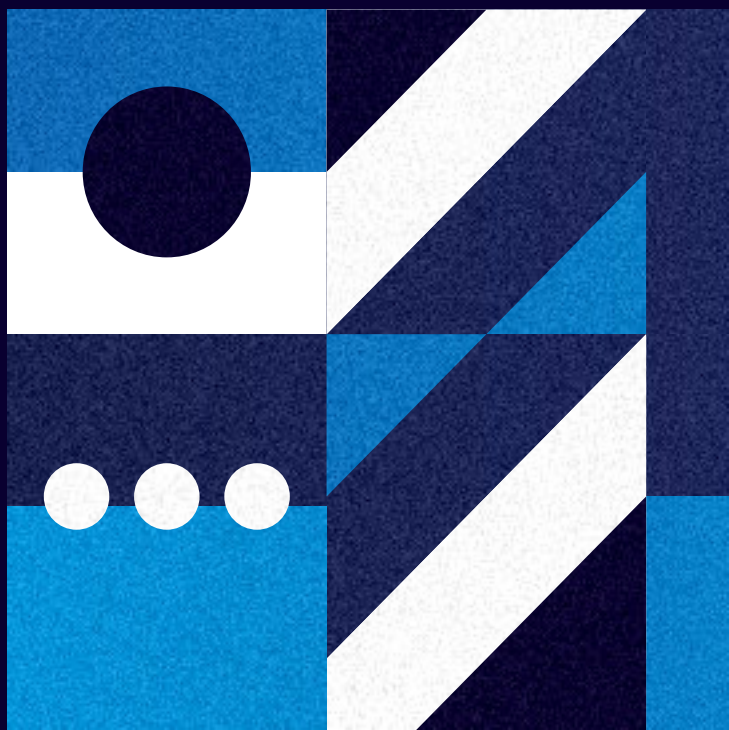
Introduction to Static Site Generators

DARREN DUBE



darrendube.com

We've all experienced it.

Websites suddenly becoming slow for no reason, loading times in the double digits, or even the dreaded *"Error establishing a database connection"*. It may come down to the hosting provider you are using, the size of your website, or even the complexity of your aesthetic design, but the crux of the matter is that:

WordPress is ageing.

Running on code first written in 2003, it was designed for a time in the evolution of the internet when waiting minutes for a website to load was not uncommon. We weren't as dependent on the internet then. But now? As a tech generation becomes more and more impatient, website loading times are becoming a crucial factor in the design process of a website, even being factored in by search engines when ranking.

So what's the solution to this problem? Static Site Generators! (surprise, surprise).

But to understand how SSGs work and how they solve WordPress' problems, we first need to understand how WordPress works, and why this is causing slow websites.

KEY TAKEAWAYS:

- WordPress works by generating HTML and CSS from website data each time a user visits the website.
- This makes WordPress websites slow.
- Static Site Generators work by prebuilding HTML and CSS so that they are ready by the time the user visits the website.
- Pros: faster, cheaper, and more secure websites when compared to WordPress.

How WordPress Works

A typical, barebones website consists of HTML (for the content and structure), CSS (for the design), and JS (for any logic or processing). For anyone starting out in Web Development, the normal inclination would be to write a separate HTML file for every page on your website. We would call this a **static website**.

Simple enough: you get easy customisation, no complex scripts to slow your site down, and a compact website that doesn't take up much space. Sure, this works for a simple website with 5 pages. But what about a blog with 50 posts? Or an e-commerce website with hundreds of products? I'm sure you can see where I'm going with this. With all these examples, sure, you can copy and paste a blog post template and edit the contents each time you want to write a new blog post, but that's taking away precious time. Time you could be spending writing blog posts.

WordPress solves this problem by (and I'm massively oversimplifying here) storing page and blog post data in a database, separate from the styles. When a user visits a page on the website, WordPress executes PHP code, retrieving data from a database, and assembling this data into HTML and CSS. This is an example of what we call a **dynamic website**.

All this code has to execute before the website has been rendered, while the user is waiting for the page to load, in a context where 40% of people will abandon a website that takes more than 3 seconds to load.

An optimised WordPress website with the best hosting and minimal assets will take about 1-2 seconds to load. But a website with cheap, low-cost hosting (the type most of us bloggers/ portfolio owners will probably use) will take about 4-5 seconds.

