der eigenen Web-Agentur oder der Suchmaschine einholen. Und in vielen Fällen wird einem die Empfehlung bekannt vorkommen: WordPress, Jimdo, Typo3, Drupal, Joomla!, der Adobe Experience Manager oder vielleicht ein Headless CMS, was immer das auch ist.

Man mag sich durch solche pauschalen Empfehlungen bestätigt fühlen. Viel Wert haben sie dennoch nicht. Denn für eine erfolgreiche Suche gibt es zwei Grund-Bedingungen:

- Man sollte die eigenen **Anforderungen** und das geplante Einsatzszenario kennen und im besten Fall präzise formulieren können.
- Man sollte sich zumindest einen groben **Markt-Überblick** verschaffen, um die Suche einzugrenzen und die Empfehlungen einordnen zu können.

1.2 Wie CMSstash helfen kann

CMSstash ist kein Ratgeber im klassischen Sinne und kann – wenn überhaupt – nur indirekt bei der Definition von Anforderungen helfen. Für die

Anforderungs-Definition ist zumindest bei größeren Projekten im Unternehmens-Umfeld oft eine individuelle Beratung sinnvoll.

CMSstash kann jedoch eine Übersicht über den Markt der Content Management Systeme vermitteln, sodass der Leser seine Suche eingrenzen und Empfehlungen besser einordnen kann. Dazu erklärt CMSstash viele gängige CMS-Kategorien und stellt deren wichtigste Vertreter in Kurzform vor. Ergänzend dazu werden ausgewählte Systeme unter den CMS-Reviews ausführlicher besprochen. Derzeit gibt es die folgenden Kategorien:

- **Digital Experience Plattformen (DXP)**: Umfassende Software-Lösungen für große Unternehmen, bei denen das CMS ein Baustein von vielen ist.
- **Enterprise CMS**: Ein CMS für große Unternehmen als Einzel-Lösung.
- **Headless CMS**: Eine neue Form des CMS, bei dem Inhalte lediglich verwaltet und neutral ausgeliefert werden ohne eine konkrete Webseite als Präsentations-Ebene.
- **Hybrid CMS**: Ein CMS, das gleichzeitig als Headless CMS und als traditionelles CMS genutzt werden kann.
- **Flat File CMS**: Ein CMS, das keine klassische Datenbank benötigt und oft für kleinere und schlanke Web-Projekte geeignet ist.
- **Open Source CMS**: Content Management Systeme aller Größenordnungen, die unter einer Open Source Lizenz frei verfügbar sind.
- **Cloud CMS**: Content Management Systeme, die nicht lokal gehostet werden, sondern als Service (SaaS) oder Cloud-Lösungen verfügbar sind.
- **Static Site Generatoren**: Keine CMS im klassischen Sinne, sondern Programme, mit denen sich Text-Dateien (Markdown) in einen statischen Web-Auftritt umwandeln lassen.

Bei der großen Bandbreite an Kategorien und Content Management Systemen sind nicht alle Informationen für jeden Leser relevant. Daher werden einzelne Kategorien mit passenden Reviews auch als kompaktere eBooks publiziert.

1.3 Warum die CMS-Auswahl so wichtig ist

Wozu der ganze Aufwand bei der Auswahl eines CMS, werden einige Fragen? Warum nicht einfach WordPress nehmen wie so viele andere auch? Denn alle großen Content Management Systeme sollten so flexibel sein, dass man nahezu jede Anforderungen damit erfüllen kann. Was nicht

passt, wird eben passend gemacht, daher ist das verwendete CMS eher zweitrangig.

Auch wenn die Argumentation auf den ersten Blick verfängt, verkennt sie einen zentralen Punkt: Der Erfolg eines neuen CMS hängt ganz wesentlich von der Akzeptanz der Anwender ab. Entscheidend ist nicht, ob ein CMS irgendwie funktioniert, sondern wie gut es die Anwender in ihrer Arbeit unterstützt. Nur dann tritt der berühmte Return on Investment ein. Im anderen Fall sind Content Management Systeme eher Hindernisse, die durch umständliche Klickstrecken, durch einen unverhältnismäßigen Ressourcen-Verbrauch oder durch hohe Wartungsaufwände die Produktivität verhindern. Daher zahlt sich die oft mühsame Suche nach einem passenden CMS früher oder später immer aus.

2. Flat File CMS im Überblick

Mit einem Flat File CMS lassen sich vor allem kleine Web-Projekte schnell realisieren. Diese Übersicht stellt die besten Content Management Systeme ohne Datenbank vor und hilft, ein einfaches CMS für eine Webseite zu finden.

2.1 Was ist ein Flat-File-CMS?

Die meisten Content Management Systeme verwenden eine Datenbank, um Inhalte zu speichern. Ein sogenanntes Flat-File-CMS funktioniert anders. Es verzichtet vollständig auf eine Datenbank und speichert die Inhalte und Einstellungen stattdessen in ganz normalen Dateien auf dem Server ab. Für Text-Inhalte haben sich in den letzten Jahren Markdown-Dateien durchgesetzt, doch auch andere Formate sind hin und wieder in Gebrauch.

Trotz des Verzichts auf eine Datenbank zählen Flat-File-CMS zu den vollwertigen Content Management Systemen: Sie verfügen über eine Autoren- und Administrationsoberfläche und erzeugen einen dynamischen Webauftritt beim Besuch einer URL. Anders als die beliebten Static Site Generatoren, die einen statischen Web-Auftritt über Befehle generieren, erfordern Flat-File-Systeme keine besonderen technischen Kenntnisse von den Autoren und Administratoren.

2.2 Vorteile und Nachteile eines Flat-File-CMS

Flat File CMS sind seit einigen Jahren ziemlich en vogue und sie haben viele Vorteile:

- Die meisten Flat-File-Systeme sind **klein**, **schlank** und relativ **einfach**. Dadurch eignen sie sich hervorragend für kleinere Web-Projekte.
- Da Flat-File-Systeme **keine Datenbank** benötigen, kann man mit ihnen auch auf sehr kleinen Hosting-Paketen beliebig viele Web-Auftritte betreiben.
- Die **Installation** eines Flat-File-CMS ist in der Regel sehr einfach. Oft muss nur der Quellcode auf den Server hochgeladen und ein User eingerichtet werden.

- Die **Inhalte** von Flat-File-Systemen können jederzeit in Form von Markdown-Dateien heruntergeladen und anderweitig genutzt werden.
- Für **Backups** und **Server-Umzüge** reicht meist ein simples Kopieren der Dateien.
- Bei vielen Flat-File-CMS ist die **Performance** so gut, dass die Seiten auch ohne Optimierungen hohe Punktzahlen beim Page-Speed erreichen. Außerdem schont der geringe Ressourcen-Verbrauch nicht zuletzt die **Umwelt**, auch wenn in dieser Hinsicht statische Seiten z.B. mit Static Site Generatoren noch deutlich effizienter sind.
- Die **Programmierung** ist bei den meisten Flat-File-CMS aufgrund einer schlanken und vielfach modernen Code-Basis vergleichsweise schnell und einfach.
- Mit der Datenbank fällt auch ein beliebtes **Angriffsziel** für Hacker weg, SQL-Injections sind beispielsweise nicht möglich.

Natürlich haben Flat-File-CMS auch einige Nachteile:

- Die Auszeichnungssprache Markdown wird zwar immer beliebter. Autoren, die noch keine Berührungspunkte mit Markdown hatten, müssen ihre Schreib-Gewohnheiten jedoch umstellen.
- Flat-File-CMS bewegen sich immer noch in einer Nische, daher ist die Auswahl an Themes und Plugins in der Regel begrenzt.
- Man sollte kein zweites WordPress oder einen Website-Builder erwarten. Für die Umsetzung von individuellen Wünschen ist meist ein Entwickler erforderlich.
- Sehr umfangreiche und stark strukturierte Inhalte wie beispielsweise Produkt-Daten funktionieren mit relationalen Datenbanken besser. Bei sehr vielen Inhalten sind auch Performance-Probleme möglich.
- Die Erweiterbarkeit hat zumindest in der Praxis seine Grenzen. Beim Thema eCommerce bieten nur wenige Systeme passende Plugins, beispielsweise Statamic mit Simple Commerce und Kirby mit Erweiterungen für SnipCard oder Shopify.

Bei einem komplexen Projekt im Enterprise-Umfeld würde man nur in seltenen Fällen auf ein Flat-File-CMS zurückgreifen. Umgekehrt ist allerdings auch die gängige Praxis bedenklich, simple One-Pager, Online-Portfolios oder einfache Webseiten mit schwergewichtigeren Systemen wie WordPress zu realisieren. Denn gerade für die große Masse der einfachen Webseiten sind Flat-File-Systeme wie geschaffen. Ein weiteres Argument ist die sehr gute Author Experience einiger Flat-File-Systeme. Nicht ohne

Grund setzt beispielsweise DER SPIEGEL seit einiger Zeit ein Flat-File-CMS als Autoreninterface für seine Webauftritte ein.

2.3 Flat-File-CMS: Eine Übersicht

Der Markt der Flat-File-CMS ist ausgesprochen bunt und vielfältig. Die Komplexität reicht von einem halben Megabyte bis hin zu schwergewichtigen Systemen, die auf große Frameworks setzen. Einige Flat-File-CMS eignen sich gut für die Verwendung im Unternehmens-Umfeld. Andere bieten sich eher für private Webseiten an. Es gibt eher traditionelle Systeme mit simplen Formular-Eingaben, es gibt kleine Page-Builder mit Inline-Editing, es gibt Block-Editoren, HTML-Editoren und Markdown-Editoren und es gibt Systeme mit hochspezialisierten Features für die Nische.

Bei all dieser Vielfalt wirken einige Flat-File-CMS eher wie Experimente mit kurzer Lebensdauer. Daher sollte man bei der Auswahl nicht zuletzt auf die Kontinuität achten.

Name	Start	letzte Version	Kosten	Website	CMSstash Review
Aloia	2019	06.2021	keine	Aloia Website	./.
Automad	2014	11.2021	keine	Automad Website	Automad Review
BatFlat	2017	02.2020	keine	BatFlat Website	./.
Bludit	2015	11.2021	keine	Bludit Website	Bludit Review
CMS Simple_XH	2009	10.2021	keine	CMSimple_XH Website	./.
FlatPress	2007	06.2021	keine	FlatPress Website	./.
Flextype	2018	01.2021	keine	Flextype Website	./.
GetSimple	2009	03.2020	keine	GetSimple CMS	./.
Grav	2014	01.2022	keine	Grav Website	Grav Review
HTMLy	2014	04.2021	keine	HTMLy Website	HTMLy Review
Kirby	2009	02.2022	99 €	Kirby Website	Kirby Review
MoziloCMS	2007	01.2022	keine	Mozilo Website	./.

Pico	2013	08.2020	keine	Pico Website	./.
Reboot CMS	2021	01.2022	keine	Reboot auf GitHub	./.
Saaze	2020	10.2020	keine	Saaze Website	./.
Statamic	2012	08.2021	259 $	Statamic Website	Statamic Review
Typemill	2017	01.2022	keine	Typemill Website	Typemill Review
Typesetter	2009	11.2020	keine	Typesetter Website	./.
WonderCMS	2012	01.2022	keine	WonderCMS Website	./.
Yellow	2013	01.2022	keine	Yellow Website	./.

2.4 Generelle Empfehlungen

Im Dienstleister-Umfeld haben sich vor allem die Systeme Statamic, Kirby und Grav etabliert. **Statamic** und **Kirby** sind kostenpflichtig und bieten dadurch ein hohes Maß an Verlässlichkeit und Stabilität. Das Open Source System **Grav** bietet diese Verlässlichkeit über eine vergleichsweise große Community. Wie bei manch anderem Open Source CMS wird bei Grav die Perspektive der Anwender hin und wieder zugunsten technischer Features vernachlässigt. Bei Kirby und Statamic erhält man dageben eine hervorragende Author Experience inklusive einem modernen Block-Editing. Während Kirby durch den Verzicht auf große Frameworks bis heute sehr schlank geblieben ist, hat sich Statamic vor einigen Jahren für das Laravel-Framework entschieden und kommt entsprechend schwergewichtig daher.

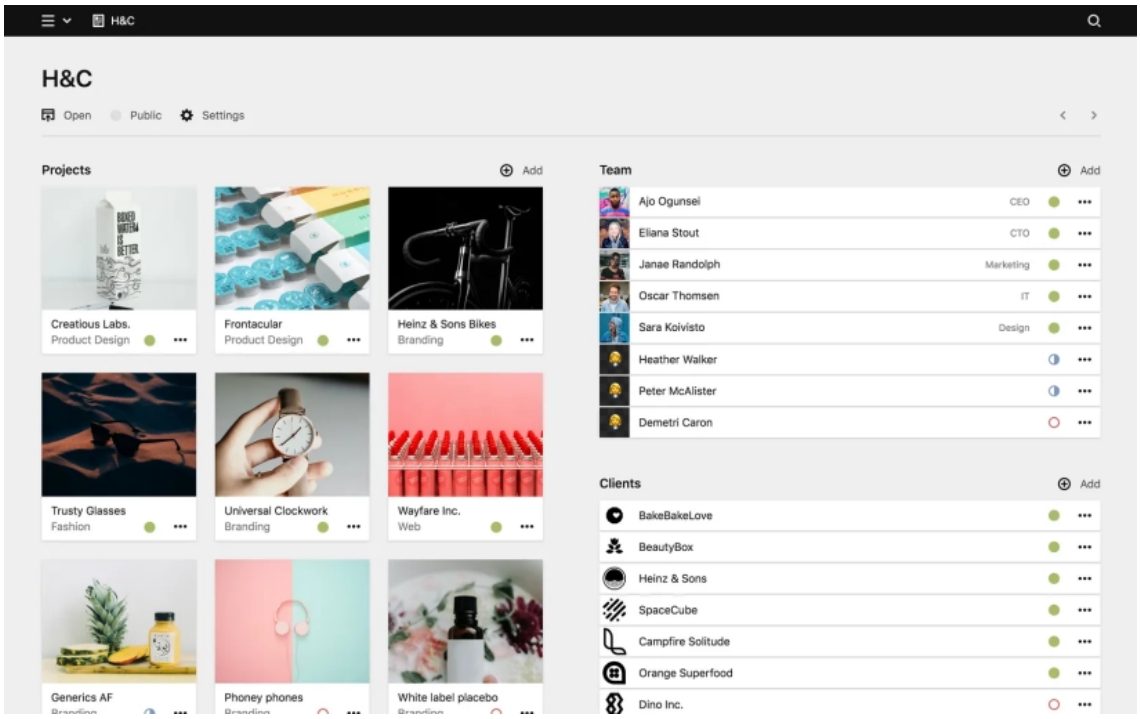
Alle anderen Flat-File-CMS werden von Einzelpersonen oder kleineren Communities betrieben. In diesem Umfeld hat vor allem **Bludit** mit seinem einfachen und direkten Ansatz viele Freunde gefunden. Es eignet sich ebenso für kleine Blogs wie auch für kleine private Webseiten. Ausschließlich für kleine Blogs ist das minimalistische System **HTMLy** entwickelt worden. Nach einigen Jahren des Stillstands ist das Blog-System zuletzt wieder aktualisiert worden. Ob die Wiederbelebung nachhaltig bleibt, ist noch nicht absehbar. Erfolgreich wiederbelebt wurde vor einigen Jahren auch **Automad**. Dabei handelt es sich um eine Art Mini-Page-Builder, mit dem Autoren recht komplexe Designs eigenhändig erstellen können. **Typemill** wiederum legt seinen Schwerpunkt auf informative Webseiten und hat sich auf das Publishing spezialisiert. Mit einem eBook-

Plugin lassen sich aus der Webseite Dokumente und eBooks generieren, sodass sich das System gut für Dokumentationen und Buch-Projekte eignet. Wer im Flat-File-Bereich ein Headless-System sucht, der wird mit **Flextype** fündig. Wie alle Headless-Systeme bietet Flextype eine recht vollständige API an, mit der Frontend-Entwickler eine Webseite erstellen können. Etwas ins Stocken geraten ist inzwischen die Weiterentwicklung von **Pico**, dem minimalistischen Klassiker unter den Flat File CMS. Aktiv entwickelt wird dagegen das minimalistische CMS **Yellow**, das auf eine Administrations-Oberfläche komplett verzichtet und stattdessen die Markdown-Dateien im Browser bearbeiten lässt.

Die genannten Systeme werden in den folgenden Kapiteln kurz charakterisiert und zum Teil in den CMS-Reviews ausführlicher besprochen.

2.5 Kirby: Der Pionier unter den Flat-File-CMS

Die erste Version von Kirby ist im Januar 2012 veröffentlicht worden. Das schlanke System hat damals neue Maßstäbe für Flat File CMS gesetzt. Das blieb auch in der weiteren Entwicklung so. Im November 2015 ist Kirby 2 mit einer komplett neuen Code-Basis veröffentlicht worden. Auch die im Januar 2019 veröffentlichte Version 3 von Kirby legt die Messlatte für andere Systeme hoch: Das Dashboard wurde mit Vue.js erneuert und setzt auf einer vollständigen REST-API auf. Die API eignet sich auch für Single-Page-Applications oder Headless-Konzepte und ermöglicht eine Integration von Static Site Generatoren. Trotz der stetigen Weiterentwicklung ist Kirby jedoch immer schlank, elegant und sehr nutzerfreundlich geblieben. Auf CMSstash gibt es eine eigene Review zu Kirby.



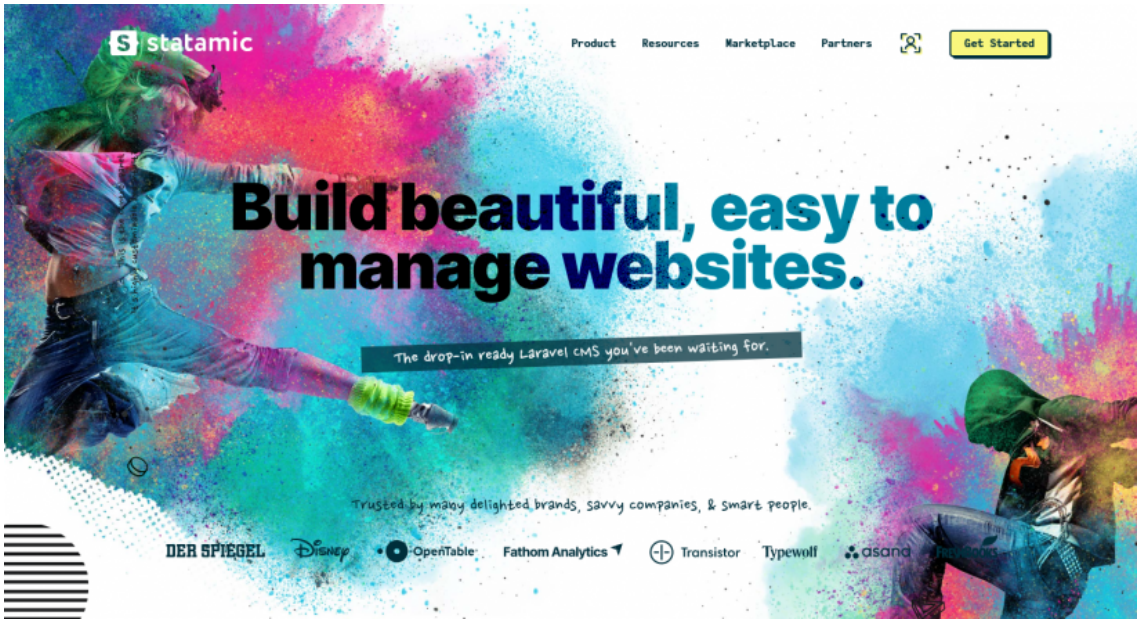
Kirby ist ein kommerzielles System und wird von einem Team um den Gründer Bastian Allgeier entwickelt. Ebenso wie das System ist auch der Preis vergleichsweise stabil. Zwar wurde mit Version 3 der Preis von 79,- Euro auf 99,- Euro angehoben und die günstigere Version für eine private Nutzung aufgegeben. Im Vergleich zur Preisumstellung von Statamic in der Version 2 bleibt Kirby jedoch immer noch sehr moderat.

Besonders angenehm bei Kirby ist die Installation, die praktisch nur aus dem Herunter- bzw. Hochladen der Code-Basis besteht. Die Seite ist anschließend sofort besuchbar, ohne Konfigurationen oder sonstige Anpassungen. Anders als gewohnt nutzt Kirby für Templates reines PHP und verzichtet auf Twig, was der Performance des Systems zugute kommt. Etwas eigen ist der Umstand, dass es keinen separaten Theme-Ordner gibt. Stattdessen ist das Theme mit dem System eng verwoben, ein Theme-Wechsel per Knopfdruck ist nicht vorgesehen.

2.6 Statamic: Ein Enterprise Flat-File-CMS

Statamic kann als das Enterprise System unter den Flat-File-CMS bezeichnet werden. Es richtet sich mit einem entsprechenden Preis vorwiegend an Freelancer und Agenturen. Bekanntester Kunde von Statamic ist der Spiegel Verlag, der im Januar 2020 vollständig auf Statamic umgestiegen

ist, nachdem zuvor bereits Spiegel Plus und Bento mit Statamic liefen. Auf CMSstash gibt es eine Review zu Statamic.

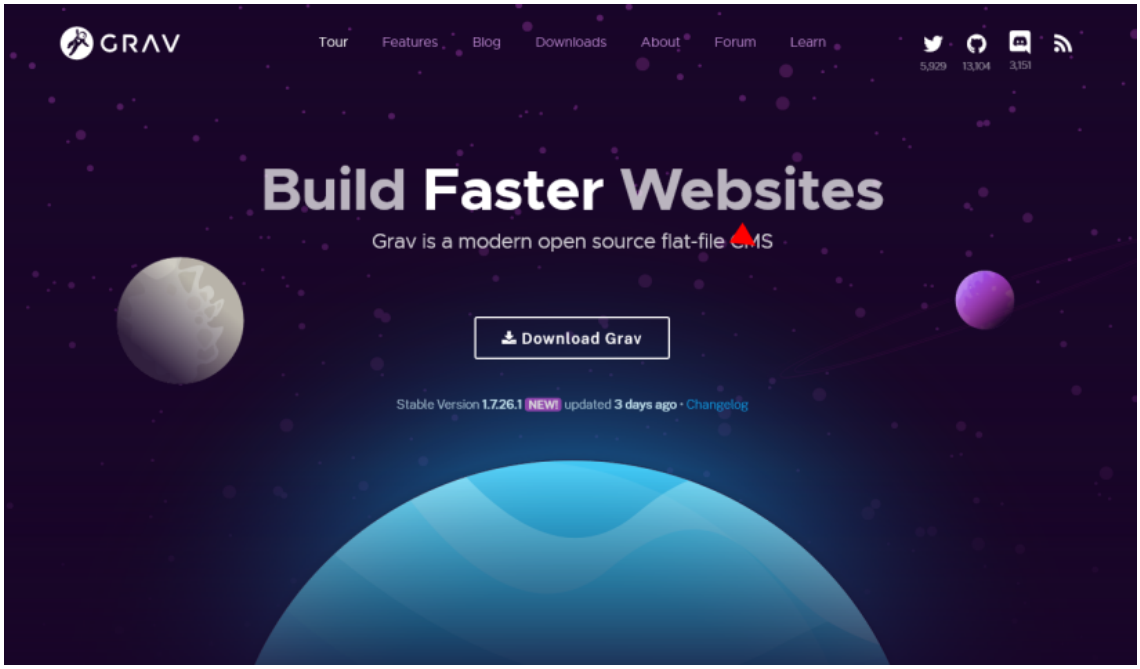


Statamic ist enorm feature-mächtig und dennoch sehr nutzerfreundlich. Die erste Version von Statamic ist im Juni 2012 veröffentlicht worden, also nur wenige Monate nach dem ersten Release von Kirby. Wie Kirby ist auch Statamic sehr beständig, hat allerdings mit der Veröffentlichung von "Statamic 2" Anfang 2016 einen recht weitreichenden Strategie-Wechsel vollzogen und ist technisch vom Micro-Framework Slim auf das deutlich größere PHP-Framework Laravel umgestiegen. Nach über einem Jahr Entwicklungs-Zeit ist im August 2020 Statamic 3 erschienen. Die neue Version hat noch einmal deutlich aufgestockt und bietet so ziemlich alle Features, die man sich auch von einem ausgewachsenen CMS wünschen kann.

