Using Jekyll with Pages

MAC | WINDOWS | ALL

In addition to supporting regular HTML content, GitHub Pages supports Jekyll, a simple, blog-aware static site generator. Jekyll makes it easy to create site-wide headers and footers without having to copy them across every page. It also offers some other advanced templating features.

Using Jekyll

Every GitHub Page runs through Jekyll when you push content to a specially named branch within your repository. For User Pages, use the `master` branch in your `username.github.io` repository. For Project Pages, use the `gh-pages` branch in your project's repository. Because a normal HTML site is also a valid Jekyll site, you don't have to do anything special to keep your standard HTML files unchanged. Jekyll has thorough documentation that covers its features and usage. Simply start committing Jekyll formatted files and you'll be using Jekyll in no time.

Installing Jekyll

We highly recommend installing Jekyll on your computer to preview your site and help diagnose troubled builds before publishing your site on GitHub Pages.

Jekyll is not officially supported for Windows. For more information, see "Jekyll on Windows" in the official Jekyll documentation.

To ensure your computer most closely matches the GitHub Pages settings, you can use the GitHub Pages Gem, which downloads the dependencies you need. To install Jekyll, you'll need a few things:

1. **Ruby** - Jekyll requires the Ruby language. If you have a Mac, you've most likely already got Ruby. If you open up the Terminal application, and run the command `ruby --version` you can confirm this. Your Ruby version should be at least `2.0.0`. Otherwise, follow these instructions to install Ruby version `2.0.0` or higher.

2. **Bundler** - Bundler is a package manager that makes versioning Ruby software like Jekyll a lot easier if you're going to be building GitHub Pages sites locally. If you don't already have Bundler installed, you can install it by running the command `gem install bundler`.

3. **Jekyll** - The main event. You'll want to create a file in your site's repository called `Gemfile` and add the line `gem 'github-pages'`. After that, simply run the command, `bundle install` and you're good to go. If you decided to skip step #2, you can still install Jekyll with the command `gem install github-pages`, but you may run into trouble down the line. Here's an example of a `Gemfile` you can use (placed in the root directory of your repository):

```
source 'https://rubygems.org'
gem 'github-pages'
```

Setting up Jekyll

1. Open Terminal (for Mac and Linux users) or the command prompt (for Windows users).

2. Use the command `jekyll new` followed by the name of a new directory to generate a Jekyll site scaffold.

```
$ jekyll new YOUR-JEKYLL-SITE
New jekyll site installed in /Users/YOUR-USERNAME/YOUR-JEKYLL-SITE
```

3. Use the `cd` command to navigate into your Jekyll site directory.

```
$ cd YOUR-JEKYLL-SITE
```

4. Once inside your Jekyll site directory initialize it as a Git repository.

```
$ git init
Initialized empty Git repository in /Users/YOUR-USERNAME/YOUR-JEKYLL-SITE/.git/
```

For more details, see the Jekyll QuickStart guide.

Running Jekyll

1. Use the command `git checkout` to switch to the default branch that the GitHub Pages build server uses to generate your site. The default branch you switch to depends on the type of GitHub Pages site you're building.
   - For **Project Pages sites**, switch to `gh-pages`.
   - For **User Pages or Organization Pages sites**, switch to `master`. For more information, see "User, Organization, and Project Pages".
2. Use the command `bundle exec jekyll serve` in the root of your repository to run the GitHub

Pages build server with Bundler.

```
$ bundle exec jekyll serve
```

3. Navigate to `http://localhost:4000` to see your local site.

For a full list of Jekyll commands, see the Jekyll documentation.

Keeping Jekyll up to date

Jekyll is an active open source project, and is updated frequently. As the GitHub Pages server is updated, the software on your computer may become out of date, resulting in your site appearing different locally from how it looks when published on GitHub. To keep Jekyll up to date, you can run the command `bundle update` (or if you opted out of step 2, run `gem update github-pages`).

Configuring Jekyll

You can configure most Jekyll settings by creating a `_config.yml` file.

Defaults

The following defaults are set by GitHub, which you are free to override in your `_config.yml` file:

```
highlighter: pygments
github: [Repository metadata]
```

For the content of the repository metadata object, see repository metadata on GitHub Pages.

Configuration Overrides

We override the following `_config.yml` values, which you are unable to configure:

```
safe: true
lsi: false
source: your top-level directory
```

Keep in mind that if you change the `source` setting, your pages may not build correctly. GitHub Pages only considers source files in the top-level directory of a repository.

Frontmatter is required

Jekyll requires that Markdown files have *front-matter* defined at the top of every file. Front-matter is just a set of metadata, delineated by three dashes:

```
---
title: This is my title
layout: post
---

Here is my page.
```

If you like, you can choose to omit front-matter from your file, but you'll still need to make the triple-dashes:

```
---
```

```
---

Here is my page.
```

If your file is within the *_posts* directory, you can omit the dashes entirely.

For more information, check out the Jekyll docs.

Troubleshooting

If your Jekyll site is not rendering properly after you push it to GitHub, it's useful to run Jekyll locally so you can see any parsing errors.

To ensure your local development environment is using the same version of Jekyll and its dependencies as GitHub Pages, you can periodically run the command `gem update github-pages` (or `bundle update github-pages` if using Bundler) once Jekyll is installed. For more information, see the GitHub Pages Gem repository.

If your page isn't building after you push to GitHub, see "Troubleshooting GitHub Pages build failures".

> If you are having issues with your Jekyll Pages, make sure you are not using categories that are named the same as another project, as this could cause path conflicts. For example: if you have a blog post named 'resume' in your User Page repository and a project named 'resume' with a `gh-pages` branch, they will conflict with each other.

Turning Jekyll off

You can completely opt out of Jekyll processing by creating a file named `.nojekyll` in the root of your Page repository and pushing that file to GitHub. This should only be necessary if your site uses directories that begin with an underscore, as Jekyll sees these as special directories and does not copy them to the final destination.

Contributing

If there's a feature you wish that Jekyll had, feel free to fork it and send a pull request. We're happy to accept user contributions.

Contact a human