Static Site Generators

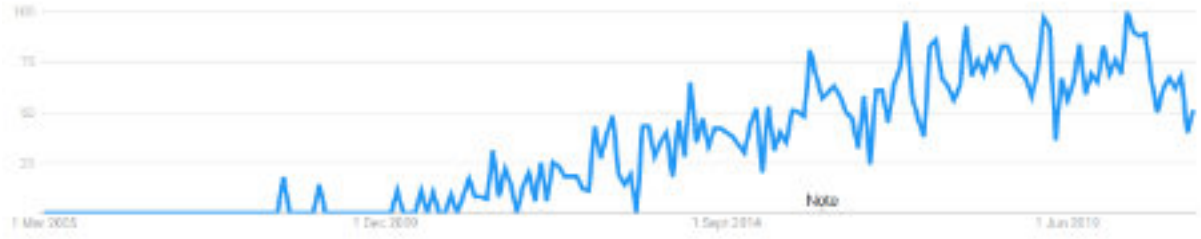
Now, instead of retrieving data from a database and assembling HTML and CSS each time a website loads, imagine if we just prebuilt this HTML and CSS before the user visited the website, and stored this instead. Instead of executing code, we deliver these files as-is when a person visits our site. This processing happens

before the user visits the site, meaning the files are ready to be served as soon as a request is made.

Think of an SSG as an application that inputs website data and outputs a folder with all the HTML, CSS, and assets.

Therefore, while with WordPress we speak of load time, with Static Site Generators, this becomes almost negligible, and we shift our focus to build time.

Static Site Generators are relatively new - below is the growth in popularity of the search term *"Static Site Generator"*:



Google Trends for 'Static Site Generator'

PROS

Performance

Because Static Site Generators prebuild websites before they are requested by users, load times are cut from several seconds to mere milliseconds!

In-site navigation is also unbelievably fast. In fact, navigating from one page to another is almost instantaneous (Try it now! This website is powered by an SSG). This is because the SSG I use (GatsbyJS) prefetches the files and assets of other pages on the website so that by the time you want to go to another page, it has everything ready!

Security

Databases, servers, and outdated software running on them are a hacker's dream. A study found that about 70% of WordPress websites had a security vulnerability!

The reason is simple - the more servers involved and the more processes and executions done by a server, the more loopholes and vulnerabilities can be found.

But by preloading our pages, we eliminate the need for any logic and work to be performed, thereby eliminating the risk of hackers injecting malicious code into these processes.

Cost

To run a self-hosted WordPress website, you need to spend on average $5-10 a month on hosting plus about $10 annually for a domain. All in all, you could easily spend over $100 a year. With Static Site Generators, this goes down to $0 (or $10 a year for a custom domain).

Because no processing power is needed, you can host your website on free hosting providers like Netlify and GitHub Pages.

Version Control

Most Static Site Generator websites are published by pushing code to a remote git repository, where it is automatically built and deployed. Apart from making the deployment process easy, this makes it easy to revert to a previous version of your site should the need arise.