Die Neuerungen haben allerdings ihren Preis: Statamic kostet mit der Standard-Lizenz 249 Dollar pro Installation. Statamic überlässt damit Kirby das günstigere Preis-Segment und richtet sich mehr oder weniger ausschließlich an professionelle Dienstleister. Mit Version 3 ist zwar eine kostenfreie Variante hinzugekommen. Die darf allerdings nur für private Webseiten genutzt werden, was bei dem Code-Umfang von Statamic nur in den wenigsten Fällen sinnvoll sein dürfte. Ähnlich wie das beliebte CraftCMS ist Statamic aus dem Umfeld von Expression Engine entstanden. Durch die Ähnlichkeiten bietet sich Statamic als Flat-File-Alternative zum datenbank-basierten CraftCMS an.

2.7 Grav: Das populäre Open Source System

Grav war lange Zeit das einzige Open-Source-CMS im Flat-File-Bereich, das einigermaßen in der Liga von Statamic und Kirby mitspielen konnte. Grav hat bei seinem ersten Release im Jahr 2014 für viel Furore gesorgt. Bis heute ist die Entwickler-Gemeinde sehr aktiv geblieben (Github-Grav), allerdings ist die Dynamik der Produkt-Entwicklung nicht so hoch wie bei Statamic oder Kirby. Initiator von Grav ist der RocketTheme-Chef und Joomla!-Mitgründer Andy Miller. Auf CMSstash gibt es eine ausführliche Review zu Grav.



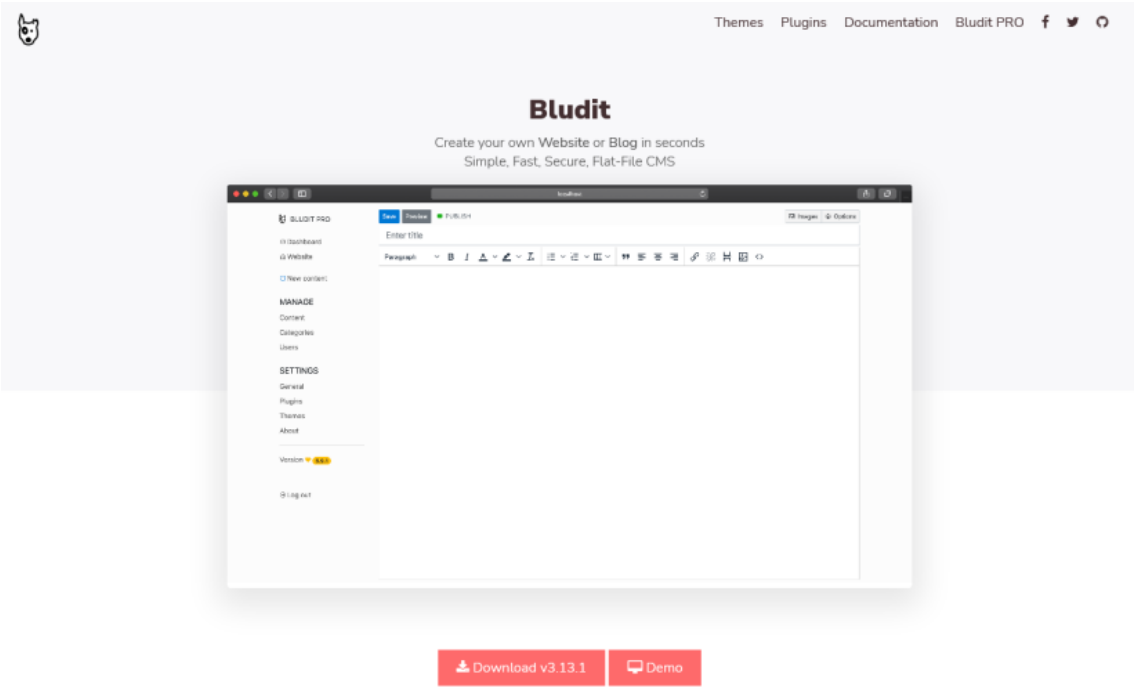
Durch den RocketTheme-Hintergrund bietet Grav eine recht beeindruckende Bandbreite an Themes und Plugins an. Seit Dezember 2020 sind mit Grav Premium einige kostenpflichtige Plugins und Themes hinzugekommen. Doch auch ohne Premium-Features ist der Funktions-Umfang von Grav ausgesprochen groß. Eine Besonderheit von Grav ist das Konzept der modularen Seiten, bei dem sich eine Seite wie beim Lego aus mehreren Unterseiten (Modulen) zusammensetzt. Mit dem Konzept lassen sich zum Beispiel komplexe One-Pager bauen.

Im Vergleich zu Statamic oder Kirby wirkt das Dashboard von Grav allerdings deutlich überkomplex und weniger nutzerfreundlich. Man merkt, dass Grav als Open Source Projekt erst einmal die Bedürfnisse von Entwicklern im Blick hat und weniger auf unerfahrene Autoren achtet. Zwar

wurden inzwischen einige Kinderkrankheiten wie fehlende Sortier-Möglichkeiten für modulare One-Pager behoben. Dennoch sollte man dem Redakteur das System erst einmal vorführen, bevor man sich von den sehr flexiblen Entwickler-Möglichkeiten des Systems überzeugen lässt. Auf der anderen Seite gibt es im Open-Source-Bereich derzeit kein zweites System mit einer vergleichbar großen Community und einer entsprechenden Auswahl an Erweiterungen.

2.8 Bludit für Generalisten

Zunehmend beliebt ist Bludit. Bludit ist ebenfalls Open Source und ein besonders einfaches CMS unter den Flat-File-Systemen. Einfach bedeutet jedoch nicht begrenzt, sondern Bludit ist ganz im Gegenteil extrem flexibel und bietet sogar eine relativ vollständige API als Core-Plugin an. Und das alles bei gerade einmal 4 MB.



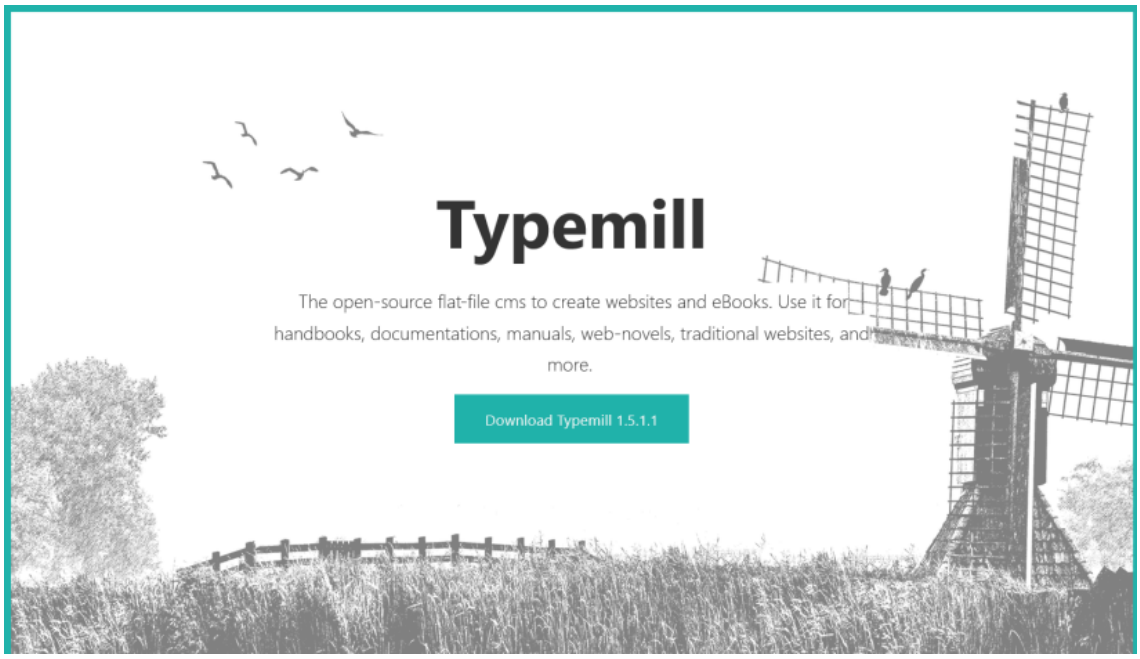
Bludit hat im Jahr 2015 das etwas ältere XML-System Nibbleblog abgelöst. Das schlanke CMS profitiert sichtbar von den Erfahrungen mit dem Alt-System und macht bei der Erst-Installation einen sehr guten Eindruck: Es nimmt den Anwender genauso an die Hand, wie man es sich wünscht und macht den Installations-Prozess zu einem Kinderspiel.

Auch die Admin-Oberfläche ist aufgeräumt und das Bild-Management komfortabel. Eine kleine Auswahl an Themes und Plugins machen das System für einfache Projekte komplett, und mit der API können sogar moderne frontend-getriebene Webseiten realisiert werden. Nach einer kurzen Umstellung auf Markdown ist Bludit wieder auf den Standard-Editor TinyMCE umgestiegen, Autoren können bei Bludit also mit einem gewohnten HTML-Editor arbeiten.

Bludit wurde seit 2015 konstant weiterentwickelt, sodass das System für einfache Seiten absolut empfehlenswert ist, auch als Alternative zum etwas größeren und bekannteren Open-Source-CMS Grav. Derzeit ist eine neue Version 4 von Bludit in Arbeit.

2.9 Typemill für Publisher

Typemill ist mein eigener Beitrag zur Flat-File-Welt. Das System richtet sich vor allem an Autoren und Micro-Publisher, wird aber auch von zahlreichen Unternehmen beispielsweise für Dokumentationen genutzt. Für Publisher bietet das CMS ein eigenes eBook-Plugin an, mit dem sich aus einer Typemill-Webseite ein oder mehrere eBooks im PDF- und EPUB-Format generieren lassen.

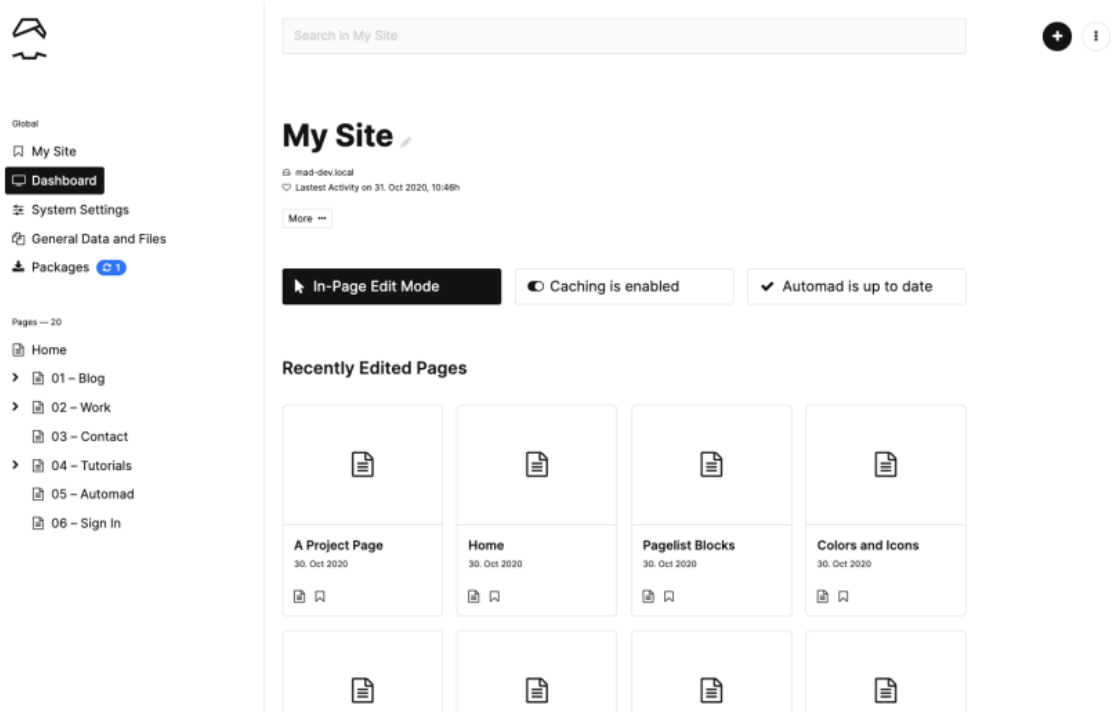


Zu den weiteren Besonderheiten von Typemill zählt ein block-basierter Visual-Editor, mit dem sich Markdown-Inhalte in einer Art WYSIWYG-Stil

erstellen lassen. Zum Herzstück des Systems gehört außerdem ein Formular-Manager, der für Plugin-Einstellungen, für Theme-Einstellungen, für öffentliche Formulare und für zusätzliche Seiten-Informationen genutzt werden kann. In jüngster Zeit sind Shortcodes hinzugekommen, mit denen beliebige Inhalte und Features über entsprechende Plugins integriert werden können. Typemill bietet diverse kostenfreie Plugins, Themes und eBook-Layouts an. Das CMS nutzt das Micro-Framework Slim (PHP) und das Frontend-Framework Vue.js.

2.10 Automad für Ästheten

Das Flat-File-CMS Automad stammt aus dem Jahr 2013 und ist anschließend in einen längeren Dornröschen-Schlaf versunken. Im Juli 2018 ist Automad dann als sehr moderne, frische und aufgeräumte Version aus dem Schlaf erwacht und wird seitdem regelmäßig von seinem Gründer Marc Anton Dahmen weiterentwickelt.



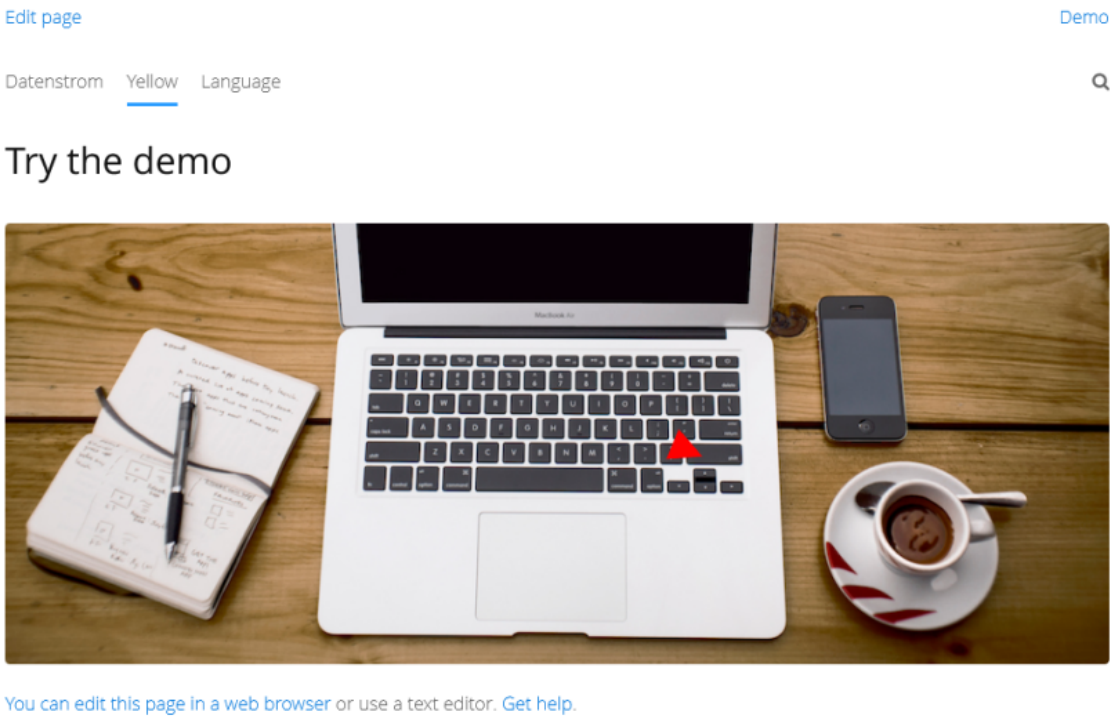
Automad bietet den Autoren mit seinem Block-Editor sehr große Freiheiten, wenn es um die Erstellung anspruchsvoller Designs geht. Bei der Bearbeitung von komplexen Gestaltungs-Elementen kann man als Autor zwar die Übersicht verlieren. Doch es gibt kaum ein anderes Flat-File-CMS, das so viel Gestaltungs-Spielraum ohne die Unterstützung durch ei-

nen Entwickler ermöglicht. Eine Option ist Automad daher vor allem für design-lastige Webseiten, wie man sie oft in der Kreativ-Branche antrifft.

2.11 Yellow für Minimalisten

Unter den vielen anderen Flat-File-Varianten sorgt vor allem Yellow für Abwechslung. Yellow verzichtet auf eine Autoren- oder Administrations-Oberfläche. Man kann sich jedoch einloggen und anschließend alle Seiten über einen kleinen aufpoppenden Markdown-Editor bearbeiten.

Der Ansatz ist extrem minimalistisch, erfordert allerdings von den Autoren auch eine gewisse Affinität zur Technik, da neben den Markdown-Inhalten auch die Seitenkonfigurationen direkt im sogenannten Frontmatter der Markdown-Dateien bearbeitet werden.



Yellow wird bis heute von dem schwedischen Entwickler Mark Seuffert regelmäßig weiterentwickelt. Es bietet eine sehr gute Dokumentation an und verfügt über einen erstaunlichen Feature-Umfang inklusive Shortcodes und einer umfangreichen und vor allem sehr geradlinigen Entwickler-API. Bei GitHub hat das kleine CMS inzwischen über 300 Sterne eingesammelt und gehört damit schon zu den bekannteren Systemen.

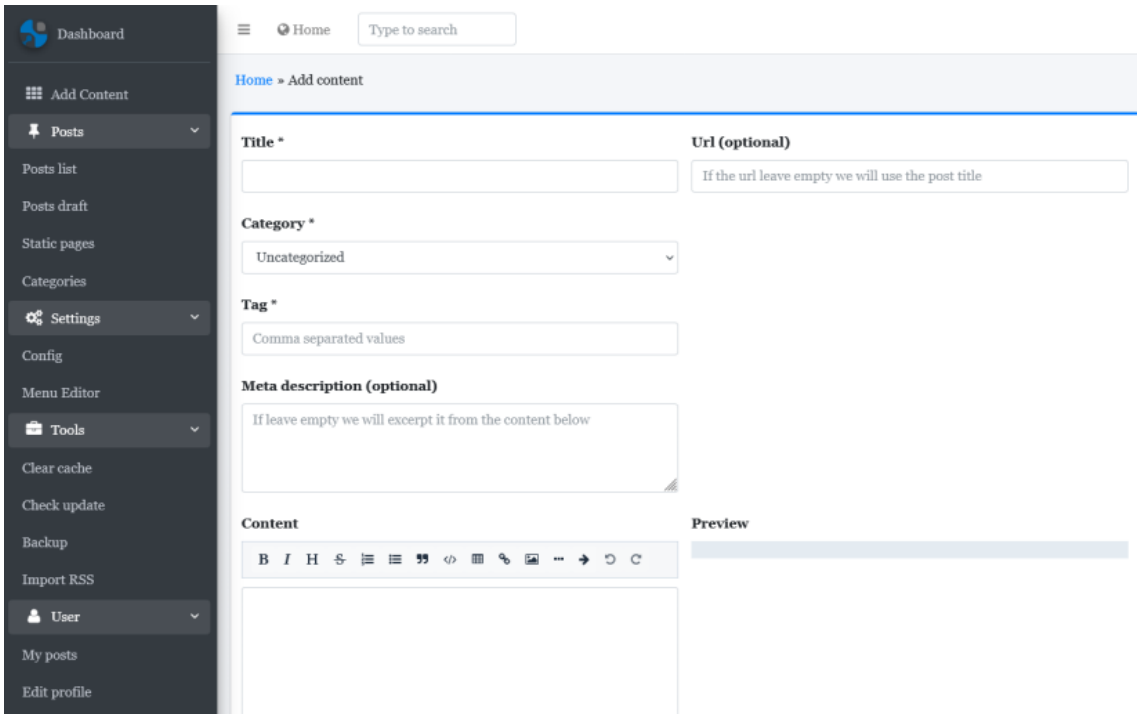
2.12 Pico für Puristen

Pico hat eine lange Tradition und ist vor allem unter Puristen sehr beliebt. Das Micro-CMS ist schlank, nicht abhängig von Frameworks oder anderen Bibliotheken und bietet eine recht komfortable Anzahl an Erweiterungen an.

Pico ist von dem schottischen Entwickler Gilbert Pellegrom Mitte der 2000er Jahre ins Leben gerufen worden, dürfte also eines der ältesten noch aktiven Flat File CMS sein. Pellegrom hat erst im Jahr 2020 mit Saaze ein neues Flat File CMS vorgestellt, das bislang jedoch nicht viel Beachtung gefunden hat. Inzwischen ist Daniel Rudolf der Lead-Entwickler von Pico. Die Entwicklung von Version 3 ist etwas ins Stocken geraten, es gibt jedoch ein Pre-Release auf GitHub.

2.13 HTMLy für Blogger

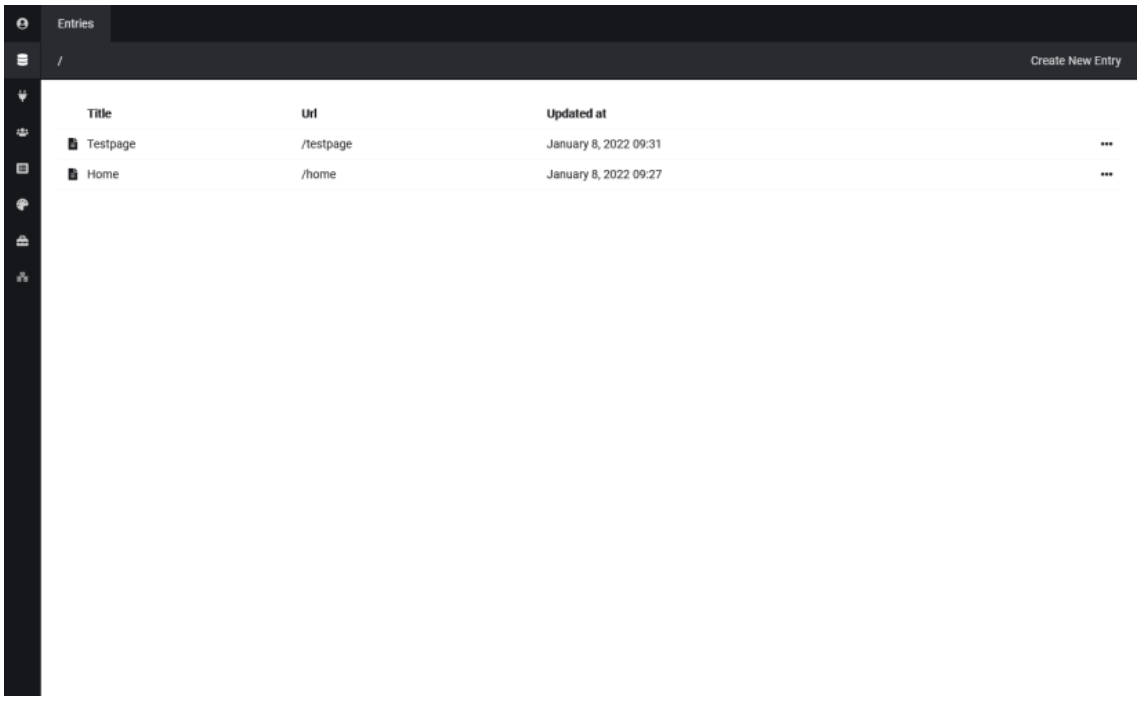
Wer nach einem Flat-File-CMS für Blogs Ausschau hält, der wird möglicherweise mit HTMLy fündig. Das Projekt wurde im Jahr 2013 von dem Entwickler Danang Probo Sayekti explizit als Alternative zu WordPress vorgestellt. Von 2017 bis 2020 hat HTMLy allerdings eine Pause eingelegt. Die Wiederbelebung aus dem Jahr 2020 hat bis heute Bestand, sodass man Projekt wieder in die Auswahl nehmen kann.



