Creating a Starter

Starters are boilerplate projects that Gatsby developers can use to set up a new site quickly. Before creating a starter, it may be helpful to peruse the Gatsby Starter Library to see what already exists and determine how your starter will provide value.

Basic requirements

For a starter to work properly, it needs to include some files (see the Hello World starter for a barebones example):

- `README.md`: instructions for how to install and configure your starter, a list of its features or structure, and any helpful tips.
- `package.json`: the "command center" for Gatsby dependencies and scripts. Find an example in the Hello World starter's package.json file.
- `gatsby-config.js`: a space to add configurable data and plugins. See Gatsby Config for more information.
- `src/pages`: a directory for page components to live, with at least one index.js file (example).
- `static`: a directory for static assets, such as a `favicon.ico` file.
- `.gitignore`: a file telling Git which resources to leave out of source control, such as the `node_modules` directory, log files, Gatsby `.cache` and `public` directories.
- `.prettierrc` *(optional)*: a configuration file for Prettier, a JavaScript linter and formatter used for Gatsby development.
- `LICENSE`: a file containing an appropriate open source license, preferably with a BSD Zero Clause License.

Your starter should also have these qualities:

- Available from a stable URL
- Open source license
- Configurable
- Fast
- Web accessible

Let's expand upon these items to prepare you for creating a winning starter.

Available from a stable URL

The Gatsby CLI allows users to install a new site with a starter using the command `gatsby new <site-name> <starter-url>`. For this to work, your starter needs to be available to download. The easiest way to accomplish this is to host your starter on GitHub or GitLab and use the publicly available repo URL, such as:

```
gatsby new my-app https://github.com/gatsbyjs/gatsby-starter-blog
```

Although the official starters live in the Gatsby repo, community members can offer their own starters from their own repos. Here's an example of installing a community starter:

```
gatsby new my-app https://github.com/netlify-templates/gatsby-starter-netlify-cms
```

Open source license

Gatsby recommends all starters use the BSD Zero Clause License(0BSD). While the MIT License is more common (and was used in Gatsby official starters for a long time), MIT's "License and copyright notice" condition make it a bad fit for starters. Starters, by design, are for people to customize and use for their own sites (open and closed source). The 0BSD has no "License and copyright notice" condition. This allows folks to use and customize the starter without reference to the original license or source. This is how most creators expect starters to be treated and how developers expect to use starters, despite many having the MIT license. Using 0BSD represents better alignment between expectations and licensing.

Configurable

Starters should utilize metadata in `gatsby-config.js` wherever possible, as this is typically the first place users will look for site configuration information. Some examples of things you could make configurable in `gatsby-config` include:

- Site title
- Author name, contact information, and bio
- Social media description

Alternatively, for starters connecting to a headless CMS, author-specific items could be pulled in to the starter from a CMS platform using a source plugin and GraphQL instead. Showing users how this is done can make your starter very helpful!

It's also highly appreciated if the built-in theming capabilities are used and a "theme" file is exposed for configuration, for example when using styled-components or a design system.

Fast

Gatsby is pretty darned fast out of the box. To make a starter that supports users in the "Gatsby way", you'll want to set up a test implementation with your starter for debugging performance. Using tools like Lighthouse and Webpagetest.org, you can evaluate the speed of your test site and make any necessary improvements to the starter source code so your users will benefit from fast defaults.

If there are areas of the starter that could be impacted by the user, it may help to add some documentation or code comments reminding them to optimize for performance (e.g. compressing images)!

Web accessible

In addition to performance, creating a starter free of accessibility issues is a wonderful way to contribute to the Gatsby ecosystem. Here are some tips for creating an inclusive, accessible starter:

- Use adequate color contrast. (This is the most common accessibility issue on the web!)
- Preserve visible keyboard focus indicators.
- Use image alt text in your examples.
- Recommend and use semantic HTML wherever possible.
- Label your form inputs.

For more accessibility help, check out the A11y Project checklist and WebAIM. You can also check out tips on creating accessible web apps heavy on client-side JavaScript.

Run your starter locally

Since starters are Gatsby projects, you can run `gatsby develop` or `gatsby build` and then `gatsby serve` in order to ensure your starter is working. If you'd like to be extra thorough and make sure the `gatsby new` command works with your starter, you can run `gatsby new project-name ../relative/path/to/your/starter`, replacing the final part of that command with the appropriate relative path.

Add your starter to the Gatsby Starter Library

To make sure your starter is easily discoverable, you are welcome (but not required) to add it to the Gatsby Starter Library. Add tags to your starter by first checking for existing ones (like `contentful`, `csv`, etc.), and adding more if needed!

Further reading:

- How to create a Gatsby Starter by Emanuel Suriano

- Introducing Gatsby Themes (including info on starters)