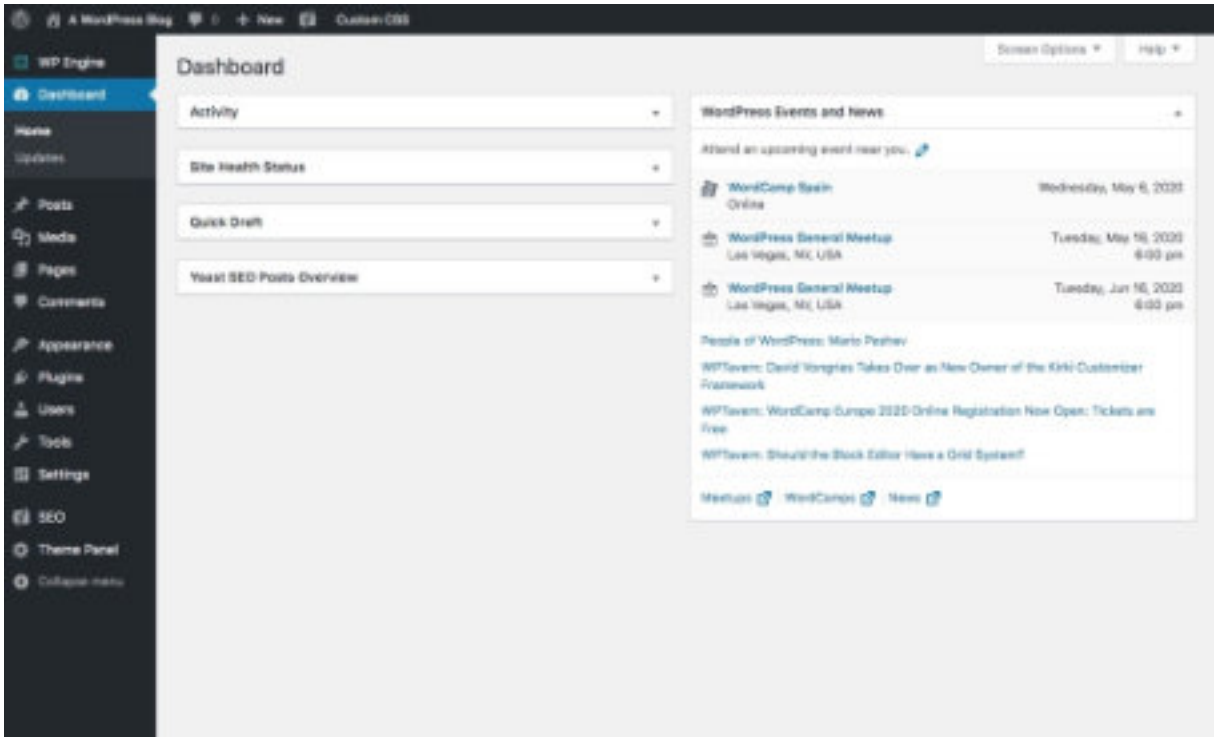
But, as with anything, Static Site Generators also have some:

CONS

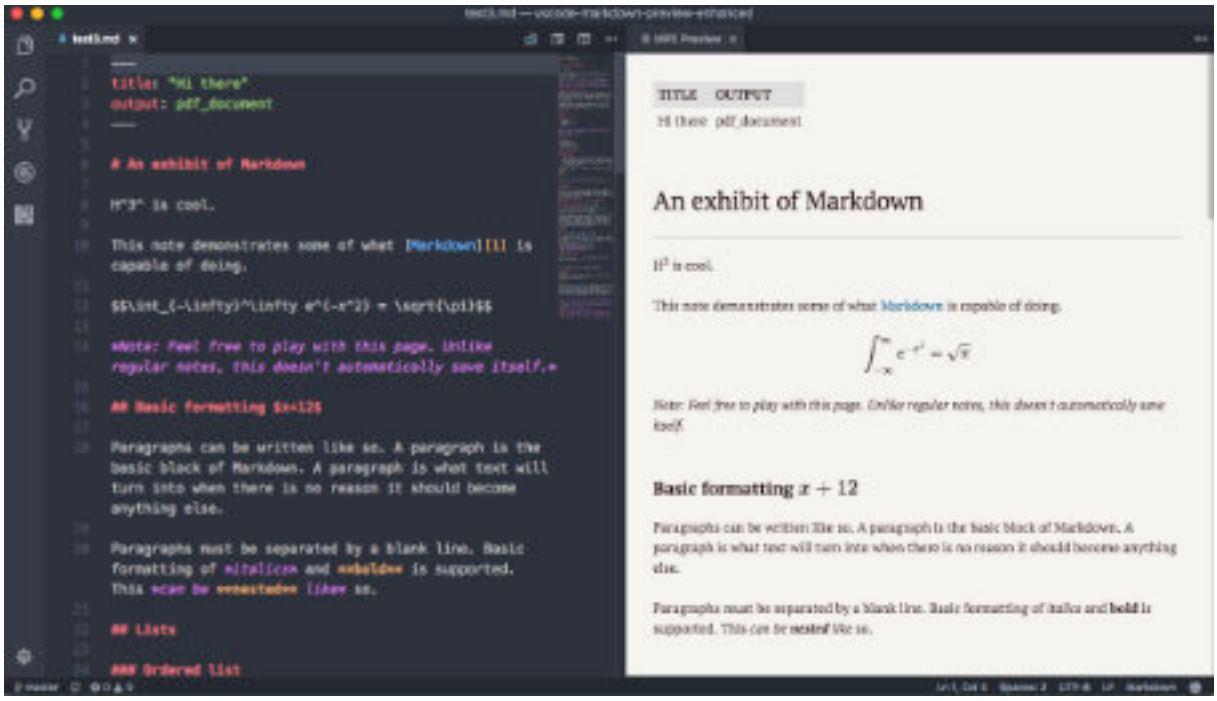
Pretty steep learning curve

(Unless you are already a programmer).