HTMLy verfügt über alle notwendigen Features für den Betrieb eines einfachen Blogs. Es gibt verschiedene Content-Typen für Posts nach dem Muster von Tumblr, außerdem gibt es Related Posts, RSS-Feeds und die Möglichkeit, externe Kommentar-Systeme einzubinden. Die Theme-Auswahl ist mit etwa ein Dutzend Varianten zwar beschränkt. Wenn jedoch nur ein simpler Blog geplant ist, dürfte für jeden Geschmack etwas dabei sein. Entwickler und vor allem Code-Einsteiger werden sich über die sehr schlank gehaltene Code-Basis im prozeduralen Stil freuen, das hat auch unter Flat-File-Systemen heute Seltenheitswert und stellt eine schöne Abwechslung dar.

2.14 Flextype für Headless-Entwickler

Flextype ist als Nachfolgesystem des XML-basierten CMS Monstra entstanden. Im Laufe der Entwicklung ist Flextype auf den Headless-Trend aufgesprungen und bezeichnet sich heute als Hybrid-CMS. Die nächste Version 0.9.17 ist derzeit in Arbeit.



Der Headless-Philosophie entsprechend basiert das gesamte System auf einer API, über die sich die Inhalte verwalten und ausliefern lassen. Für die Nutzung der API generiert das System klassische Token. Zusätzlich kann ein Token für die Authentifizierung erstellt werden, sodass Inhalte auch über externe Systeme nutzbar sind.

Flextype bietet derzeit zwei Varianten zum Download an: Das reine Core-System für die Headless-Nutzung und eine erweiterte Version mit Admin-Plugin und einem Demo-Theme. Mit der erweiterten Version (Flextype CMS) können auch normale Anwender eine einfache Webseite betreiben. Die Auswahl an verfügbaren Themes allerdings sehr klein, da normale Anwender nicht zur Kern-Zielgruppe von Flextype gehören. Kern-Zielgruppe sind Entwickler sein, die ein kleines Headless-CMS ohne Datenbank suchen.

2.15 Weitere Optionen

Bis heute entstehen immer wieder neue Flat-File-Systeme. Ein Beispiel ist das genannte CMS "Saaze" aus dem Jahr 2020. Nur ein Jahr zuvor ist Aiola entstanden, ein Flat-File CMS für das beliebte PHP-Framework Laravel. Beide Systeme haben jedoch bislang kaum Aufmerksamkeit erlangt und die Weiterentwicklung wirkt nicht sehr dynamisch.

Daneben gibt es noch weitere Systeme, die einen Blick wert sind. Zum Beispiel Batflat. Das CMS wirkt vergleichsweise modern, konnte sich jedoch als kostenpflichtiges System nicht durchgesetzt und ist daher vor einiger Zeit auf eine Open-Source-Lizenz umgestiegen. Die letzte Aktualisierung ist allerdings schon einige Zeit her. Erst vor kurzem wurde dagegen WonderCMS aktualisiert. Das System ist ebenfalls vergleichsweise modern und bietet einen Inline-Editing-Modus an.

Auch Systeme wie Mozillo, GetSimple, CMS Simple XH, FlatPress und Typesetter sind eine Option. Die Systeme sind zwar schon etwas in die Jahre gekommen und die Entwicklungs-Dynamik wirkt nicht mehr so hoch. Dafür können die Systeme jedoch eine lange Entwicklungs-Geschichte vorweisen und sind entsprechend erprobt.

3. CMS-Reviews

CMSstash bietet zahlreiche Reviews zu ausgewählten Content Management Systemen an. Content Management Systeme gehören zu den vergleichsweise komplexen und sehr variantenreichen Technologien. Zwar gibt es einige Standard-Funktionalitäten. Entscheidend sind jedoch die unterschiedlichen Ausprägungen, die ein CMS charakterisieren.

3.1 Ein subjektiver Zugang

Die hohe Individualität von Content Management Systemen macht einen objektiven Vergleich schwer. Eine Möglichkeit besteht in der Zusammenstellung großer Feature-Matrixen. Der Aufwand für solche Matrixen ist allerdings sehr hoch und die Aussagekraft vergleichsweise begrenzt. Denn das Vorhandensein eines Features sagt nur wenig über dessen Qualität aus. Die Matrix verrät außerdem nur wenig über die User Experience des Gesamt-Systems und die Eignung für ein individuelles Szenario. Feature-Matrixen werden daher nur noch für ein erstes Ausschluss-Verfahren nach geeigneten K.O.-Kriterien empfohlen.

Eine andere Möglichkeit bei der CMS-Auswahl ist der subjektive Zugang. Dabei kann man die eigenen Anforderungen und Arbeits-Workflows in kurzen User-Stories formulieren und anschließend prüfen, wie diese User-Stories in den verschiedenen Systemen abgebildet werden können. Dieser Ansatz passt zu der allgemeinen Weisheit, dass es nicht das "beste CMS" für alle gibt, sondern nur ein passendes CMS für eine individuelle Situation.

Diesen subjektiven Ansatz verfolgt auch CMSstash bei seinen Einzel-Reviews. Es geht nicht um eine erschöpfende und objektive Darstellung. Vielmehr geht es darum, stellvertretend für den Leser einen ersten subjektiven Eindruck von dem System zu vermitteln und dem Leser bei der Entscheidung zu helfen, ob eine tiefere Beschäftigung mit dem CMS sinnvoll ist.

3.2 Der Aufbau der Reviews

Um trotz des subjektiven Ansatzes eine gewisse Vergleichbarkeit herzustellen, folgen alle Reviews einem ähnlichen Aufbau:

- **Ein kurzes Intro** mit Informationen zur CMS-Kategorie, zur Geschichte und zum Hersteller.
- **Die Author Experience** mit kurzen Informationen zur Autoren- und Administrations-Oberfläche. Als Schlaglicht wird häufig das Bild-Management getestet, da sich in diesem Detail oft große Unterschiede in der Author Experience offenbaren.
- **Die Entwickler-Perspektive** mit einigen Schlaglichtern zur verwendeten Technologie, zum Installationsprozess und zur Template-Entwicklung.
- **Nicht-Entwickler** sollen in einem eigenen Absatz Anhaltspunkte bekommen, ob sich ein CMS mit keinen oder nur geringen Code-Kenntnissen für den Aufbau eines Web-Auftritts nutzen lässt.
- **Die Kosten** entstehen nicht nur durch eine Lizenz, sondern auch durch die Implementierung und die Anpassung des Systems. Wenn Kosten nicht ausgewiesen sind, gibt es eine grobe Einordnung.
- **Die Einsatzszenarien** mit einer subjektiven Einschätzung, für welche Projekte sich ein CMS besonders gut eignen könnte.

3.3 Wie Reviews entstehen

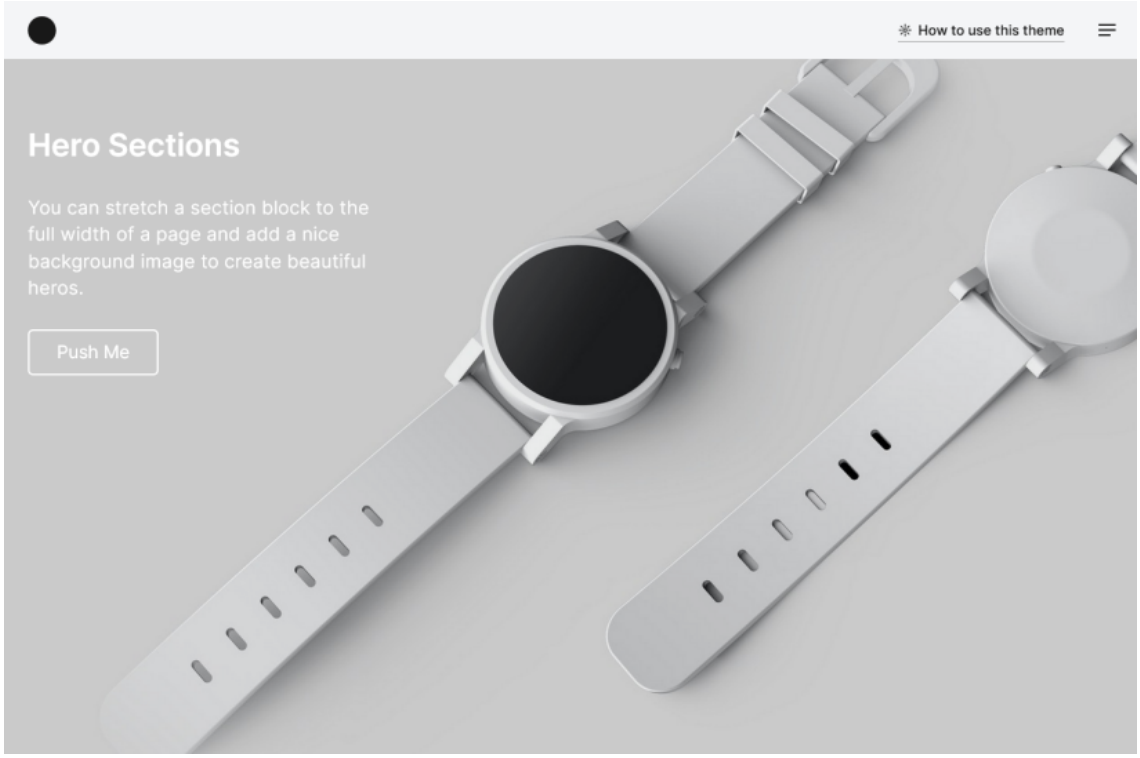
Die Reviews entstehen grundsätzlich durch eigene Recherche und – soweit die Möglichkeit besteht – durch eigene Tests. Beim Vorgehen gibt es eine klare Priorität:

1. Das CMS wird lokal installiert und getestet.
2. Das CMS wird mit einer öffentlichen Demo-Installation getestet.
3. Das CMS wird nicht getestet und die Review basiert auf allgemeinem Informations-Material.

Gerade bei den großen und kommerziellen Enterprise-Systemen sind eigene Tests oft nicht möglich. Die Reviews bieten jedoch immer noch Anhaltspunkte zur allgemeinen Ausrichtung des CMS. Je komplexer die Systeme und die Einsatz-Szenarien werden, desto mehr empfiehlt sich eine individuelle Begleitung des Auswahlprozesses durch eine neutrale Beratung.

4. Automad: Ein Flat-File-CMS für Ästheten

Das Flat-File-CMS Automad legt einen besonderen Fokus auf ein hochwertiges Design. Die Handhabung ist für Autoren etwas anspruchsvoller. Dafür bietet das kleine System eine erstaunliche Gestaltungsfreiheit und ermöglicht komplexe Layouts, wie man sie vor allem aus der Kreativ-Szene kennt.



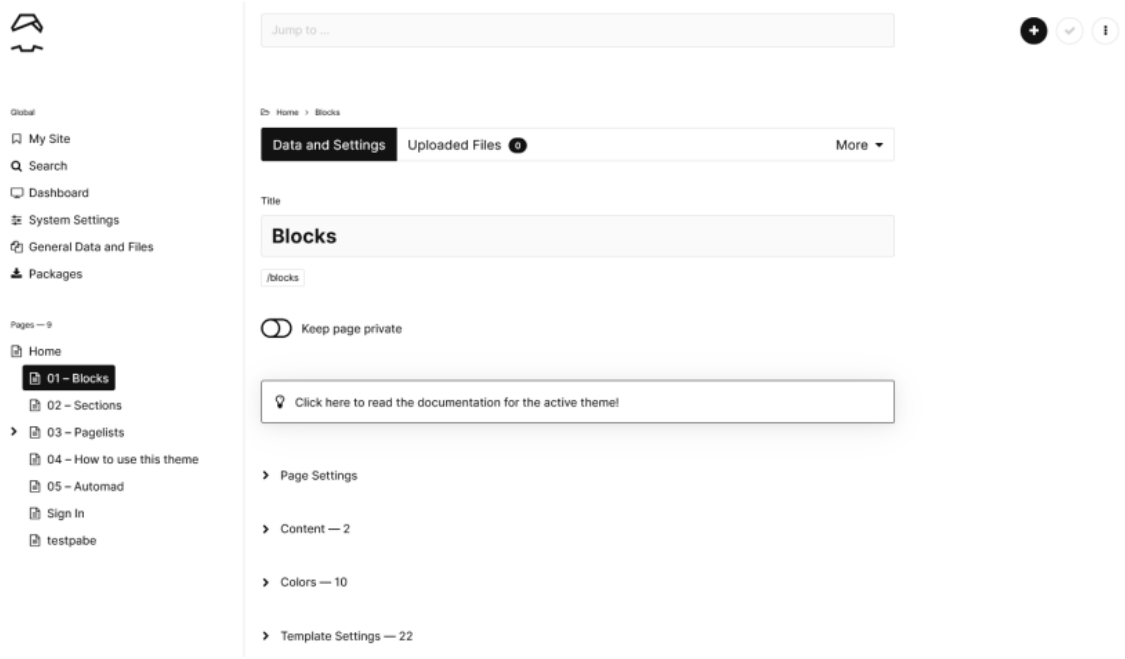
Automad wird seit 2013 von Marc Anton Dahmen entwickelt. Nach einer längeren Pause ist im Jahr 2018 eine rundum erneuerte Version von Automad veröffentlicht worden. Seitdem wird das System kontinuierlich weiterentwickelt.

4.1 Automad für Autoren

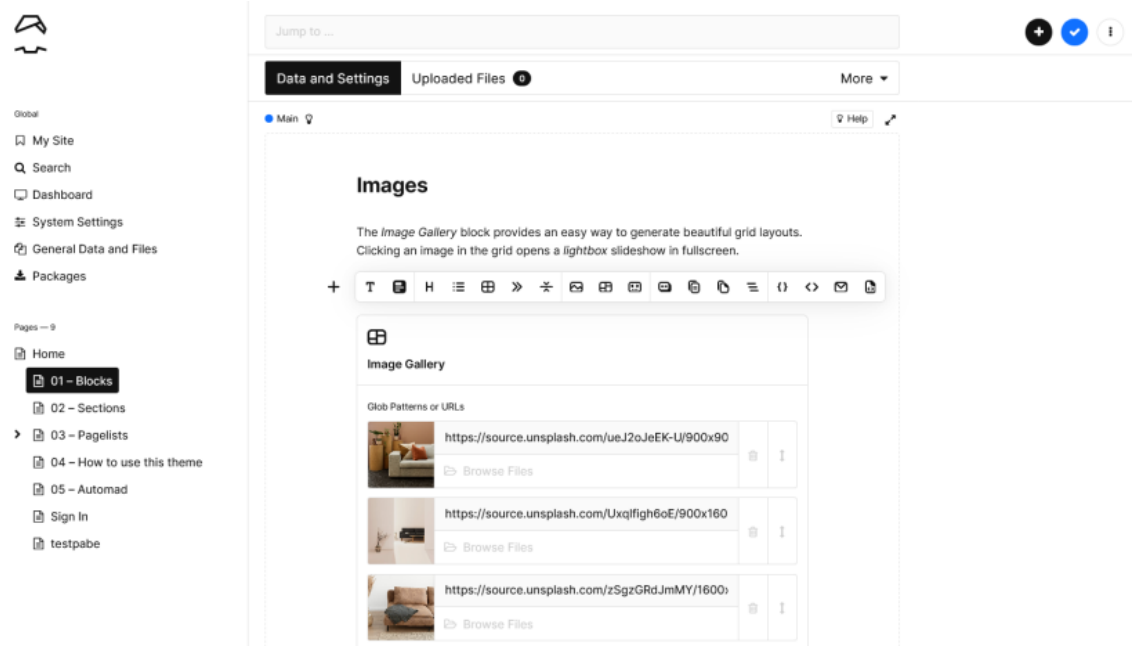
Automad bietet den Autoren eine hohe Flexibilität bei der Gestaltung von Webseiten. Selbst komplexe Layouts mit verschiedenen Abschnitten, Spalten und verschachtelten Elementen sind mit dem kleinen CMS möglich.

Das Dashboard präsentiert sich im reduzierten Schwarz-Weiß-Look. Die Gewichtung der einzelnen Elemente ist etwas eigenwillig: Im Haupt-Tab "Data and Settings" werden die Einstellungen, Konfigurationen und die In-

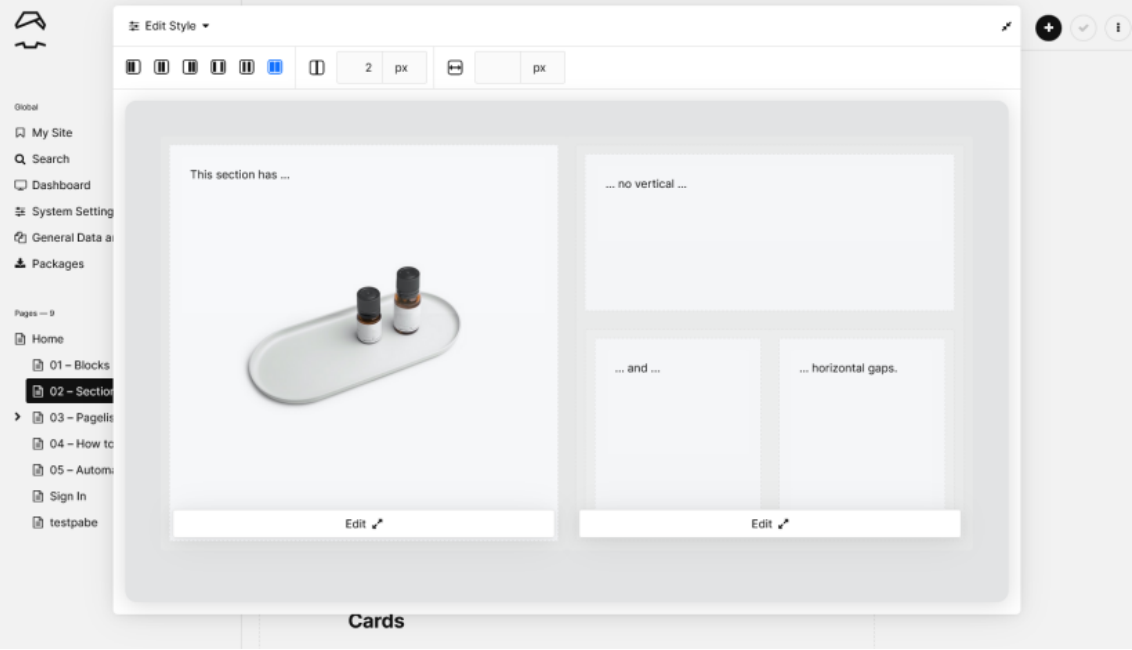
halte gleichwertig im Akkordeon-Stil dargestellt. Im zweiten Tab "Uploads" können Bilder und Dateien zu einer Seite hochgeladen werden.



Im Content-Bereich arbeitet der Autor bei Automad mit Blöcken. Der Block-Editor ist recht intuitiv und die Block-Auswahl bietet bereits viel Spielraum: Neben Überschriften, Listen oder Tabellen gibt es zum Beispiel Bilder, Bild-Gallerien, Slideshows, Buttons, Pagelists und sogar ein E-Mail-Formular. Alle Elemente werden visuell dargestellt und lassen sich ohne Kenntnisse einer Auszeichnungssprache wie Markdown bearbeiten. Bei einigen Elementen sind noch zusätzliche Layout-Varianten wie zum Beispiel die Text-Ausrichtung wählbar.



Noch mehr Flexibilität bieten die sogenannten Sections. Sections können als Design-Rahmen für Content-Blöcke verstanden werden. Sie lassen sich über ein Icon in der Edit-Bar einfügen. In eine Section können sowohl weitere Sections, als auch Content-Blöcke positioniert und verschachtelt werden. Mit solchen Sections lassen sich beispielsweise Spalten im Card-Layout oder auch Action-Buttons vor Bildhintergründen erstellen.



Der Block-Editor von Automad wirkt ausgesprochen intuitiv. Wer komplexe Layouts mit Sections erstellen will, der wird etwas Zeit für die Ein-

arbeitung benötigen. Neben der generellen Funktionsweise des Section-Editors muss man sich zusätzlich mit den vielen Einstellungsmöglichkeiten vertraut machen. Hinzu kommt eine mitunter sehr tiefe Verschachtelung komplexer Layout-Elemente, sodass man die eigentlichen Inhalte oft erst im dritten, vierten oder fünften Overlay bearbeiten kann. Hilfreich ist dabei der WYSIWYG-Ansatz der Editoren und die Option, die Inhalte in der fertigen Seitenvorschau über einen In-Page-Edit-Mode zu bearbeiten.

Den Spagat zwischen der gestalterischen Freiheit und einer komplexen Bedienung müssen alle Systeme meistern, die ein Page-Builder-Konzept verfolgen. Beispiele dafür sind WordPress, Webflow oder auch High-End-Lösungen wie der Adobe Experience Manager. Bei Automad lassen sich schon mit dem Block-Editor recht intuitiv hochwertige Designs erstellen. Wer die volle Flexibilität von Automad mit den Sections ausnutzen will, wird sicher etwas zu kämpfen haben. Es gibt jedoch unter den Flat-File-CMS kein anderes System, das einem Autoren ohne die Unterstützung durch einen Entwickler derartige Gestaltungsspielräume bietet.

4.2 Automad für Entwickler

Automad läuft mit PHP ab Version 7.4 und unterstützt die Server Apache und Nginx. Im Backend verzichtet das System auf ein separates Framework und nutzt mit Urlify und PHP-Markdown nur zwei externe Bibliotheken. Im Frontend bzw. für die Autorenoberfläche setzt Automad auf JQuery, UI-Kit und Editor.js.

Automad lässt recht einfach manuell, per Composer oder mit Docker installieren. Wie bei vielen Flat-File-Systemen sind keine großen Konfigurationen nötig, sondern man entpackt die Dateien, navigiert zur Seite und erstellt über ein Formular einen initialen User. Anders als üblich wird bei Automad nicht direkt ein User angelegt, sondern das System generiert eine PHP-Datei mit den Account-Details, die man anschließend in den Config-Ordner ablegen muss. Bei einer Installation über Composer erfolgt dieser Schritt automatisch.