One of the reasons WordPress is so popular is its relatively shallow learning curve. You get a GUI, a dashboard, and a text editor - literally everything short of a WYSIWYG editor.



Static Site Generators require writing content in Markdown - not very intuitive for the average person.



Writing in Markdown

Add to this their relatively small community compared to WordPress, and this makes the learning curve that much steeper.

No built-in commenting

One of the many side-effects of not running on a server is that you don't get any services out of the box that require processing. That means comments, search, and most forms of contact forms will need third party services. A popular free third-party commenting service is Disqus.

Updating and Publishing the site requires tools on your computer

This means that you can only update your website from a computer that has that particular Static Site Generator installed - plus other dependencies. A WordPress site, however, can be edited from any computer with a browser and an internet connection.

Common Static Site Generators out there

GatsbyJS

Based on react and running on Javascript, GatsbyJS is very easy to transition to for existing web-developers who have already been working with Javascript. What makes it a popular choice is that it can be used to make Progressive Web Apps (PWAs), and its websites are designed to be very fast for users. The gatsby-plugin-image plugin (and its predecessor, gatsby-image) give GatsbyJS image-optimisation out of the box, with features such as resizing, blurring, and preoptimisation. GatsbyJS also has a very rich ecosystem with clear and comprehensive documentation, and a fast growing community on GitHub. This all helps make GatsbyJS sites (like this one) faster for users than websites made by other Static Site Generators.

Examples of websites powered by GatsbyJS are: Airbnb's Engineering and Data Science website, Figma, ReactJS, Hopper.

To **get started** with GatsbyJS, head to Sitepoint's tutorial, Scott Spence's tutorial, and GatsbyJS' official Quick Start page.

Hugo

Hugo's strong point is that it is fast. Built with Go, Hugo websites often take milliseconds to build (compared to 30-60s for GatsbyJS). This is good for deployment and hosting services like Netlify, which allots 200 free build minutes a month and charges for minutes over this. Go, however, is not as ubiquitous in the Web Development space as Javascript, so this may be a turn off for some.

Examples of websites powered by Hugo are: Bootstrap, Kubernetes, Ghost,and CoreUI.

To **get started** with Hugo, head to The New Stack's Hugo tutorial, and Hugo's quick start page.

Jekyll

Being one of the earlier ones (made in 2008), Jekyll popularised the concept of a Static Site Generator. Running on Ruby, its shallow learning curve and its relative maturity make it a popular choice for many web developers. While other Static Site Generators have come on the scene, Jekyll remains widely used in the Web Development space.

Examples of websites powered by Jekyll are: Ruby on Rails, and Frame AI.

To **get started** with Jekyll, head to Open Source's Jekyll tutorials, Tania Rascia's tutorial, and Jekyll's official tutorials.

Eleventy

Billing itself as a *simpler* Static Site Generator, Eleventy is a Static Site Generator for those that just want the job done. Eleventy improved on the main problems with other Static Site Generators like Hugo and Jekyll. Jekyll has been around for some time, but it is viewed as too slow for some. Hugo is fast, but it requires using Go, a programming language that is unfamiliar for most Web

Developers. Eleventy is fast, and uses Javascript, a language that most Web Developers already know.

An example of a website powered by eleventy is their website.

To **get started** with Eleventy, head to Craig Buckler's tutorial, Tatiana Mac's tutorial, and Eleventy's official getting started page.

Conclusion

You can find a more exhaustive list of Static Site Generators at staticgen.com. If you are interested in adapting your website to Static Site Generators, or making a new one altogether, visit this page which talks about JAMstack, the greater movement behind Static Site Generators, and a new approach to web development that creates faster and more secure websites.

DARREN DUBE

darrendube.com