Automad geht in vieler Hinsicht eigene Wege. Das fängt bereits bei der **Speicherung von Inhalten** an. Automad nutzt dazu `txt`-Dateien. Darin findet man die Konfigurationen als einfache Werte-Paare. Die Inhalte des Block-Editors werden im JSON-Format mit den Präfixen `+hero:` und `+main:` abgespeichert. Das JSON-Format stammt von dem recht neuen

und beliebten Script editor.js, auf dem das gesamte Editor-Konzept von Automad beruht. Das JSON-Format bietet natürlich eine enorme Flexibilität bei der Content-Erstellung und muss nicht mit den Beschränkungen des Markdown-Formats leben. Allerdings stellt das JSON-Format auch eine Hürde für Nicht-Entwickler dar, denn eine Bearbeitung der Inhalte außerhalb von Automad ist kaum möglich.

Einen eigenen Weg verfolgt Automad auch mit einer **eigenen Sprache für Templates**. Während sich Statamic mit seiner Template-Sprache Antler stark an Twig orientiert, weicht Automad etwas mehr von den verbreiteten Vorbildern ab. Das verkürzte Template einer Pageslist kann wie folgt aussehen:

```
<@ elements/header.php @>
    <@ elements/navbar.php @>
    <section class="section">
        <div class="container">
            <div class="columns is-8 is-variable">
                <main class="column is-three-quarters content">
                    <h1 class="am-block title">
                        @{ title }
                    </h1>
                    <p class="am-block is-size-5">
                        Building a portfolio or blog page is a very common task.
                    </p>
                    <@ newPagelist {
                        filter: @{ ?filter },
                        match: '{":level": "/(1|2)/"}',
                        search: @{ ?search },
                        sort: @{ ?sort | def ('date desc') }
                    } @>
                    <div class="columns is-multiline is-variable">
                        <@ foreach in pagelist @>
                            <div class="column is-one-third is-flex">
                                <div class="card is-flex-grow-1">
                                    <div class="card-content">
                                        <div class="content">
                                            <h3>@{ title }</h3>
```

```
@{ date | dateFormat ('F Y') }
                                                    </span>
                                                    <p class="mt-3 mb-5">
                                                        @{ +main | findFirstParagraph | 150
}
                                                    </p>
                                                    <a href="@{ url }" class="button">More
...</a>
                                            </div>
                                        </div>
                                    </div>
                                </div>
                            <@ end @>
                        </div>
                    </main>
                    <nav class="column is-one-quater">
                        <@ elements/menu.php @>
                    </nav>
                </div>
            </div>
        </section>
<@ elements/footer.php @>
```

Die Template-Sprache bietet Statements und Content-Variablen. Content-Variablen ähneln dem Twig-Muster einer Variabel mit folgenden Filtern, die durch Pipes getrennt werden. Allerdings werden die Variablen mit einem at-Zeichen eingeleitet:

- `@{ variable|filter }`
- `@{ date | dateFormat ('F Y') }`

Bei den Statements gibt es fünf Varianten:

- snippet definitions
- includes
- function calls
- extension methods
- control structures

Die Varianten für Statements sind relativ selbsterklärend und sehen im Code wie folgt aus:

```
<@ snippet @>
<@ file.php @>
<@ function { key: 'Value' } @>
<@ namespace/name { options } @>
<@ foreach in pagelist @>
    ...
<@ end @>
```

Im Gegensatz zu den meisten anderen Flat-File-Systemen bietet Automad kein Event-System an, mit dem üblicherweise Plugins gebaut werden. Es ist jedoch möglich, das System über die Extension Methods zu erweitern. Beispiele für Extensions findet man bei den Packages von Automad.

Zusätzlich kann man für Automad einen "Headless-Mode" aktivieren. Nach Aktivierung werden im Frontend beim Aufruf einer URL einige Angaben im JSON-Format zurückgegeben, beispielsweise Titel, Meta-Angaben oder Teile der Navigationsstruktur mit Children und Siblings. Eine Authentifizierung ist nicht vorgesehen und die Dokumentation zu dem Headless-Mode ist insgesamt etwas knapp gehalten.

Wer in die Template-Entwicklung von Automad einsteigen will, der findet neben den Informationen in der Dokumentation einen ausführlichen Gast-Beitrag von Marc-Anton Dahmen auf CSS-Tricks.

4.3 Automad für Nicht-Entwickler

Mit seinem Page-Builder-Ansatz und den sehr flexiblen Layout-Optionen gehört Automad zu den ganz wenigen Flat-File-CMS, die den Autoren auch ohne Rückgriff auf einen Entwicklern eine sehr hohe Gestaltungsfreiheit bieten. Natürlich bewegen sich Autoren immer noch im Rahmen eines Themes. Innerhalb des Themes ermöglichen die vielen Optionen jedoch eine sehr weitgehende Individualisierung und Gestaltung der Webseite.

Für Entwickler oder Einsteiger ist Automad vergleichsweise einfach zu erlernen. Man arbeitet allerdings mit recht eigenen Konzepten, die sich na-

türlich bis zu einem gewissen Grad auf andere Systeme abstrahieren lassen.

4.4 Kosten

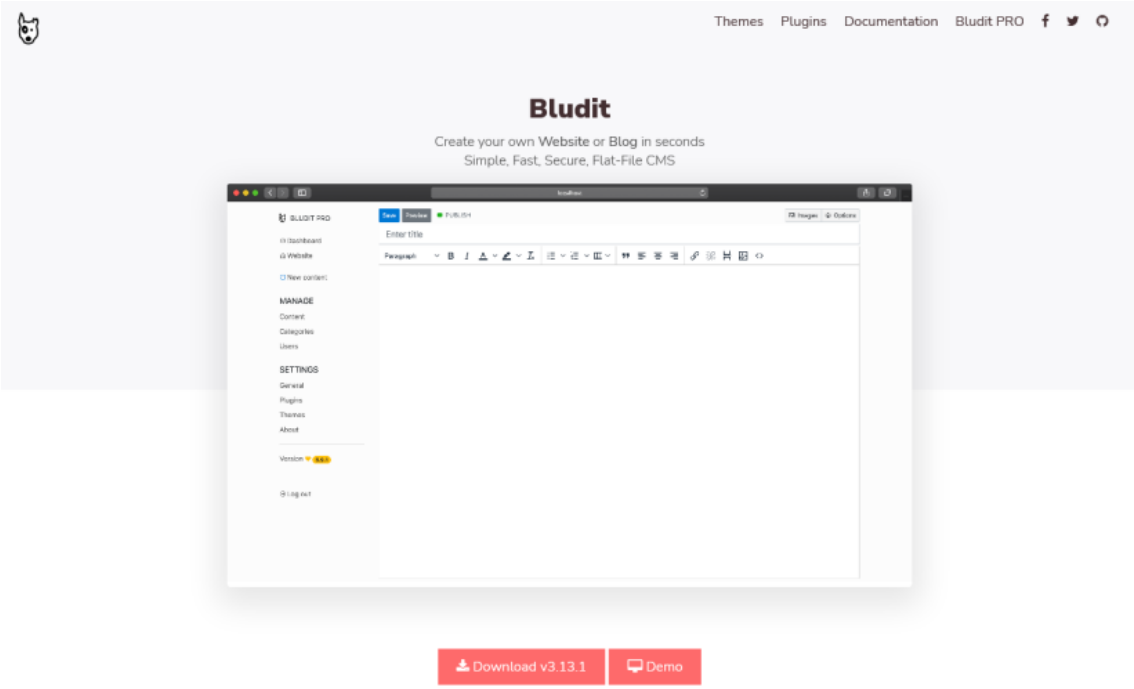
Automad und alle seine Themes sind Open Source und kostenfrei. Die Entwicklung dürfte vergleichsweise günstig sein. Zwar wird man durch die eher geringe Bekanntheit des Systems kaum spezialisierte Entwickler finden. Die Einarbeitung in das System dürfte jedoch nur wenige Stunden erfordern.

4.5 Einsatzsfelder für Automad

Automad ist eine sehr gute Alternative zu WordPress, wenn es um gestalterisch anspruchsvolle Webseiten aus dem Kreativ-Bereich geht: Selbstständige Designer, Künstler und andere Kreativ-Berufe, aber auch kleinere kulturelle Einrichtungen oder kleine Unternehmen aus dem Kreativ-Sektor, die eine aufwändigere Gestaltung wünschen. Die Herausforderung bei der Erstellung solcher Designs wächst zwar mit deren Komplexität. Doch Automad bietet unter den Flat-File-Systemen vermutlich die größten Gestaltungsfreiheiten ohne Zuhilfenahme eines Entwicklers an.

5. Bludit: Einfaches CMS für schlanke Webseiten

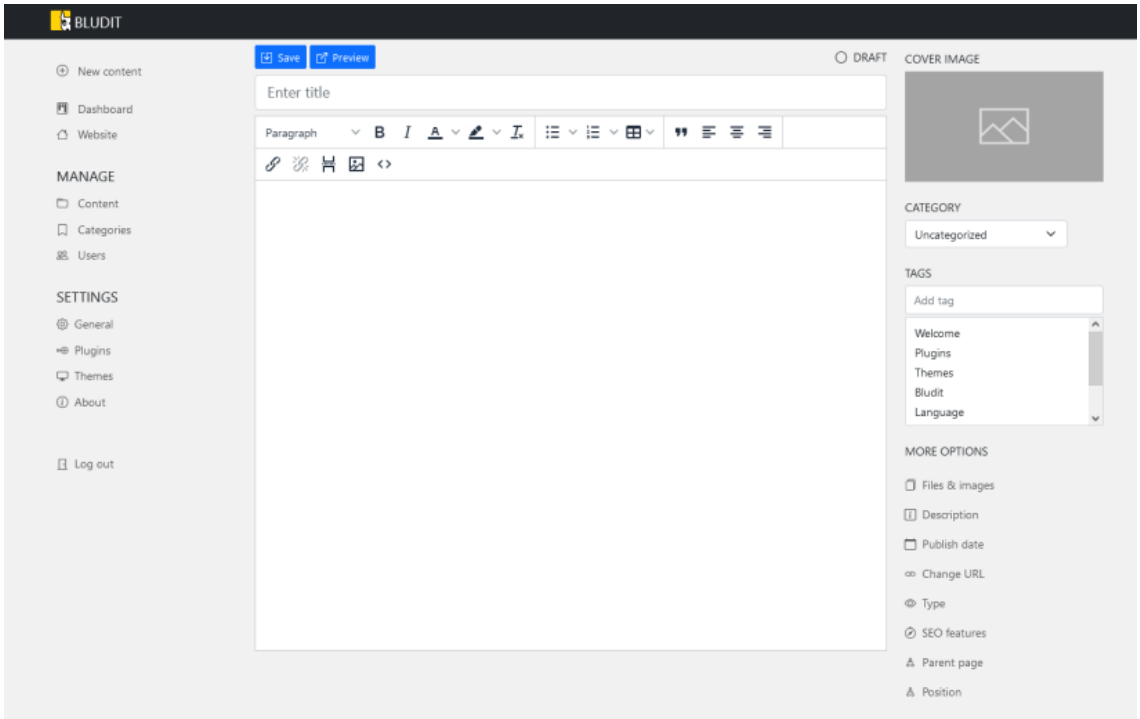
Wer ein einfaches CMS für schlanke Webseiten und Blogs sucht, der wird mit dem Flat File CMS Bludit fündig. Das kleine Open Source System verzichtet weitestgehend auf Besonderheiten und macht genau das, was man erwartet: Autoren bietet Bludit eine nutzerfreundliche Oberfläche an und Entwickler freuen sich über ein einfaches Theming und eine erstaunliche Flexibilität im Hintergrund.



Bludit ist 2015 als modernisiertes Nachfolgesystem des älteren Flat-File-CMS Nibbleblog entstanden. Während Nibbleblog auf XML-Dateien basierte, setzte Bludit bis Version 2 auf den üblichen Markdown-Standard. Mit Bludit 3 ist standardmäßig der HTML-Editor TinyMCE aktiviert. Über ein mitgeliefertes Plugin kann der Autor auf SimpleMDE umsteigen und Markdown nutzen. Derzeit ist Version 4 von Bludit in Arbeit, die bereits als Beta-Release über GitHub getestet werden kann. Für die Review haben wir die Version 4 Beta 2 genutzt. Entwickelt wird Bludit von Djego Najar.

5.1 Bludit für Autoren

Bludit ist für Autoren extrem einfach zu handhaben und weitestgehend selbsterklärend. Angefangen von einer kinderleichten Installation bis hin zu einer anwenderfreundlichen Autoren- und Administrations-Oberfläche, die den normalen Autoren-Erwartungen entspricht und auf Besonderheiten komplett verzichtet. Das macht lästige Erklärungen und selbst eine umfangreiche Dokumentation weitgehend überflüssig.



Auch für anspruchsvollere Autoren hält Bludit einige Features bereit: Neben einem Draft-Management und einem konfigurierbaren Autosafe können Artikel auch zeitversetzt veröffentlicht werden. Ein weiteres nützliches Feature von Bludit sind modulare Inhalte, mit denen sich One-Pager vergleichsweise autorenfreundlich umsetzen lassen. Bei all dem ist das Design der Autorenoberfläche optisch so schlicht gehalten, dass auch unerfahrenere Autoren damit problemlos zurechtkommen dürften.

Die zusätzlichen Optionen für Administratoren sind erstaunlich breit gefächert, angefangen von einfachen Settings wie Website-Title und Sprach-Einstellungen bis hin zu feingranularen Einstellungen der URL-Struktur oder SEO-Einstellungen wie dem Title-Aufbau von Seiten. Mit einem User-Management, einem Tag-Management sowie zahlreichen Plugins und Themes ist Bludit ein komplettes und kompaktes CMS, das Autoren und

Entwicklern alle nötigen Werkzeuge für den Betrieb von Standard-Webseiten, Portfolios, Blogs oder One-Pagern bietet.

5.2 Bludit für Entwickler

Bludit ist ein einfaches aber dennoch sehr sauberes PHP-CMS und kommt weitestgehend ohne externe Abhängigkeiten aus. Im Backend verzichtet Bludit auf ein Framework, im Frontend setzt Bludit etwas konservativ auf JQuery und Bootstrap.

Für die **Installation** von Bludit muss man lediglich über ein Mini-Formular einen initialen Nutzer anlegen und die Sprache auswählen. Anschließend gelangt man auf eine kurze Welcome-Seite und kann sofort loslegen. Einfacher geht es kaum. Die Mindest-Anforderung für die Installation sind PHP 5.6 und ein paar PHP-Module (mbstring, gd, dom, json), die so gut wie überall vorhanden sein dürften. Mit Bludit 4 wird PHP nur noch ab Version 7 unterstützt. Bludit läuft auf diversen Webservern, unter anderem Apache und nginx.

Bei der Speicherung der Inhalten geht Bludit eigene Wege: Sofern nicht der Markdown-Editor ausgewählt wurde, werden alle Text-Inhalte im HTML-Format in einfachen txt-Dateien gespeichert. Alle Meta-Angaben zu den Seiten ebenso wie Kategorien, Tags oder auch die User werden im JSON-Format in separaten PHP-Dateien abgelegt. Diese strickte Trennung von Content und Meta-Inhalten hat Bludit auch die schnelle Umstellung von Markdown auf Rich-Text in Version 3 ermöglicht, da auch in der Markdown-Phase die Content-Files nicht wie üblich mit Frontmatter (YAML) vermischt wurden. Die separate Speicherung von Meta-Angaben im JSON-Format ermöglicht natürlich auch einen sehr einfachen API-Zugriff.

Ein **Theme** besteht bei Bludit aus zwei JSON-Dateien: Eine für die Sprache (`languages/en.json`) und eine für die Meta-Informationen (`metadata.json`).

Das eigentliche HTML-Template findet man in der `Index.php`. Bludit arbeitet also nicht mit separaten Template-Sprachen wie Twig, sondern nutzt gewöhnliche PHP-Tags für die dynamischen Inhalte. In der einfachsten Version sieht ein Template so aus:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Bludit</title>
</head>
<body>
    <?php foreach ($pages as $page): ?>
    <h1><?php echo $page->title(); ?></h1>
    <div><?php echo $page->content(); ?></div>
    <?php endforeach; ?>
</body>
</html>
```

Bei Bludit gibt es tatsächlich keine große Template-Hierarchie, sondern man kann über die reichlich sprechende Variable `$WHERE-AM-I` prüfen, ob man sich auf der Homepage oder auf einer Unterseite befindet. Entsprechend kann man die Inhalte ausgeben:

```
<?php if ($WHERE_AM_I=='page'): ?>
<p>The user is watching a particular page</p>
<?php elseif ($WHERE_AM_I=='home'): ?>
<p>The user is watching the homepage</p>
```

Man kann sein Theme natürlich auch in einzelne Teile (Sidebar, Footer, Home, Page etc.) zerlegen und diese Partials dann einbinden. Die Partials werden in dem Ordner "php" abgelegt und über die Konstante `THEME_DIR_PHP` eingebunden:

```
<?php include(THEME_DIR_PHP.'navbar.php'); ?>
```

Zum Schluss gibt es auch noch diverse Helfer, um Assets wie CSS und JavaScript einzubinden und den Code von Plugin auszuführen:

```
<!-- CSS -->
<?php echo Theme::css('css/style.css') ?>
```

```
<!-- Javascript -->
<?php echo Theme::javascript('js/jquery.min.js') ?>
<!-- Load plugins with the hook siteHead -->
<?php Theme::plugins('siteHead') ?>
```

So einfach kann es gehen und tatsächlich kann man mit diesem simplen und flexiblen Ansatz so gut wie jede einfache Standard-Webseite abbilden.

Bludit lässt sich über **Plugins** erweitern und liefert gleich ein ganzes Dutzend Standard-Plugins mit. Unter den Standard-Plugins findet man auch ein API-Plugin, mit deren CRUD-Methoden sich die Inhalte relativ umfassend verwalten lassen. Die Authentifizierung erfolgt über einen API-Token. Bei gerade mal 4 MB Gesamt-Größe des CMS lässt sich kaum mehr erwarten.

5.3 Bludit für Nicht-Entwickler

Die einfache Installation, die einfache Bedienbarkeit und die vielen Einstellungsmöglichkeiten und Erweiterungen machen Bludit auf jeden Fall auch für Nicht-Entwickler attraktiv. Die Auswahl an Themes ist in den letzten Jahren auf gut 50 angewachsen. Die Erstellung eines eigenen Themes dürfte auch Einsteigern keine großen Schwierigkeiten bereiten. Auch das Plugin-Verzeichnis ist inzwischen vergleichsweise umfangreich.

Einsteiger in PHP dürften vor allem von der schlanken Code-Basis und dem einfachen und sauberen Entwicklungs-Konzept profitieren. Möglich, dass die Dokumentation etwas knapp gehalten ist. Dennoch dürfte man deutlich schneller zum Erfolg kommen, als mit komplexen Systemen wie WordPress.

5.4 Kosten

Bludit ist Open Source und damit völlig kostenfrei (MIT-Lizenz). Es gibt eine Pro-Version von Bludit, die spezielle Plugins enthält und nur für Unterstützer des Projekts zugänglich ist. Bludit kann über die Plattform Patreon unterstützt werden.

Angesichts des sehr flexiblen und dennoch einfachen Konzepts dürfte die Entwicklung von Webseiten mit Bludit relativ kostengünstig sein, solange

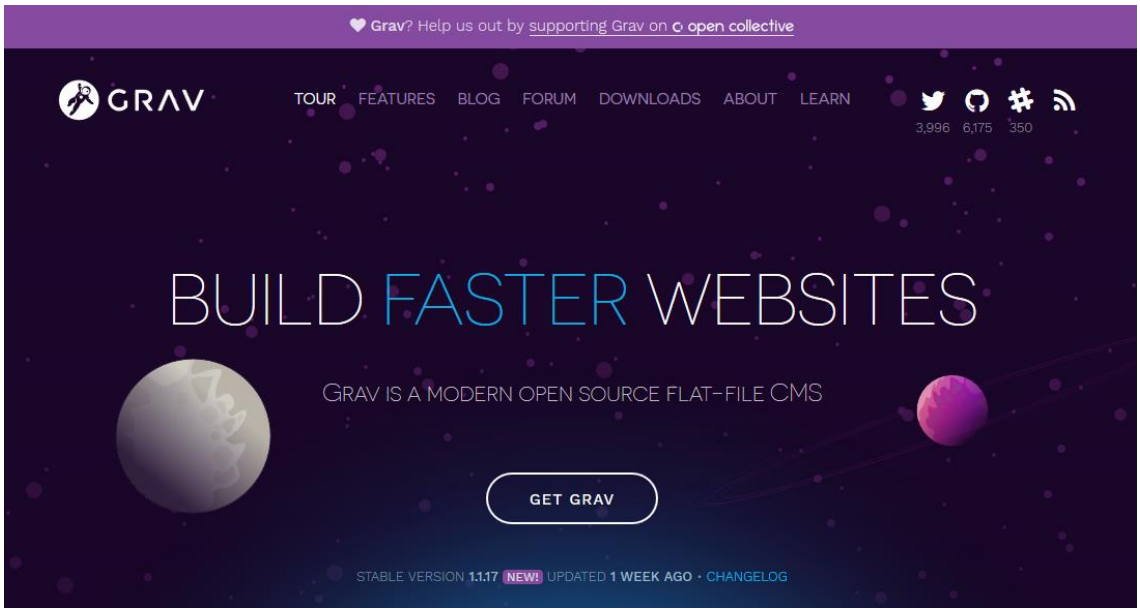
es bei den Standard-Funktionen und eine Anpassung der Standard-Themes bleibt. Will man ein komplett individuelles Konzept, ein individuelles Design oder gar Funktionen, die ein eigenes Plugin erfordern, dann ist der Aufwand weitestgehend unabhängig vom verwendeten CMS immer deutlich höher.

5.5 Einsatzfelder für Bludit

Bludit eignet sich für Standard-Webseiten und einfache Blogs. Größere Funktionen wie Kommentar-Systeme können nur über externe Dienste realisiert werden. Multi-Language wird nicht unterstützt. Für Multi-Language sind entweder zwei separate Installationen von Bludit erforderlich, oder man kann es über etwas kompliziertere Tricks mit verschiedenen Content-Ordnern realisieren. Bei solchen Anforderungen sollte man jedoch eher ein anderes CMS in Erwägung ziehen.

6. GRAV: Das Open Source Flat-File-CMS

Grav ist das bekannteste Open Source System aus der Familie der Flat-File-CMS. Grav bietet viele Features sowie zahlreiche Plugins und Themes, mit denen sich eine große Bandbreite an unterschiedlichen Webseiten realisieren lassen. Vom Blog über Corporate Websites bis hin zu flexiblen One-Pagern ist alles möglich.



Initiator von Grav ist Andy Miller, ein Mitgründer von Joomla! und Chef des bekannten Theme-Marktplatzes "Rocket Themes". Grav erlangte nach seiner Vorstellung im Sommer 2014 schlagartig eine hohe Popularität, denn das System bot erstmals eine ausgereifte Open Source Alternative zu kommerziellen Systemen wie Kirby oder Statamic.

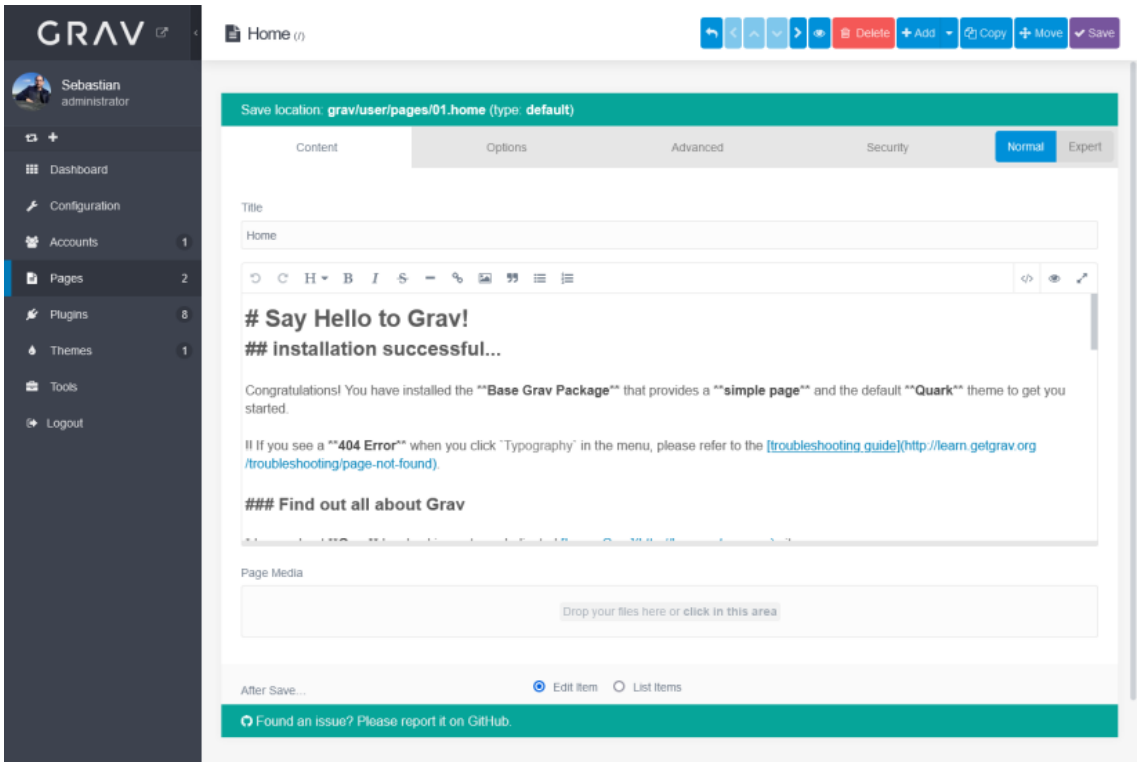
Im Januar 2021 ist mit Version 1.7 das letzte große Release erschienen, seitdem werden eher kleinere Änderungen publiziert. Seit 2020 kann man mit Grav Pro einige Premium-Erweiterungen erwerben. Mit Trilby Media bietet das Grav Team außerdem einen professionellen Entwicklungsservice für Grav an.

6.1 Grav für Content Manager

Grav bietet zwei verschiedene Versionen zum Download an: Eine Version mit einem integrierten Admin-Dashboard und eine Version ohne Dashboard. Grav kann also auch ohne Administrationsoberfläche als Publi-

shing-Tool für Markdown-Files genutzt werden – ein Nutzungs-Szenario, das sich eher an Entwickler richtet.

Das ursprünglich recht dunkel geratene Dashboard ist inzwischen aufgehellt worden. Geblieben ist allerdings der etwas erschlagende Feature-Reichtum. Ein Entwickler kann das Dashboard jedoch recht weitreichend konfigurieren und an die individuellen Bedürfnisse anpassen.



Unter dem Menü-Punkt "Page" werden die Inhalte der Webseite aufgelistet. Dabei unterscheidet Grav nicht zwischen statischen oder dynamischen Inhalten. Stattdessen kann der Autor den Seiten-Typ durch die Auswahl eines Templates bestimmen. Die Eingabefelder für die Inhalte können bei Grav vom Entwickler an die individuellen Bedürfnisse der Webseite angepasst werden.

Für die Hauptinhalte bietet Grav einen Markdown-Editor mit einem Teil-WISYWIG-Modus an. Die Markdown-Syntax bleibt im Editor erhalten, einige Formate wie Überschriften oder Fettungen werden jedoch direkt visuell umgesetzt.

Das Bild-Management von Grav ist etwas eigenwillig: Mit dem Bild-Button des Editors öffnet sich keine visuelle Bild-Auswahl, sondern es wird lediglich die Markdown-Syntax für ein Bild in den Editor integriert. Für die

meisten Autoren dürfte das wenig hilfreich sein, da die fehlende Bild-Url in der Regel nicht bekannt ist. Man kann Bilder jedoch außerhalb des Markdown-Editors per Drag & Drop hochladen und dann auch per Drag & Drop in den Editor verschieben. Im Editor wird dann die Markdown-Syntax für das Bild inklusive der URL zum Bild angezeigt. Fraglich bleibt, ob weniger technisch affine Autoren mit diesem Workflow glücklich werden. Meta-Angaben für Bilder können über eine separat hochgeladene YAML-Datei editiert werden, Bild-Unterschriften sind also nur über diesen Umweg möglich. Wenn Slideshows mit Bild-Unterschriften gewünscht sind, würde man als Entwickler vermutlich auf ein Listenfeld zurückgreifen, mit dem sich wiederholbare Blöcke aus mehreren Eingabefeldern erstellen lassen. Insgesamt stößt man bei solchen Anforderungen zumindest in Bezug auf die Author Experience an die Grenzen des Systems.

Sehr spannend ist dagegen das Konzept der Modular Pages von Grav, mit dem sich komplexe One-Pager recht einfach umsetzen lassen. Die One-Pager setzen sich dabei aus beliebig vielen Abschnitten zusammen, die im Autoren-Dashboard als separate Sub-Pages editiert werden können. Inzwischen hat Grav auch eine Sortierung dieser Abschnitte über die Administrations-Oberfläche ermöglicht, allerdings bleibt die Handhabung etwas umständlich.

Keine Frage, Grav ist enorm flexible und bietet Administratoren und Autoren einen sehr großen Feature-Reichtum an. Tags, Multi-Language-Features, Multi-Site-Fähigkeit - das alles ist mit dem System möglich. Ein kleines Fragezeichen bleibt jedoch hinter der Author Experience, die in den letzten Jahren weitgehend unverändert geblieben ist. Administratoren und Technik-affine Autoren werden mit dem System keine Problem haben und eher die vielen Möglichkeiten begrüßen, beispielsweise die umfangreichen Features zur Bild-Manipulation über URL-Parameter im Markdown Editor. Unerfahrene Autoren könnten jedoch hin und wieder Schwierigkeiten dabei haben, den richtigen Button oder die passende Syntax zu finden. Je nach Einsatzszenario empfiehlt sich daher eine Testrunde mit den Endanwendern.

6.2 Grav für Entwickler

Grav stellt nur minimale Anforderungen an das System. Es läuft auf Web-Servern wie Apache, Nginx, LiteSpeed, Lightly oder IIS mit PHP ab Version

7.3.6.

Der Technologie-Stack des Systems ist inzwischen recht umfangreich, sodass Grav mit ca. 50 MB nicht mehr als leichtgewichtig gelten kann. Für das Frontend der Admin-Oberfläche nutzt Grav JQuery. Im Backend setzt Grav ein hausinternes Framework ein und nutzt zahlreiche externe Libraries. Zu den bekanntesten Libraries zählen:

- **Twig** für das Templating
- **YAML** für Konfigurationen
- **Doctrine-Cache** für die Performance
- **Pimple Dependency Injection Container** für die Erweiterbarkeit
- **Symfony Event Dispatcher** für das Plugin-Event-Handling
- **Symfony Console** als CLI-interface

Wie bei den meisten Flat-File-CMS ist die Installation von Grav sehr einfach. Man lädt die System-Dateien herunter und geht zu `lokalhost` oder im Falle einer Live-Installation zu `deiner-webseite.com`. Sofern man die Version mit der Administrationsoberfläche heruntergeladen hat, erscheint ein Formular zur Erst-Einrichtung eines Admin-Users.

Bei der Erstellung eines neuen Themes für Grav arbeitet man hauptsächlich mit Twig Templates und YAML-Konfigurations-Dateien. Wie bei den meisten Flat-File-Systemen kommt es auf das Zusammenspiel zwischen den Templates, den Konfigurations-Dateien und den Inhalten aus der Markdown an:

- **default.md**: Das ist die Content-Datei im Markdown-Format. Die Markdown-Datei wird mit dem gleichnamigen Twig-Template `default.html.twig` gerendert.
- **default.html.twig**: Im Twig-Template hat man über ein `page`-Objekt Zugriff auf sämtliche Inhalte und Meta-Angaben der Seite. Die Inhalte und Meta-Angaben kann der Autor im Dashboard über Formulare eingeben, die der Entwickler vorher in der YAML-Datei `default.yaml` definiert hat.
- **default.yaml**: Die YAML-Dateien mit den Konfigurationen und Formular-Definitionen werden bei Grav "blueprints" genannt.

Die Definition eines Titel-Feldes und eines Markdown-Editors für das Autoren-Dashboard sieht in der YAML-Datei etwa so aus:

```
fields:
  header.title:
    type: text
    label: Headline
  content:
    type: markdown
    validate:
      type: textarea
```

Wählt der Autor bei der Erstellung einer Seite das Template "Default" aus, dann werden die Meta-Angaben wie üblich im Frontmatter der Markdown-Datei gespeichert. Unter dem Frontmatter folgen die Inhalte im Markdown-Format:

```
---
title: My page
---
# This is a headline in markdown
This is some text in markdown. There are some formattings like **bold** or even a
[Link to a website](http://cmsstash.com).
```

Im Template "default.html.twig" stehen sämtliche Inhalte als "page"-Objekt zu Verfügung und können vom Entwickler mit der typischen Twig-Syntax ausgegeben werden:

```
{% block content %}
    <h1>{{ page.header.title }}</h1>
    {{ page.content }}
{% endblock %}
```

Gerade bei der Definition von Formularen bietet Grav sehr viele Zusatz-Features an, angefangen von Extend-Features über Function Calls bis hin zur Definition eigener Feld-Typen.

Das Plugin-System von Grav baut auf dem Event-Dispatcher von Symfony auf, der weit verbreitet ist und vielen Plugin-Entwicklern bekannt sein

dürfte. Der schnellste Weg zur Plugin-Entwicklung dürfte über das kurze Plugin-Tutorial in der Grav-Dokumentation führen.

Mit Version 1.7 sind im Januar 2021 umfangreiche technische Änderungen veröffentlicht worden, insbesondere die neuen Flex-Objects, mit denen sich eigene Datentypen definieren und nutzen lassen. Im Gegensatz zu den meisten anderen Flat-File-Systemen bietet Grav allerdings bis heute keine REST-API im Core-System an. Headless-Konzepte sind dadurch nur über Plugin-Erweiterungen oder andere Umwege möglich.

6.3 Grav für Nicht-Entwickler

Grav ist einfach zu installieren, es bietet eine gute Auswahl an Ressourcen an und kann über die Administrations-Oberfläche sehr weitreichend konfiguriert werden. Von daher können auch Nicht-Entwickler dem System eine Chance geben. Dennoch sollte man kein Ready-To-Go-Konzept analog zu WordPress erwarten. Was man erwarten kann sind über 100 Themes, mehr als 300 Plugins und einige Dutzend "Skeletons", die als fertige Lösungen für diverse Anwendungsfälle genutzt werden können. Die Bandbreite ist dabei recht hoch: Von der Business-Webseite über Portfolio-Seiten, Landing-Pages bis hin zu Blogs sind viele Standard-Fälle abgedeckt. Die Administrations-Oberfläche ist allerdings recht komplex und zum Teil mit technischen Fachbegriffen bestückt, sodass man eine gewisse technische Affinität haben sollte, wenn man mit Grav ohne Hilfe eines Entwicklers loslegen will.

6.4 Kosten

Das Core-System von Grav sowie die meisten Themes und Plugins sind Open Source und kostenfrei. Seit 2020 sind mit Grav Premium bislang zehn kostenpflichtige Erweiterungen hinzugekommen. Die Lizenz wird einmalig gekauft und gilt für eine Produktivseite. Die Kosten variieren je nach Erweiterung zwischen 25,- und 100,- Dollar.

Neben den kostenpflichtigen Erweiterungen bietet das Team von Grav mit Trilby Media einen Entwicklungsservice an. Außerdem kann Grav über Open Collective unterstützt werden. Damit nutzt Grav zahlreiche Möglichkeiten, um das doch recht umfangreiche Open Source Projekt zu refinanzieren.

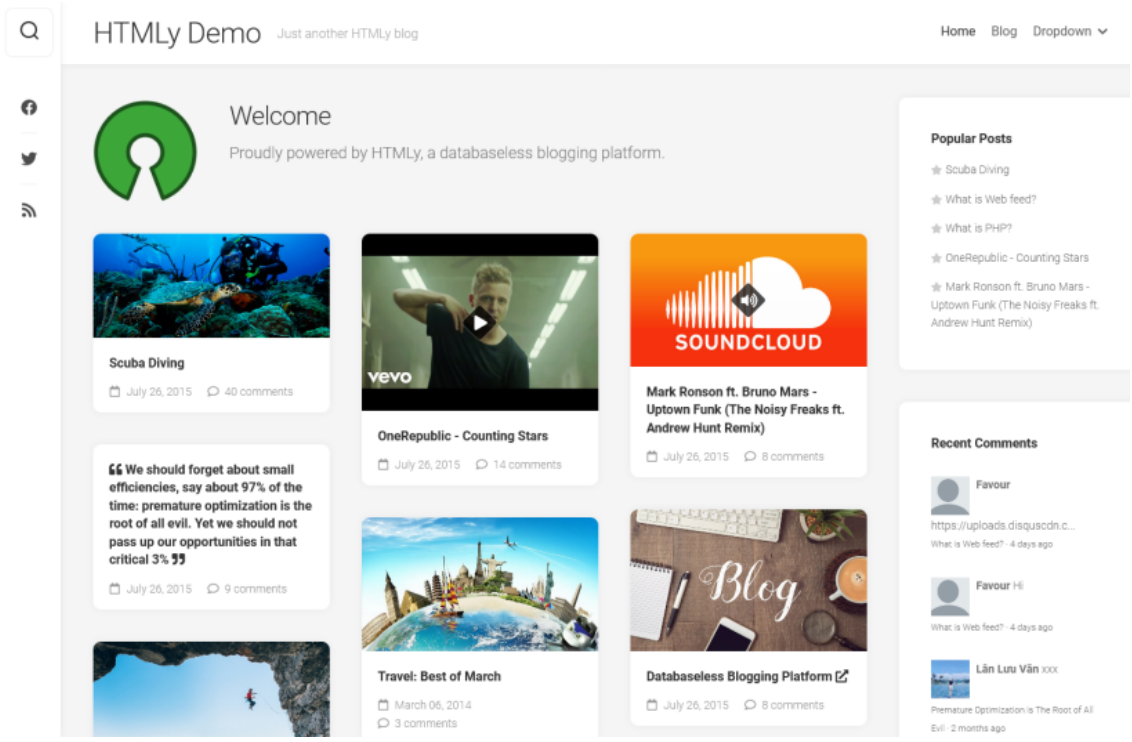
6.5 Einsatzfelder für Grav?

Flat-File-CMS sind generell sehr gut für alle Arten von kleineren Webseiten geeignet, seien es Landing-Pages, Portfolio-Seiten oder kleine Business-Auftritte. Durch die Skeletons ist Grav auch für viele andere Anwendungsfälle geeignet, beispielsweise für Verzeichnisse, Dokumentationen und sogar für kleine E-Commerce-Seiten. Besonders sinnvoll ist der Einsatz von Grav bei One-Pagern.

Grundsätzlich sollte man vor dem Einsatz von Grav prüfen, ob das jeweilige Seiten-Konzept mit akzeptablen und verständlichen Workflows für die Autoren realisierbar ist, denn grundsätzlich gehört die Author Experience für technisch unerfahrenere Autoren nicht unbedingt zu den Stärken von Grav. Auch Headless-Konzepte sind aufgrund der fehlenden REST-API mit Grav eher weniger sinnvoll. Darüber hinaus sind die weitreichenden Konfigurationsmöglichkeiten und generell die Flexibilität des Systems jedoch beeindruckend.

7. HTMLy: Ein Flat-File-CMS für Blogger

HTMLy ist ein leichtgewichtiges Flat-File-System speziell für Blogs. Wer einen einfachen und schnellen Standard-Blog ohne großen Aufwand für Installation, Konfiguration und Wartung sucht, der ist mit HTMLy gut bedient.



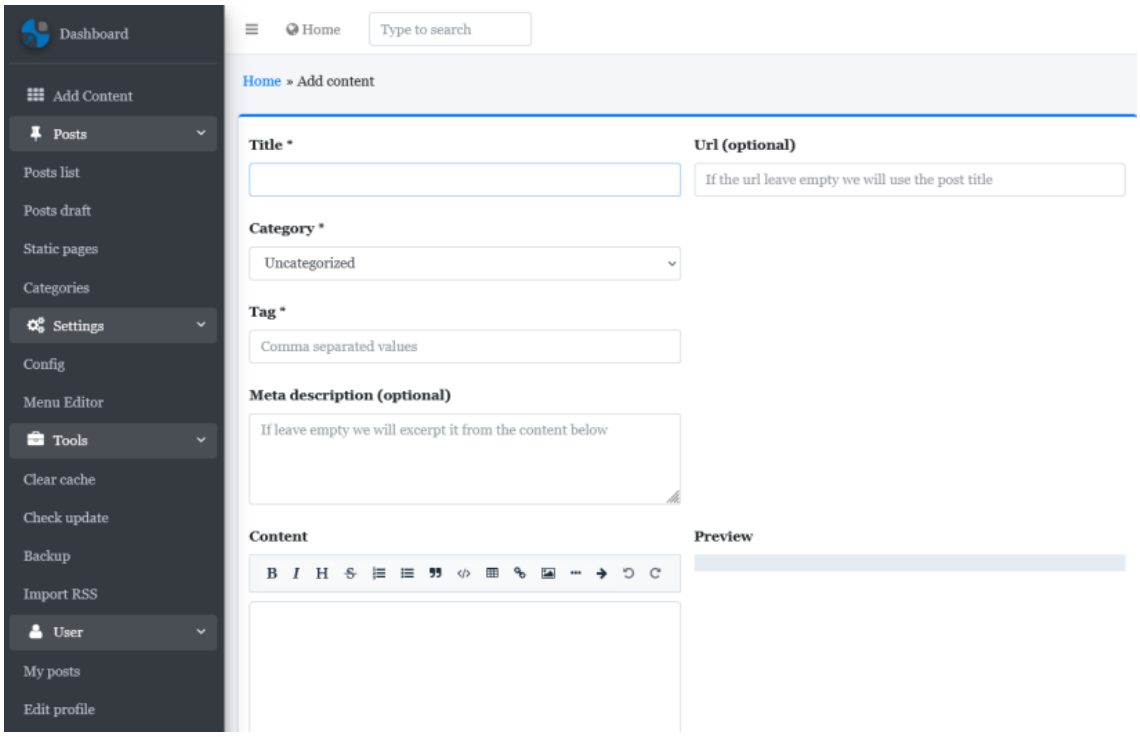
HTMLy wurde erstmals im Jahr 2013 von dem Entwickler Danang Probo Sayekti veröffentlicht. Im Jahr 2016 wurde die Weiterentwicklung ausgesetzt. Seit 2020 gibt es unregelmäßige Aktualisierungen, allerdings ist die Weiterentwicklung im Vergleich zu anderen Systemen weniger dynamisch und nicht sehr stetig.

7.1 HTMLy für Autoren

Als kleines und schlankes CMS verfügt HTMLy über alle notwendigen Features für den Betrieb eines einfachen Blogs. Ähnlich wie Tumblr ermöglicht HTMLy verschiedene Arten von Content-Beiträgen. Neben normalen Text-Inhalten gehören dazu Zitate, Bilder sowie Audio- und Video-Beiträge. Dabei unterstützt HTMLy allerdings nur spezielle Plattformen wie YouTube für Videos oder Soundcloud für Audio-Streams. Wer diese Plattformen ohnehin nutzt, für den ist HTMLy ein schneller Startpunkt für das

Instant-Blogging. Wer andere Plattformen wie Vimeo verwendet, der muss HTMLy dafür eigenhändig erweitern.

HTMLy nutzt für die Erstellung von Inhalten die Auszeichnungssprache Markdown und bietet dazu einen Split-Screen mit einer WYSIWYG-Preview an. Auch Autoren, die bislang mit HTML-Editoren gearbeitet haben, dürften damit zurechtkommen.



Auch die Konfigurations-Möglichkeiten von HTMLy lassen für einen schlichten Blog kaum Wünsche übrig: In der Konfiguration kann der Administrator zwischen sechs verschiedenen Themes auswählen. Es gibt Beiträge, Entwürfe, Tags und Feeds. Bei der Installation kann man optional die externen Kommentarsysteme von Facebook oder Disqus integrieren. Außerdem bietet das System eine Google-Sitemap, Content-Teaser, Read-More-Buttons, verwandte Artikel, neueste Posts, beliebteste Beiträge, Permalinks und Social-Media-Buttons. Hinzu kommen nützliche Features wie ein automatisches Update, einen RSS-Import und eine Backup-Möglichkeit. Mit Version 2.7.7 ist als neues Feature ein einfacher Menu-Builder hinzugekommen.

Mit der Wiederbelebung im Jahr 2020 hat HTMLy die zuvor reichlich schmucklos daherkommende Administrations-Oberfläche erneuert und

auf Bootstrap umgestellt. Damit ist die Arbeit für Autoren und Administratoren insgesamt deutlich angenehmer geworden.

7.2 HTMLy für Entwickler

HTMLy ist ein schlankes PHP-System mit minimalen Anforderungen. Es läuft ab PHP 5.3 und benötigt nur einige Standard-Libraries wie PHP-XML und PHP-ZIP. Unterstützt werden die Server Apache, Nginx und Lighttpd. Für die Installation müssen lediglich die Dateien auf den Server geladen und das Installationsscript unter `install.php` aufgerufen werden. Nach ein paar Angaben in der Installations-Routine läuft das System.

Im Backend verzichtet HTMLy vollständig auf Frameworks und setzt mit Markdown, Urlify und einem Passwort-Script lediglich drei externe Libraries ein. Für das Admin-Panel werden Bootstrap und JQuery verwendet. Ein Großteil des Codes von HTMLy folgt dem prozeduralen Stil. Auch Templates werden lediglich mit HTML und PHP erstellt.

Für Themes wird eine relativ feste Template-Struktur vorgegeben. Die Templates decken Varianten wie Profil-Seiten, Blog-Einträge, Beitrags-Listen oder auch Tag-Listen ab. Sämtliche Template-Typen können über Bennenungsregeln auch für Unterseiten definiert werden. In den Templates selbst stehen dann individuelle Funktionen und Variablen zur Verfügung. Der Code für das Main-Template kann beispielsweise so aussehen:

```
<!DOCTYPE html>
<html>
<head>
    <?php echo head_contents(); ?>
    <title><?php echo $title; ?></title>
    <meta name="description" content="<?php echo $description; ?>"/>
    <link rel="canonical" href="<?php echo $canonical; ?>" />
    <?php if (publisher()): ?><link href="<?php echo publisher(); ?>"
rel="publisher" /><?php endif; ?>
    <link rel="stylesheet" type="text/css" href="<?php echo site_url();?
>themes/theme-name/css/style.css">
</head>
<body>
<?php if (facebook()) { echo facebook(); } ?>
```

```
<?php if (login()) { toolbar(); } ?>
    <div class="header">
        <?php if(is_index()) {?>
            <h1 class="blog-title"><a href="<?php echo site_url();?>"><?php echo
blog_title();?></a></h1>
        <?php } else { ?>
            <h2 class="blog-title"><a href="<?php echo site_url();?>"><?php echo
blog_title();?></a></h2>
        <?php } ?>
    <div>
    <div class="wrapper">
        <div class="main-content">
            <?php echo content(); ?>
        </div>
        <div class="sidebar">
            <div class="search">
                <?php echo search(); ?>
            </div
        </div>
    </div>
    <div class="footer">
        <?php echo copyright() ?>
    </div>
<?php if (analytics()): ?><?php echo analytics() ?><?php endif; ?>
</body>
</html>
```

Komplexere Inhalte wie Artikellisten nach Tags oder nach Kategorien können mit Hilfs-Funktion wie `get_tag()` oder `get_category()` ausgegeben werden. Daneben gibt es sogenannte Widgets. Dahinter verbergen sich Funktionen, mit denen sich ganze Features wie eine Suche, eine Tag-Cloud und ähnliche Blog-Standards integrieren lassen. Insgesamt stehen neun Widget-Funktionen zur Verfügung:

```
search()
menu()
recent_posts()
popular_posts()
```

```
archive_list()
tag_cloud()
category_list()
get_related($p->related)
recent_type($type)
```

Ein Problem bei HTMLy war schon immer die recht dürftige Dokumentation. Sie kann zwar in weniger als einer Stunde durchgelesen werden und liefert alle nötigen Informationen für die Erstellung eines Themes. Wer allerdings das System funktional erweitern will, ist auf sich selbst gestellt. HTMLy bringt zwar einen leeren Ordner für Plugins mit. Dazu sind jedoch nirgendwo Informationen oder Beispiele zu finden. Bei der kleinen Codebasis von HTMLy wird man sicherlich schnell Anhaltspunkte finden oder schlicht über Trial & Error ans Ziel gelangen. Als Entwickler würde man sich jedoch zu diesem Thema einen schnelleren Einstieg wünschen.

7.3 HTMLy für Nicht-Entwickler

HTMLy wirkt wie ein schlankes und rudimentäres WordPress ohne Plugins. Es wird mit sechs fertigen Themes ausgeliefert und das System bietet ausreichende Konfigurationsmöglichkeiten für die Individualisierung des Webauftritts an. Für Einsteiger in die Web-Entwicklung dürfte HTMLy aufgrund seiner vergleichsweise einfachen Code-Basis ein guter Startpunkt zumindest in die Theme-Entwicklung sein. Ähnlich wie in den Anfangszeiten von WordPress kann man bei HTMLy einfache Hilfs-Funktionen kopieren und so sein Theme anpassen oder komplett neu entwickeln. Allerdings steht im Vergleich zu WordPress nur eine sehr dünne Dokumentation zur Verfügung, die es Neulingen wiederum schwer machen dürfte.

7.4 Kosten

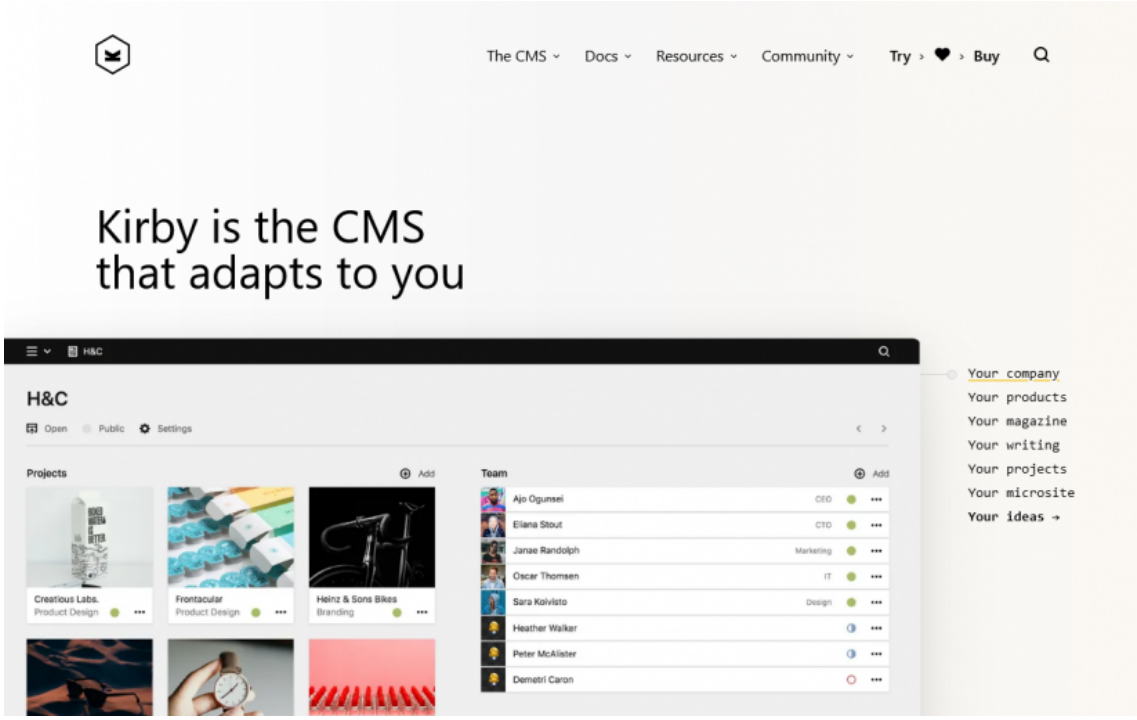
HTMLy und alle Themes sind Open Source und kostenlos. Aufgrund der einfachen Theme-Entwicklung dürften die Kosten für ein eigenes Theme im Falle einer externen Beauftragung vergleichsweise gering sein.

7.5 Einsatzfelder für HTMLy

HTMLy ist eine gute Wahl, wenn man schnell einen einfachen Standard-Blog aufsetzen will. Das System lässt sich auch ohne Code-Kenntnisse schnell installieren und einfach bedienen. Der Vorteil gegenüber großen Systemen wie WordPress ist der Verzicht auf eine Datenbank und eine Wartungsfreiheit, was dem Blog-Betreiber viel unnötige Arbeit erspart. Damit ist HTMLy auch ein guter Kandidat für ein erstes Proof of Concept, das man im Erfolgsfall immer noch auf ein komplexeres System umstellen kann.

8. Kirby: Der Flat-File-Pionier

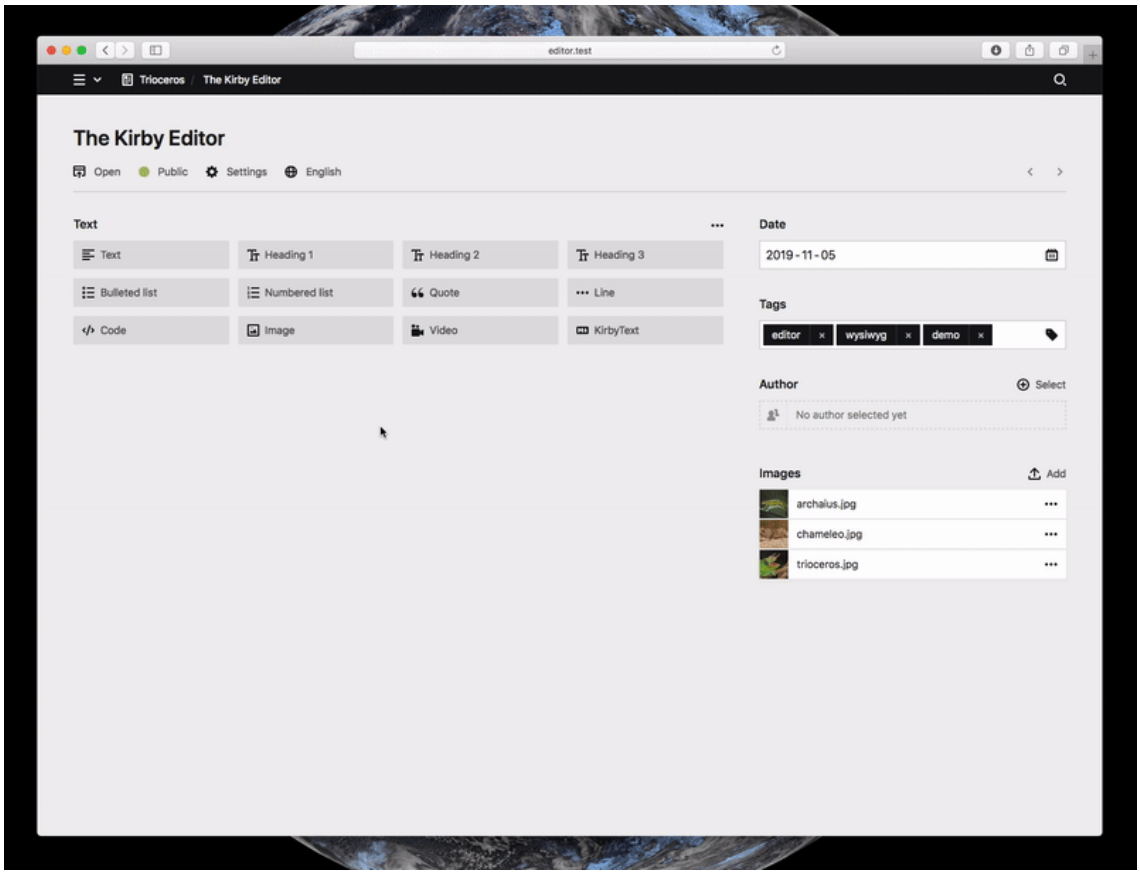
Kirby gehört zu den Pionieren unter den Flat-File CMS und steht für eine hohe Kontinuität und Professionalität. Das System ist im Jahr 2012 entstanden und hat als kommerzielles Flat-File-CMS Maßstäbe gesetzt. Im Januar 2019 ist mit Kirby 3 eine modernisierte Version veröffentlicht worden, die für Autoren und Entwickler zahlreiche neue Features mitbringt. Kennzeichen für Kirby bleibt jedoch die extrem einfache Handhabung für beide Anwendergruppen.



Kirby ist als klassisches Nebenprojekt im Jahr 2011 entstanden und hat seit dem ersten Launch Anfang 2012 eine spannende Entwicklung durchgemacht. Fünf Jahre nach dem ersten Release konnte der Gründer Bastian Allgeier aus dem Nebenprojekt schließlich einen Hauptjob machen. Inzwischen hat sich um den Gründer ein kleines Team geschart. Mit einem moderaten Preis richtet sich Kirby vor allem an Freelancer, Web-Agenturen und generell an Unternehmen, die eine schlanke Lösung für ihre Kunden bzw. ihre Webseite suchen.

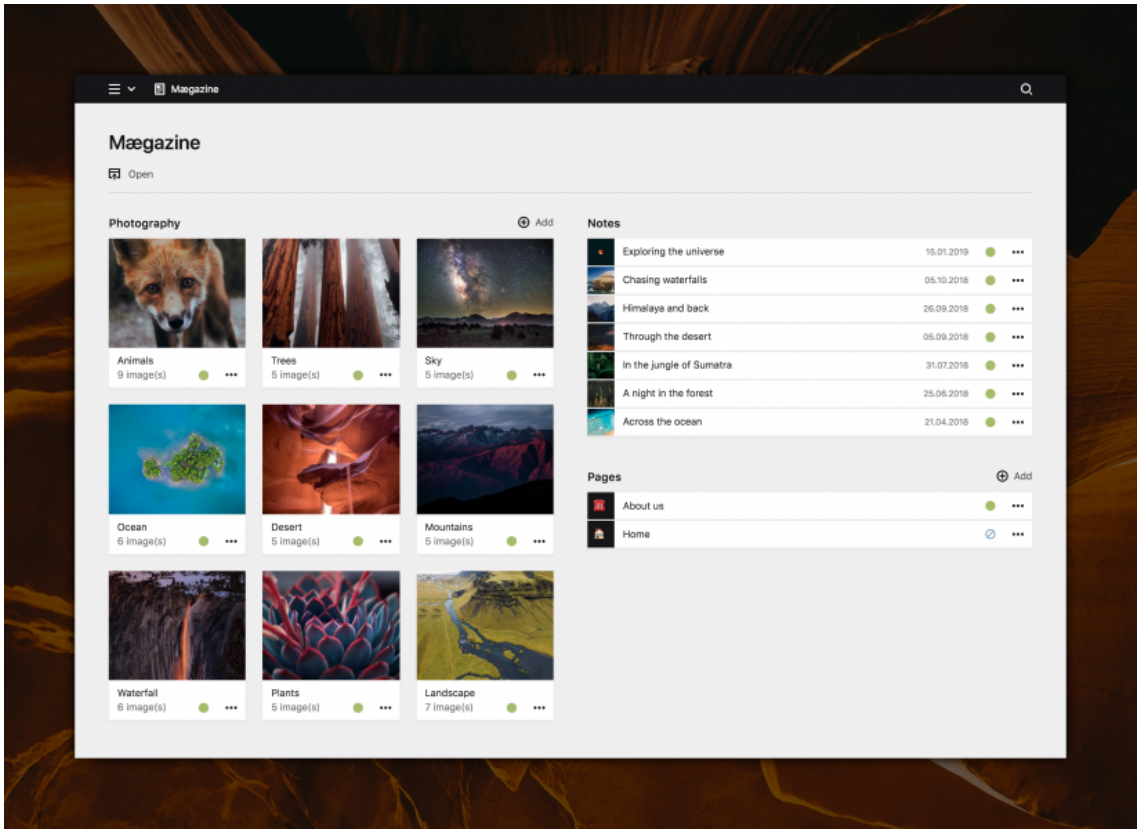
8.1 Kirby für Autoren

Modern, intuitiv und nutzerfreundlich: Die Autorenoberfläche von Kirby gehört sicherlich zu den besten in der CMS-Welt und stellt die Author Experience vieler bekannter Systeme älteren Datums in den Schatten. Mit Version 3.3.0 hat Kirby Ende 2019 ein neues Editor-Plugin vorgestellt, mit dem sich Inhalte ähnlich wie mit dem Gutenberg-Editor von WordPress visuell mit Hilfe von Content-Blöcken erstellen lassen. Der Block-Editor hat sich offenbar auch bei den Nutzern bewährt und ist im Dezember 2020 mit Version 3.5.0 in den Core des Systems integriert geworden. Ein visuelles Page-Editing im Block-Stil war im Flat-File-Bereich bereits von Automad, von Statamic mit dem Bard-Editor und in anderer Form auch von Typemill bekannt. Der Editor von Kirby überzeugt auf Anhieb mit einer guten Usability:



Ein weiteres Highlight aus Version 3.5.0 ist ein neues Layout-Field, mit dem sich komplexe Multi-Column-Layouts realisiseren lassen. Bereits in Version 3.0.0 hat Kirby auch das Bild-Management deutlich verbessert und um komfortable Bild-Galerien ergänzt. Je nach Konfiguration lassen

sich die Bilder-Gallerien zu einer Seite hinzufügen und anschließend mit dem Tag `{{ gallery }}` in den Content-Bereich integrieren, ähnlich wie man es mit den Shortcodes von WordPress gewohnt ist. Mit der Ende 2021 erschienenen Version 3.6.0 sind moderne Bildformate wie WebP & AVIF hinzugekommen. Daneben wurden mit Version 3.6.0 zahlreiche Erweiterungen beim Block-Editor vorgestellt.



Für Redakteure sehr nützlich ist auch der erweiterte und von einem Entwickler modifizierbarer Publishing-Workflow von Kirby 3. Neben den üblichen nicht öffentlichen "Drafts" und der publizierten Version gibt es nun auch eine "Review". Die Review ist zwar über eine URL öffentlich zugänglich, jedoch nirgendwo gelistet oder verlinkt. Dieser Status ist durchaus nützlich, wenn die Seite während der Überarbeitung nicht depubliziert werden soll, um zum Beispiel eine Entfernung aus dem Google-Index zu vermeiden. Das Angebot von Kirby 3 wird schließlich durch ein Internationalisierungs-Feature für Multi-Language-Seiten abgerundet.

8.2 Kirby für Entwickler

Kirby ist ein extrem schlankes PHP-CMS mit nur wenigen MBs, sofern man den Beispiel-Content ausklammert. Angesichts der vergleichsweise langen Geschichte von Kirby ist dieser Minimalismus eine erstaunliche Leistung. Kirby verzichtet vollständig auf Backend-Frameworks und setzt nur die notwendigsten Bibliotheken ein. Dieser Minimalismus ermöglicht es auch einem sehr kleinen Team, das System permanent zu erneuern und auf den aktuellsten Stand zu halten.

Und der neueste Stand von Kirby hat es in sich: Die Autoren-Oberfläche wurde komplett auf das schlanke und beliebte Framework Vue.js umgestellt und im Backend gibt es nun eine vollständige REST-API, die auch Headless-Ansätze, Single Page Applications (SPA) oder die Integration von statischen Site Generatoren (SSG) ermöglicht. Damit wurden gleich drei Konzepte in Kirby integriert, die heute fast jeden Entwickler umtreiben.

Eine spannende Neuerung hat Kirby mit Fiber in Version 3.6.0 vorgestellt. Inspiriert von Inertiajs hat sich Kirby mit Fiber beim Admin-Panel von der Single-Page-Architektur wieder verabschiedet und das Routing und Rendering zurück an den Server delegiert. Die hohe Popularität von Inertiajs ist möglicherweise ein Indikator dafür, dass traditionellere Ansätze in neuer Form wieder mehr Rückwind in der Entwicklergemeinde bekommen.

Geblieben ist die bewährte Zero-Click-Installation. Nach wie vor kann man Kirby einfach herunterladen, auf localhost oder auf einen Live-Server schieben und ohne jede weitere Konfiguration die Seite besuchen. Der Zugang zur Autoren- bzw. Administrations-Oberfläche erfordert lediglich eine Registrierung als Administrator. Geblieben ist nach wie vor auch die sehr vorbildliche Dokumentation, die Entwicklern den Einstieg in das System stark vereinfacht.

Wie die meisten modernen Flat-File-Systeme nutzt auch Kirby YAML-Dateien für die Konfiguration des Systems. Anders als andere Systeme nutzt Kirby allerdings keine separate Template-Sprache wie Twig, sondern arbeitet mit reinem PHP und HTML. Beim Templating arbeitet man überwiegend mit den beiden Objekten `$site` für seitenweite Informationen und `$page` für den Content der aktuellen Seite. Ein einfaches Template kann beispielsweise so aussehen (entnommen aus der Kirby-Dokumentation):

```
html>
<head>
  <meta charset="UTF-8">
  <meta name="description" content="<?= $site->description() ?>">
  <meta name="keywords" content="<?= $site->keywords() ?>">
  <title>
    <?= $page->title()->html() ?> | <?= $site->title()->html() ?>
  </title>
</head>
<body>
  <header>
    <h1>
      <a href="<?= $site->url() ?>">
        <?= $site->title()->html() ?>
      </a>
    </h1>
  </header>
  <main>
    <h1><?= $page->title()->html() ?></h1>
    <?= $page->text()->kirtbytext() ?>
  </main>
  <footer>
    <p class="copyright"><?= $site->copyright()->html() ?></p>
  </footer>
</body>
</html>
```

Neben der Erweiterbarkeit über Plugins dürfte vor allem die REST-API von Kirby 3 für Entwickler spannend sein. Die angebotenen Authentifizierungs-Möglichkeiten sind dabei recht eigen: Während man bei APIs heute überwiegend auf Jason-Web-Tokens (JWT) und eher selten auf klassische API-Tokens trifft, setzt Kirby auf eine Authentifizierung mit Sessions oder auf Basic Auth, sofern eine komplett Stateless-Nutzung der API erforderlich ist. Bei der Authentifizierung per Session muss ein CSRF-Token übergeben werden:

```
<html>
  <head>
    <title>API Example</title>
  </head>
  <body>
    <script>
      const csrf = "<?= csrf() ?>";
      fetch("/api/pages/example", {
          method: "GET",
          headers: {
            "X-CSRF" : csrf
          }
        })
        .then(response => response.json())
        .then(response => {
          const page = response.data;
          // do something with the page data
        })
        .catch(error => {
          // something went wrong
        });
    </script>
  </body>
</html>
```

Bei der Basic Auth erfolgt die Authentifizierung über den entsprechenden Header mit E-Mail und Passwort:

```
<?php
$response = Remote::get('https://yoursite.com/api/pages/example', [
  'headers' => [
    'Authorization: Basic ' . base64_encode($email . ':' . $password)
  ]
]);
$page = json_decode($response->content())->data;
```

Es gibt bereits eine umfangreiche Dokumentation der REST-API, sodass Entwickler keine Probleme bei der Nutzung haben dürften. Zahlreiche Hilfestellungen bekommt man auch über das Forum oder das Cookbook.

Alles in allem kann man sich kaum mehr von einem kleinen Team rund um ein schlankes Flat-File-System wünschen. Mit der Version 3 hat Kirby erneut sein Gespür für Trends bewiesen und ist gleichzeitig seinem minimalistischen Prinzip treu geblieben.

8.3 Kirby für Nicht-Entwickler

Kirby richtet sich in erster Linie an Entwickler, Freelancer, Web-Agenturen und Unternehmen, die mit einem schlanken und modernen CMS ohne wochenlange Einarbeitung eine Webseite erstellen wollen. Zwar gibt es für Kirby inzwischen deutlich über 100 Themes. Die Anpassungsmöglichkeiten sind ohne Code-Kenntnisse jedoch eingeschränkt.

Durch die Verwendung von reinem PHP, einer sauberen Code-Basis und einer sehr guten Dokumentation eignet sich Kirby bestens für PHP-Einsteiger. Im Vergleich zu WordPress ist das System sehr schlank und überschaubar, gleichzeitig gibt es ein klares Konzept und eine einheitliche Herangehensweise, was den Einstieg für Neulinge vereinfacht. Die REST-API und die damit geschaffenen Andock-Stellen für Frontend-Frameworks machen Kirby natürlich auch zu einem sehr guten Einstiegspunkt für Frontend-Entwickler.

8.4 Kosten

Mit Version 3 hat Kirby einen einheitlichen Preis von 99,- Euro pro Webseite eingeführt und die kostenlose Lizenz für private Seiten eingestellt. Vielleicht kein Zufall, dass Statamic mit seiner neuen Version nun erstmals eine kostenlose Lizenz für private Webseiten vergibt. Durch den vergleichsweise hohen regulären Preis von Statamic bleibt Kirby für kommerzielle Webseiten jedoch die deutlich günstigere Variante.

Eine Test-Version für Entwickler kann man kostenfrei herunterladen und lokal verwenden. Einzelne Gruppen wie Studenten, Bildungsprojekte, Open-Source-Projekte oder Non-Profits können unter bestimmten Bedingungen Rabatte erhalten. Zusätzlich kann man Mengenrabatte erhalten.

8.5 Einsatzfelder für Kirby

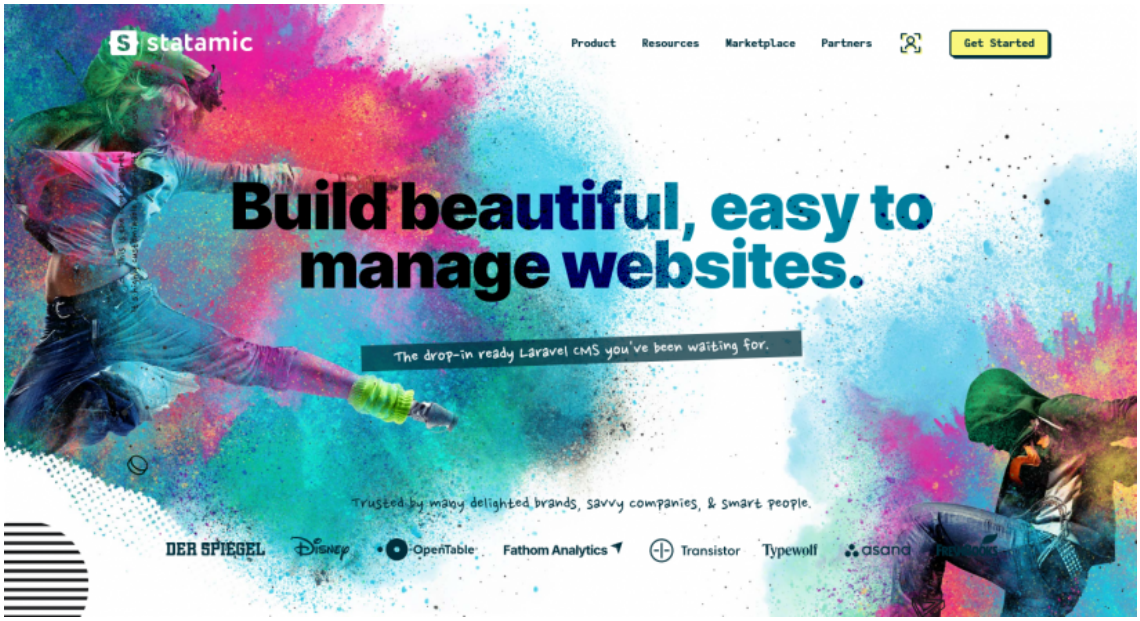
Kirby eignet sich grundsätzlich für kleine und auch etwas größere Web-Projekte, bei denen Einfachheit, Flexibilität und eine gute Usability für die Autoren gefragt sind. Mit der neuen Version 3 und der REST-API öffnet Kirby außerdem eine Spielwiese für Single Page Applications oder Static Site Generatoren und ermöglicht die Kombination mit modernen Front-end-Frameworks wie React, Vue oder Svelte.

Eine kleine Einschränkung gibt es bei Kirby: Das System ist nicht darauf ausgelegt, ein Theme per Knopfdruck zu wechseln. Stattdessen ist das Theme stark mit dem System verwoben. Wer also häufiger Design-Umstellungen plant, sollte gegebenenfalls auf ein anderes System setzen.

Kirby ist zwar ein kleines Projekt, wurde aber bereits von einer Reihe namhafter Unternehmen und Projekten eingesetzt. Darunter das Smashing Magazine, Yeebase (T3N), und auch einige Projekte von großen Konzerne. Im aktuellen Showcase findet man vor allem kleinere Auftritt unter anderem aus dem Kulturbereich, zum Beispiel Max Raabe oder Frank Schätzung.

9. Statamic: Flat-File-CMS on Steroids

Statamic ist ein junges und aufstrebendes Flat File CMS, das inzwischen auch von der großen Unternehmenwelt entdeckt wurde. So ist zum Beispiel der Spiegel mit seinem Relaunch Anfang 2020 vollständig auf Statamic als Editor umgezogen. Durch seine hohe Flexibilität und seine hervorragende Usability eignet sich Statamic jedoch vor allem für kleinere Unternehmen und den Mittelstand.

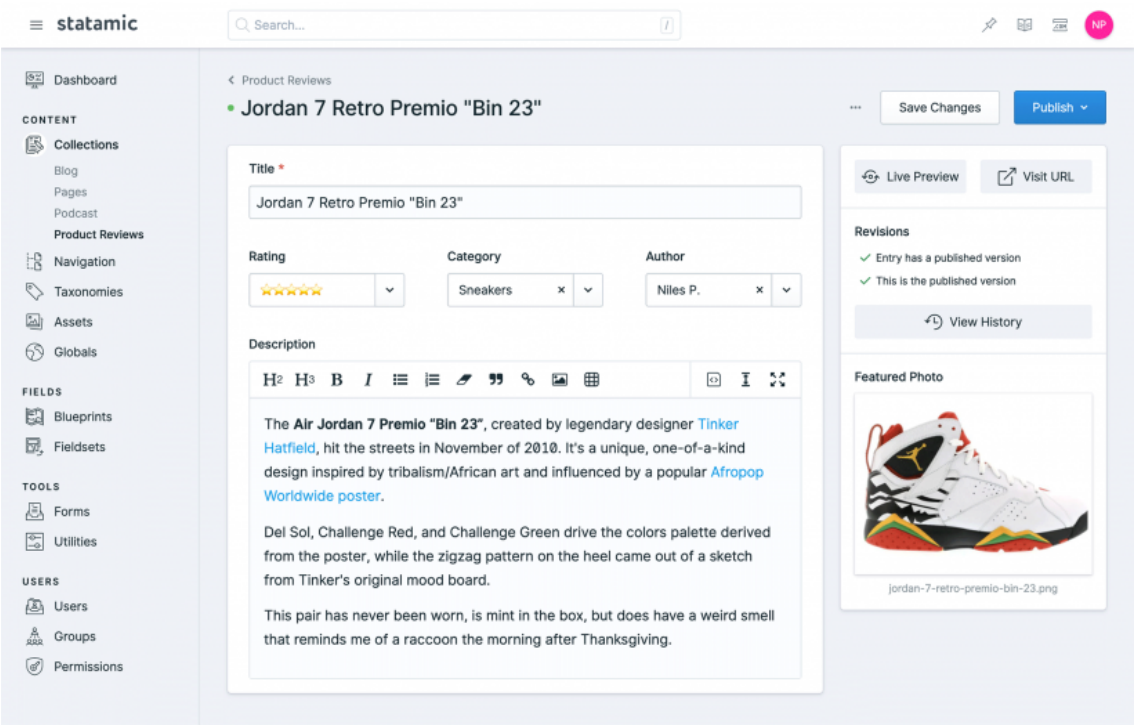


Statamic ist im Jahr 2012 als Nebenprojekt von Jack McDade entstanden und sorgte damals für frischen Wind im Flat-File-Markt. Auf Grund des großen Erfolges entwickelte sich zwischen 2014 und 2015 aus dem Projekt ein kleines Unternehmen. Mit Version 2 transformierte Statamic im Jahr 2016 zu einem Premium-System für ein etwas höheres Preis-Segment. 2020 ist Statamic 3 erschienen, das auch eine kostenfreie Version für private Webseiten anbietet.

Statamic richtet sich vornehmlich an Entwickler und Web-Agenturen, die ihr Portfolio um ein modernes und flexibles CMS ohne Datenbank erweitern wollen. Statamic bietet sich auch als datenbanklose Alternative zu dem beliebten Craft CMS an, denn beide Systeme sind aus dem Umfeld von Expression Engine entstanden und folgen ähnlichen Prinzipien.

9.1 Statamic für Content Manager

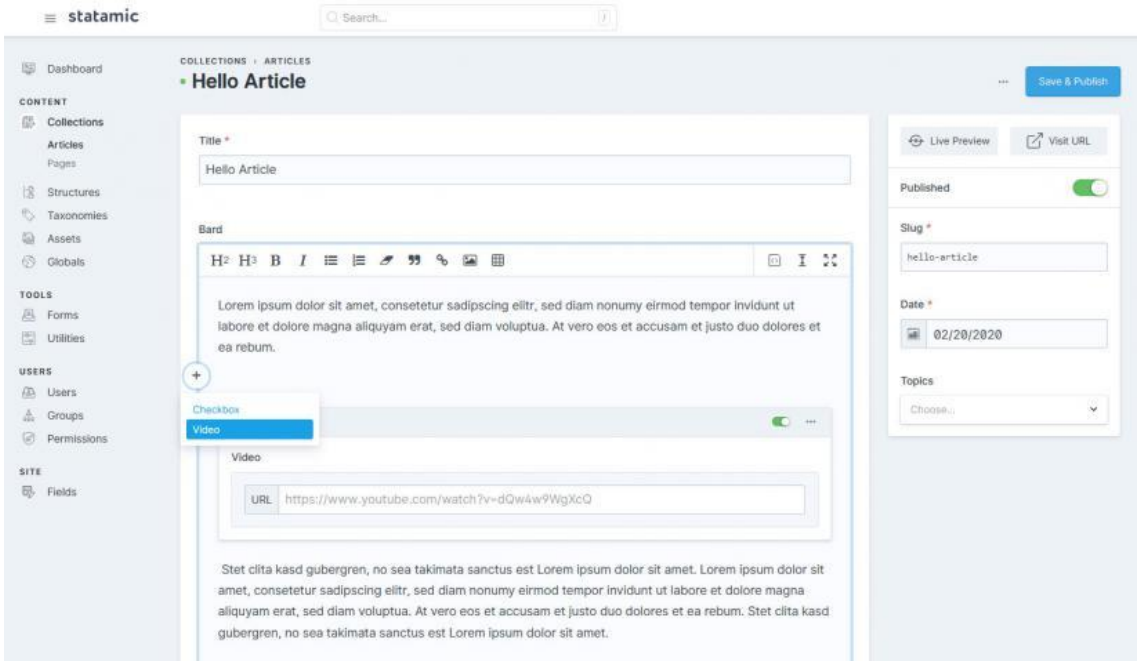
Die übersichtliche und hell gestaltete Autoren- und Administrations-Oberfläche von Statamic 3 folgt einer Drei-Teilung mit der Haupt-Navigation auf der linken Seite, den Inhalten in der Mitte und einem Context-Menü auf der rechten Seite.



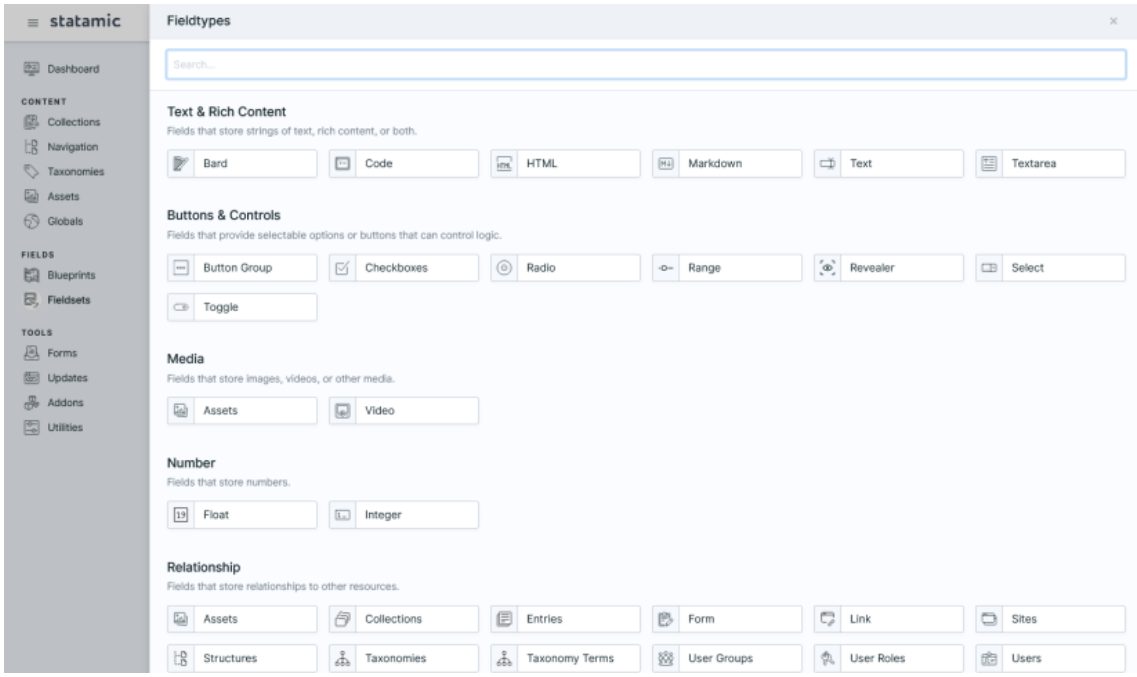
Die Haupt-Inhalte werden im Content-Menü unter dem Punkt "Collections" angelegt. Eine Collection ist lediglich ein Container, der jede Art von Eintrag ("Entries") beinhalten kann: Statische Seiten, Artikel, News, Blog-Einträge oder auch Produkt-Blätter. Über die Collection selbst werden globale Eigenschaften wie die Sortierung der Einträge, das Template und das Layout definiert. Weitere Punkte im Content-Menü sind Tags, ein ausgereifter Medien-Manager und die sogenannten "Globals", mit denen sich globale Inhalte wie ein Footer oder eine Sidebar bearbeiten lassen. Neu hinzugekommen ist der Punkt "Navigation", mit dem man die Seiten-Navigation auch unabhängig von der Inhalts-Struktur erstellen kann.

Für Fließtexte kann entweder ein Markdown-Editor mit einer Preview-Funktion definiert werden, oder ein Editor mit dem Namen "Bard". Bard startet erst einmal als gewöhnlicher HTML-Editor. Tatsächlich kann Bard jedoch zu einem kompletten Block-Editor ausgeweitet werden, sodass der Autor jede beliebigen Inhalts-Typen und Feldgruppen im Block-Stil ver-

wenden kann. Damit bricht Statamic die starre Vorgabe von Eingabeformularen vollständig auf und lässt dem Autor völlige Freiheit bei der Seitengestaltung.

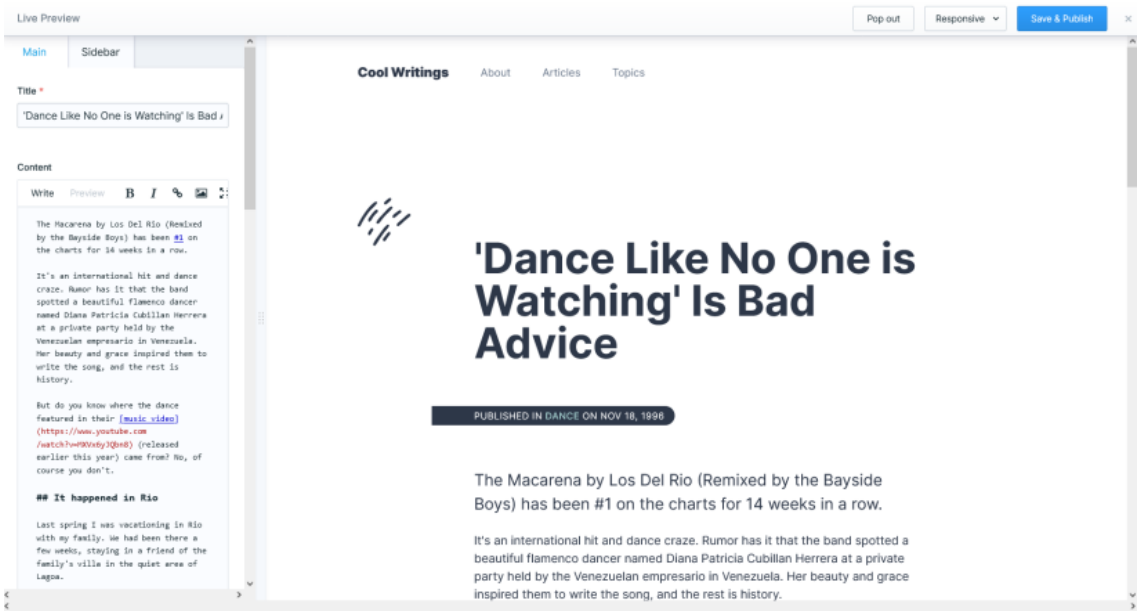


Die Konfiguration der Inhalts-Typen erfolgt bei Statamic über sogenannte "Blueprints". Dabei stehen über 40 verschiedene Eingabe-Felder zur Verfügung, mit denen der Administrator die Inhalte beliebig modellieren und strukturieren kann.



Das Prinzip der Content-Modellierung über Fieldtypes ist bei vielen modernen Flat-File-CMS verbreitet. Nirgendwo findet man das Prinzip jedoch so ausgereift und komfortabel umgesetzt wie bei Statamic. Mit den Fields lassen sich dabei nicht nur Eingabe-Oberflächen für die Autoren definieren, sondern es können auch öffentliche Formulare gestaltet werden, die der Entwickler dann über ein Template einbinden kann. Ein weiteres Feature, das Statamic von den anderen Flat-File-Systemen absetzt.

Ein anderes erwähnenswertes Feature für Autoren ist die Live-Preview, die man per Button im Editor aufrufen kann. Die Live-Preview bietet einen Split-Screen, wie er auch von Craft-CMS bekannt ist. Dabei lässt sich die Seiten-Vorschau auf der rechten Seite über die Eingabe-Leiste auf der linken Seite in Echtzeit bearbeiten.



Es gibt nicht allzu viele Wünsche, die Statamic für Autoren offen lässt. Die Experience ist mindestens auf gleicher Höhe wie bei dem etwas bekannteren Craft CMS. Es bleibt abzuwarten, ob sich die Qualität des Systems nach dem Relaunch des Spiegels nun auch hierzulande herumspricht und Statamic nach vielen Jahren in der Nische eine breitere Aufmerksamkeit erfährt.

9.2 Statamic für Entwickler

Statamic hat auch in technischer Hinsicht über die Jahre einen starken Wandel durchlaufen. Die erste Version von Statamic basierte noch auf

dem schlanken Micro-Framework Slim. Seit Version 2 setzt Statamic auf das beliebte PHP-Framework **Laravel**, das Frontend-Framework **Vue.js** und das CSS-Framework **Tailwind**. In Version 3 ist Statamic als Laravel-Package konzipiert und benötigt PHP ab der Version 7.2 (siehe die vollständigen Requirements von Statamic 3). Die technische Entwicklung macht sich auch in der Größe bemerkbar. Während Version 1 von Statamic noch mit wenigen MB auskam, kann man die Version 3 mit ca. 100 MB inklusive aller Abhängigkeiten sicher nicht mehr als leichtgewichtig bezeichnen.

Ähnlich wie bei anderen modernen Systemen erfolgt die Installation von Statamic 3 über GitHub und Composer. Wer Statamic testen will, sollte statt der Core-Version ein Starter-Kit mit einer fertigen Demo-Seite wählen, beispielsweise das Writer-Starter-Kit:

```
git clone git@github.com:statamic/statamic-starter-cool-writings.git cool-site
cd cool-site
rm -rf .git
composer install
cp .env.example .env && php artisan key:generate
```

Nach der Installation muss nur noch ein Admin-User angelegt werden:

```
php please make:user
```

Anders als bei Systemen wie Kirby führt die Installation von Statamic in einem Unterordner wie `http://localhost/statamic` zu diversen Problemen, die man im Test-Betrieb auf localhost am leichtesten Umgeht, indem man den Document Root des Apache Servers auf den Unterordner zeigen lässt.

Wie viele andere Systeme arbeitet Statamic im Hintergrund sehr umfänglich mit YAML-Konfigurationen. Über YAML werden zum Beispiel die Eingabefelder für die Autoren gespeichert. Eine einfache Field-Definition für die About-Page sieht in YAML wie folgt aus:

```
title: About
hide: true
```

```
main:
    display: Main
    fields:
      title:
        type: text
        width: 66
        localizable: true
      slug:
        type: text
        width: 33
      content:
        type: markdown
        display: Content
      fun_facts:
        type: list
        display: 'Fun Facts'
```

Statamic speichert die Inhalte, die der Autor über die Formulare eingibt, in Markdown und FrontMatter ab. Im FrontMatter werden dabei mehr oder weniger alle Inhalte abgespeichert, die nicht aus dem normalen Markdown-Content-Feld kommen:

```
---
title: About Me
template: about
fieldset: about
id: 72c016c6-cc0a-4928-b53b-3275f3f6da0a
fun_facts:
  - "I grew up in West Virginia but moved to California in the mid 90s"
  - "My life is about as organized as the $5 DVD bin at Wal-Mart"
  - "My first computer was a Commodore 64"
  - "I'm a paper cut survivor"
  - "I have a restraining order against me from J.K. Rowling but I promise it's just
a misunderstanding"
  - "I hope one day I love something the way women in commercials love yogurt"
  - "I’m not smart. I just wear glasses."
---
My name is Niles Peppertrout. I am a relatively new Park Ranger working at Redwood
```

```
Nation Park with a background in Applied Harry Potter Sciences from [Frostburg
University](http://frostburg.edu). I've always loved the outdoors but never dreamed
I'd get paid to wander them. [Join me on my adventure](/blog)!
![Me](/assets/img/me.jpg)
```

Bei der Theme-Entwicklung können die Markdown-Inhalte über das Template-Tag `{{ content }}` ausgegeben werden. Die Inhalte aus dem Front-Matter sind über die jeweiligen Namen erreichbar, in diesem Beispiel also `{{ fun_facts }}`:

```
<section class="regular">
    <header>
        <h1>{{ title }}</h1>
        {{ sub_title wrap="h2" }}
    </header>
    <article class="content">
        {{ content }}
        <h2>Fun Facts</h2>
        {{ fun_facts | ul }}
    </article>
</section>
```

Die Template-Sprache erinnert zunächst an Twig, ist aber tatsächlich eine Eigen-Entwicklung von Statamic unter dem Namen **Antler**. Es können jedoch andere Template-Sprachen wie Blade oder eben auch Twig konfiguriert werden.

Neben der Ausgabe von Variablen kann der Entwickler über Controller und ViewModels komplexere Logiken in ein Theme integrieren. Dazu kann bei dem ViewModel eine Klasse im YAML-File bzw. im Frontmatter angegeben werden, die dann zur Ausführung kommt. Das Beispiel in der Statamic-Dokumentation sieht wie folgt aus:

```
title: "A Long Article Plz Read it Mmmkay?"
view_model: App\ViewModels\ArticleStats
content:
  -
```

```
type: text
    text: # Piles of content live here
```

In der entsprechenden PHP-Klasse können beispielsweise Seitenstatistiken errechnet und zurückgegeben werden:

```php
<?php
namespace App\ViewModels;
use Statamic\View\ViewModel;
class ArticleStats extends ViewModel
{
    public function data(): array
    {
        // Combine content blocks
        $html = collect($this->cascade->get('content'))
                ->implode('text', " ");
        // Remove HTML tags
        $content = strip_tags($html);
        // Calculate stats
        $character_count = strlen($content);
        $word_count      = mb_str_word_count($content);
        $read_time       = ceil($word_count / 200);
        return [
            'character_count' => $character_count,
            'word_count'      => $word_count,
            'read_time'       => $read_time
        ];
    }
}
```

Die Version 3 von Statamic hält noch einige weitere nette Features bereit, die die Entwickler-Gemeinde erfreuen wird. Dazu gehört unter anderem die Ausgabe einer statischen HTML-Seite, mit der Statamic den Trend hin zu den Static Site Generatoren aufnimmt. Den Trend hin zu Headless-Architekturen bedient Statamic mit einer Content-API. Mit Version 3.1 kann man neben einer klassischen REST-API auch eine GraphQL-API nutzen. Version 4 von Statamic ist für Februar 2023 angekündigt.

Wer von älteren Systemen wie WordPress oder Joomla! auf Statamic umsteigt, wird sich erst einmal an die neue Logik gewöhnen müssen. Ist das Prinzip jedoch einmal verstanden, erweist sich die Entwicklung mit Statamic als unglaublich einfach, flexibel und sauber. Statamic gehört zusammen mit Craft CMS sicherlich zum State of the Art unter den CMS in der PHP-Welt und es ist beeindruckend, was das immer noch sehr kleine Team aus dem Norden von New York State über die Jahre hingezaubert hat.

9.3 Statamic für Nicht-Entwickler

Schon die Installations-Routine von Statamic über die Konsole macht klar, dass sich Statamic ausschließlich an Entwickler und Web-Agenturen richtet. Eine Nutzung durch Nicht-Entwickler wie bei WordPress ist nicht vorgesehen. Für Code-Einsteiger oder auch Liebhaber des schlanken Codes könnte Statamic mit der Laravel-Anbindung durchaus etwas überkomplex sein. Die Dokumentation von Statamic ist allerdings eine viel gepriesene und unterhaltsame Wohltat. Die Community ist vergleichsweise klein und es tummeln sich dort eher Profi-Entwickler und professionelle Dienstleister.

9.4 Roadmap

Statamic hat sich mit der Entwicklung von Version 3 viel Zeit gelassen. Entsprechend modern, komplex und vollständig präsentiert sich das System. Allerdings hat Statamic vergleichsweise spät auf den Headless-Trend reagiert und die Content-API erst in den letzten Zügen des Releases eingebaut. Die Content-API ist allerdings nur read-only. Mit Version 3.1 hat Statamic beim Thema Headless nachgeholt und neben der traditionellen REST-API auch eine neue GraphQL-API eingebaut. Mit Version 3.2 sind eher Details wie Blueprints für die Navigation verbessert worden. Auch die Roadmap lässt in naher Zukunft keine gravierenden Änderungen erkennen, was eher für die Ausgereiftheit des Systems spricht. Im Januar 2022 hat Statamic einen neuen Release-Plan veröffentlicht.

9.5 Kosten

Statamic hat sich in den letzten Jahren zunehmen als Flat-File-CMS im Premium-Segment etabliert. Nach zahlreichen Preis-Umstellungen und

Experimenten mit Lizenz-Modellen hat Statamic mit Version 3 eine überraschende Wende hingelegt: Erstmals gibt es eine kostenfreie Version für private Seiten, die allerdings ein paar Einschränkungen hat. Die Preise staffeln sich wie folgt:

- **0,-** US-Dollar. Die Lizenz gilt nur für eine private Seite, die man für sich, für Freunde oder für ein Hobby baut. Es ist nur ein Nutzer möglich.
- **259,-** US-Dollar für eine Lizenz. Die Lizenz gilt für eine Live-Seite mit unbegrenzten Updates in der Zukunft.
- **1095,-** US-Dollar für fünf Lizenzen sowie einen Discount von 25 Prozent pro zusätzlicher Lizenz im ersten Jahr.
- **Enterprise**: Individuelle Lizenzen für Enterprise-Anforderungen.

Während früher ein Account bei Statamic nötig war, um den Code zu erhalten und das CMS zu testen, reicht heute ein einfacher Composer-Befehl, um Statamic von GitHub zu installieren. Damit können Entwickler und Anwender das System ausgiebig in einer lokalen Entwicklungs-Umgebung prüfen, bevor man sich für den Kauf einer Lizenz entscheidet.

9.6 Einsatzfelder für Statamic

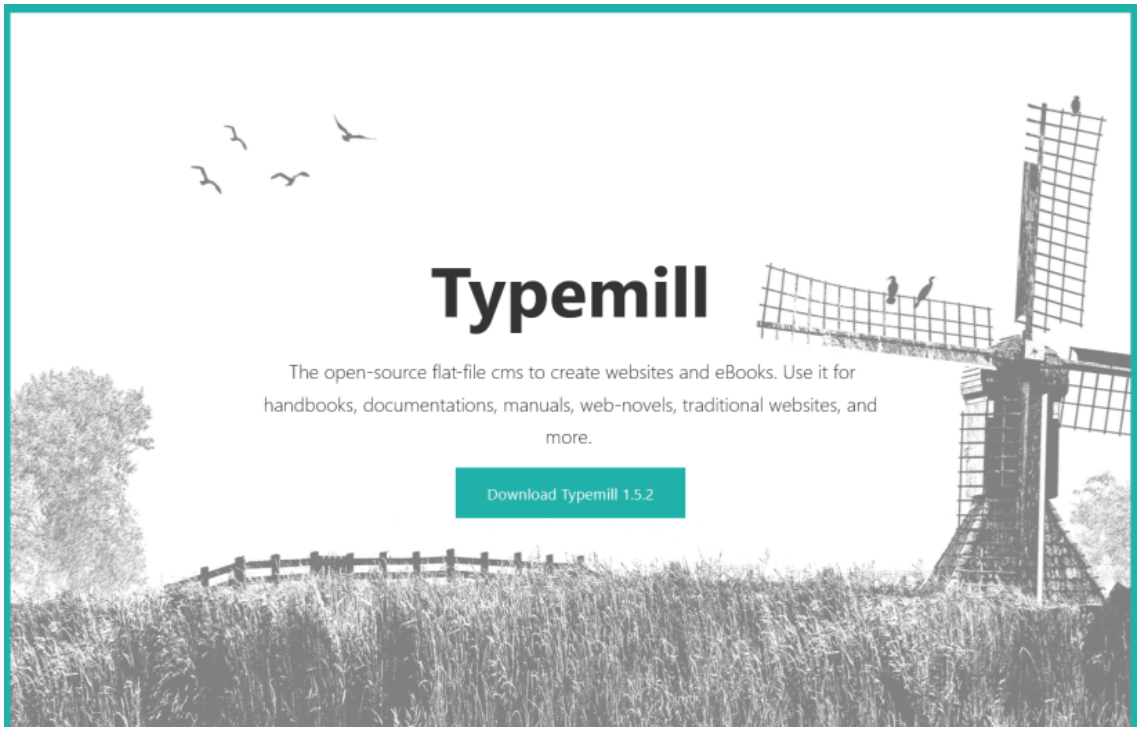
Kunden wie der Spiegel, Disney oder Asana beweisen, dass sich Statamic inzwischen zu einem sehr reifen System gemausert hat und auch komplexe Anforderungen gut meistert. Daneben dürfte der Mittelstand die Hauptzielgruppe sein, denn Statamic bietet im Vergleich zu vielen etablierten Systemen ein wesentlich einfacheres, flexibleres und auch nutzerfreundliches System an. Ein Argument dürfte sein, dass mit Statamic die Kosten für die Weiterentwicklung und für den Betrieb der Seite auf mittlere Sicht günstiger sind, denn größere Wartungen, permanente Sicherheitsupdates und eine aufwändige Erweiterung durch Plugins bleiben bei Statamic dank der flexiblen Flat-File-Architektur eher die Ausnahme.

Durch die gemeinsame Herkunft aus dem ExpressionEngine-Umfeld und zahlreiche gemeinsame Prinzipien bietet sich Statamic auch als Flat-File-Variante für das beliebte Craft-CMS an. Bei eCommerce-Projekten dürfte Craft mit seiner ausgereiften Erweiterung jedoch klar die erste Wahl bleiben.

Mit der neuen kostenfreien Variante in Version 3 eignet sich Statamic theoretisch auch für kleine Seiten und Hobby-Projekte. Insgesamt wirkt das System für einfache Webseiten im Vergleich zu anderen Flat-File-CMS jedoch zu komplex und deutlich überdimensioniert.

10. Typemill - Ein CMS für Micro-Publisher

Das Open Source System Typemill ist ein junges und schlankes Flat File CMS für informations-orientierte Webseiten und Buch-ähnliche Publikationen. Damit richtet sich Typemill vor allem an Autoren und an kleinere Unternehmen, die eine einfache Lösung für ihre Web- und Print-Veröffentlichungen im Rahmen der Dokumentation oder des Corporate Publishing suchen.

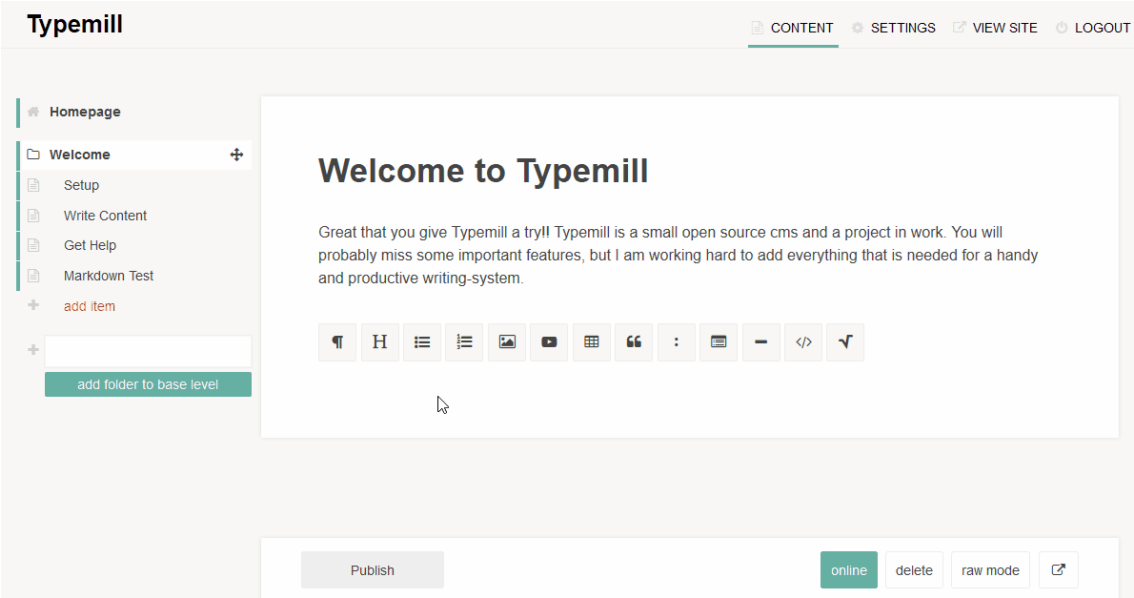


Typemill ist ursprünglich als einfaches System für den Betrieb von CMSstash entwickelt worden. Die erste Version ist im Jahr 2017 erschienen, damals noch als reines Rendering-Tool für Markdown-Dateien ohne Administrations-Oberfläche. Seitdem ist das System in regelmäßigen Releases zu einem recht vollständigen kleinen CMS herangereift, das inzwischen von mehreren Unternehmen und Organisationen eingesetzt wird. Besondere Merkmale von Typemill sind neben einem Visual Markdown Editor vor allem das eBook-Plugin. Mit der Erweiterung lassen sich aus einer Typemill-Webseite ein oder mehrere eBooks in den Formate PDF und ePUB generieren.

Haupt-Entwickler von Typemill ist Sebastian Schürmanns, der auch Autor von CMSstash ist. Um Typemill herum hat sich inzwischen eine kleine Community gebildet.

10.1 Typemill für Autoren

Typemill versteht sich als Autorenwerkzeug und legt großen Wert auf eine intuitive Bedienbarkeit. Herzstück ist der Visual Markdown Editor, der die Autoren bei der Text-Auszeichnung unterstützt. Die Stärken des Visual Editors offenbaren sich vor allem bei sperrigen Elementen wie Bildern oder auch Tabellen. Solche komplexeren Markdown-Elemente lassen sich bei Typemill in einem gewohnten WYSIWYG-Modus erstellen. Markdown-Profis können alternativ in einen schlichten Raw-Modus wechseln.

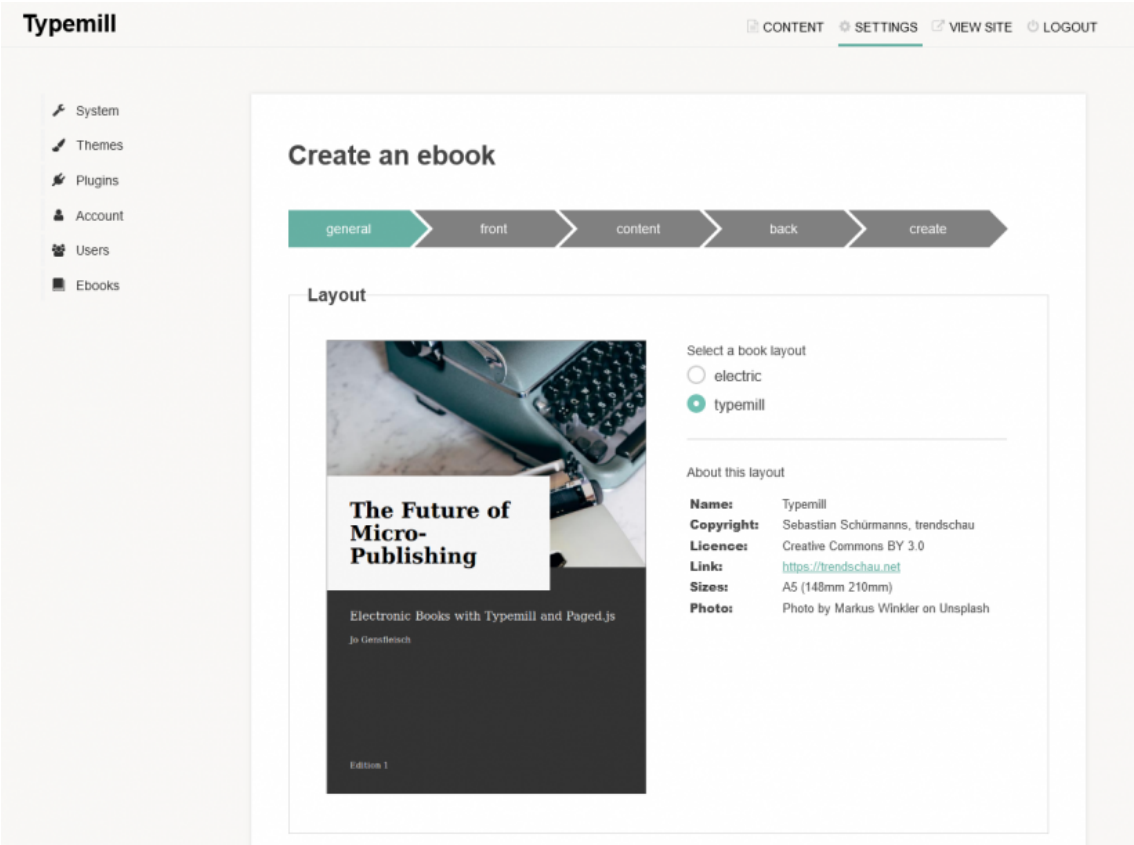


1: *Die Seitenerstellung per Drag & Drop und der Visual Markdown Editor*

Im Gegensatz zu bekannten Blog-Systemen wie WordPress geht Typemill von einer hierarchischen Seitenstruktur aus. Die Seitenstruktur lässt sich über eine interaktive Navigation auf der linken Seite per Drag & Drop erstellen. Optional können einzelne Bereiche auch nach Publikationsdatum sortiert werden, sodass einfache Blog- und News-Listen erstellt werden können. Die Möglichkeiten sind allerdings begrenzt, sodass man für klassische News- und Blog-Konzepte auf entsprechend ausgerichtete Systeme setzen sollte.

Die hierarchische Strukturierung der Inhalte ermöglicht eine nahtlose Übersetzung der Webseite in klassische Publikations-Formate, allen voran eBooks. Typemill bietet dazu ein eigenes eBook-Plugin an. Mit dem Plugin lassen sich ein oder mehrere eBooks aus einzelnen Bereichen der Webseite erstellen. Die Variante mit mehreren eBooks aus unterschiedlichen Seitenbereichen bietet sich vor allem für Publikations-Reihen an, die von

verschiedenen Autoren betreut werden. Denkbar sind Bedienungsanleitungen, Schulungsunterlagen oder auch einfache Jahresberichte.



2: *Auswahl eines Layouts im eBook Plugin*

Die Technologie zur Generierung der eBooks ist vergleichsweise experimentell (paged.js) und bringt einige Einschränkungen mit. Dafür ist die Lösung extrem leichtgewichtig und auch für Kleinst-Unternehmen nutzbar. Auch für Einzel-Autoren kann das System sinnvoll sein: Es bietet inzwischen neben einer Registrierfunktion auch eine Option für den Verkauf über Gumroad an. In Zukunft sollen Subscriptions über Plugins ohne externe Abhängigkeiten wie Gumroad möglich werden.

Neben den Publishing-Features bietet das System zahlreiche weitere Funktionen an. Nennenswert sind beispielsweise die mit Version 1.5.0 eingeführten Shortcodes, die eine Integration relativ beliebiger Inhalte und Funktionen in den Content-Bereich über Plugins erlaubt.

10.2 Typemill für Entwickler

Typemill basiert auf dem PHP-Framework Slim 3 und dem JavaScript-Framework Vue 2. Templates werden mit Twig entwickelt. Für Konfigurationen setzt das CMS wie die meisten Flat-File-Systeme auf YAML-Dateien. Derzeit wird eine Installation auf Apache-Servern mit htaccess unterstützt. Die minimale PHP-Version ist 7.4.

Ähnlich wie andere Flat-File-Systeme ist die Installation von Typemill schnell erledigt. Es müssen lediglich die Dateien auf den Apache-Server kopiert und anschließend die korrekten Rechte für Ordner und Dateien vergeben werden. Beim ersten Seiten-Aufruf legt der Nutzer einen Administrations-Account an, anschließend kann er direkt Inhalte erstellen beziehungsweise das System einrichten.

Entwickler können das System über eigene Themes individualisieren und über Plugins funktional erweitern. Typemill setzt dabei auf bekannte Technologien, sodass sich Entwickler relativ schnell zurechtfinden dürften. Das Plugin-System wird mit dem verbreiteten Event-Dispatcher von Symfonie realisiert. Themes werden wie erwähnt mit der Template-Sprache Twig erstellt. Ein typisches Twig-Snippet in einem Theme sieht wie folgt aus:

```
<article>
    <header>
        <h1>{{ title }}</h1>
    </header>
    {{ content }}
<article>
```

Die zur Verfügung stehenden Variablen und Theme-Funktionen sind in der Dokumentation für Theme-Developer aufgelistet.

Die Dokumentation enthält auch ein Tutorial für Plugin-Developer. Ein typisches Plugin nutzt in den meisten Fällen die verschiedenen Events, um Daten von Typemill während des Lifecycles zu verändern. Der Event-basierte Code sieht grundsätzlich wie folgt aus:

```php
<?php
namespace plugins\myplugin;
use \typemill\plugin;
class myplugin extends plugin
{
    public static function getSubscribedEvents()
    {
        return array(
            'onSettingsLoaded' => 'onSettingsLoaded'
        );
    }
    public function onSettingsLoaded($settings)
    {
      // do something with the $settings
    }
}
```

Neben der Event-basierten Entwicklung ist es jedoch auch möglich, eigene Routes zu erstellen oder eine eigene Middleware hinzuzufügen. Außerdem können komplett eigenständige Vue-Applikationen in die Administrations-Oberfläche integriert werden. Ein Beispiel dafür ist das eBook-Plugin, das den Nutzer in einem mehrstufigen Prozess durch die eBook-Generierung führt. Auch in anderer Hinsicht ist das eBook-Plugin beispielhaft, denn es erlaubt Entwicklern die Erstellung eigener eBook-Layouts, die ähnlich funktionieren wie Erweiterungen durch eigene Themes.

Zu den flexiblen Features von Typemill zählt außerdem das Formular-Management. Formulare werden über YAML-Dateien definiert. Dabei können die Formulare in vier Bereichen eingesetzt werden.

- Konfigurations-Formulare für Themes.
- Konfigurations-Formulare für Plugins.
- Erweiterung der Tabs für Seiten-Inhalte.
- Öffentliche Formulare.

Das Formular-Handling mit den YAML-Definitionen sind in der Dokumentation beschrieben, die Definition folgt jedoch grundsätzlich immer demselben Muster:

```
forms:
  fields:
    myfieldname:
      type: text
      label: Add a short text
    anotherfield:
      type: textarea
      label:  Add a long text
```

Interessant für Entwickler dürfte auch das Shortcode-Feature sein, denn damit lassen sich relativ leicht Inhalte in den Content-Bereich integrieren. Shortcodes können auch für den Visual Editor registriert werden, sodass Autoren ein einfaches Interface für die Integration von Shortcodes erhalten.

Typemill ist ein relativ kleines System, das sich kaum mit Statamic, Kirby oder Grav vergleichen lässt. Mit seinem Fokus auf die End-Anwender dürften eher pragmatische Entwickler Geschmack an dem System finden. Man sollte auch im Hinterkopf behalten, dass Typemill im Kern für inhaltlich abgeschlossene und begrenzte Publikationen wie Dokumentationen, Manuals, oder eben Bücher geeignet ist. Blogs, News-Webseiten, klassische Marketing-Webseiten oder visuell aufwändige Auftritte sind zwar prinzipiell auch möglich, gehören aber nicht zum Fokus des CMS.

10.3 Typemill für Nicht-Entwickler

Typemill wird relativ häufig von Nicht-Entwicklern genutzt, da die Installation sehr einfach ist und die Plugins und Themes ausreichend Anpassungsmöglichkeiten für Nicht-Entwickler bieten. Wer ein wenig CSS beherrscht, der kann die Themes auch über ein individuelles CSS-Feld anpassen. Das Standard-Theme "Cyanine" bietet darüber hinaus zahlreiche Konfigurationsmöglichkeiten hinsichtlich Schriften, Farben und auch verschiedene Inhalts-Blöcke auf der Startseite an. Typemill ist allerdings kein Baukasten wie WordPress, ab einem gewissen Punkt sind daher Entwickler-Kenntnisse für die Erstellung von individuelle Designs und Funktionserweiterungen nötig.

10.4 Kosten

Typemill ist komplett Open Source (MIT-Lizenz) und damit kostenfrei. Das gilt auch für alle bisher veröffentlichten Themes und Plugins. In Zukunft sind kostenpflichtige Premium-Erweiterungen geplant. Mit Trendschau Digital gibt es außerdem einen Implementierungs-Service, ähnlich wie Trilby Media für Grav CMS.

10.5 Einsatzfelder für Typemill

Typemill ist auf Webseiten mit strukturierten Inhalten und deren Umwandlung in buchartige Publikationen spezialisiert. Neben eBook-Projekten von Einzelpersonen wird das System inzwischen von zahlreichen kleinen und mittleren Unternehmen eingesetzt, beispielsweise für Dokumentationen, Manuals, Handbüchern, Schulungsunterlagen, Mitarbeiterhandbücher, Bedienungsanleitungen und ähnliche Projekte im Rahmen des Corporate Publishing.

11. Publikationen von CMSstash

CMSstash ist eine web-basierte Fachpublikation zu Content Management Systemen (CMS). Die Seite bietet ausführliche Markt-Übersichten zu verschiedenen CMS-Kategorien und ausführliche Reviews zu ausgewählten Systemen an.

Zusätzlich publiziert CMSstash mehrere Themen-bezogene Reports in den Formaten PDF und EPUB. Die Reports enthalten die kostenfreien und kostenpflichtigen Inhalten von CMSstash und werden jährlich aktualisiert. Alle Reports sind automatisch in dem günstigen Jahres-Abonnement von CMSstash enthalten. Alternativ können einzelne Reports über Amazon bezogen werden.

11.1 Die CMS-Reports

Die folgenden Reports sind bislang erschienen:

- Flat File CMS: Schlanke Webseiten ohne Datenbank erstellen. Februar 2022. Kostenfrei.

Die folgenden Reports befinden sich in Vorbereitung:

- Headless und Hybrid: Neue Architekturen im Überblick. In Vorbereitung. Kostenpflichtig.
- DXP und Enterprise: Die führenden Systeme im Blick. In Vorbereitung. Kostenpflichtig.
- CMS für KMUs: Ausgewählte Systeme vorgestellt. In Vorbereitung. Kostenpflichtig.
- Open Source CMS: Ein Überblick. In Vorbereitung. Kostenfrei.

11.2 Über den Autor

Sebastian Schürmanns ist Senior Projekt-Manager und Web-Entwickler mit über zehn Jahren Erfahrung. Davor war er viele Jahre lang im Verlagswesen tätig. Die großen Enterprise-CMS kennt er aus seiner Arbeit in verschiedenen Unternehmen und Agenturen. Als Web-Entwickler hat er überwiegend mit kleineren Systemen wie WordPress und Statamic gearbeitet. Seit 2017 entwickelt er das Flat-File-CMS Typemill. Ebenfalls im Jahr 2017 hat er die Fachseite CMSstash gegründet, die mit Typemill läuft

und inzwischen zu den bekanntesten Informations-Quellen zum Thema Content Management Systeme gehört.

Unterstützung bei Publishing-Projekten mit Typemill bietet Trendschau Digital.

CMSstash