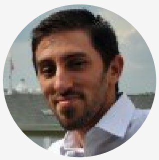
Jekyll Plugins on GitHub



(http://www.sitepoint.com/author/dlyons/)

David Lyons (http://www.sitepoint.com/author/dlyons/)

Published March 19, 2014

Tweet (Https://Twitter.Com/Share?Text=Jekyll+Plugins+On+GitHub&Via=Sitepointdotcom)

Subscribe (Https://Confirmsubscription.Com/H/Y/1FD5B523FA48AA2B)

If you use Ruby, you've most likely heard of GitHub (http://github.com), the amazing and beloved DVC repository. You've also probably heard of Jekyll (http://jekyllrb.com), the blog aware static site generator. What you might not know is that GitHub lets you host your Jekyll site on their GitHub Pages (http://pages.github.com/) service for free! Like any other free service, there is a catch. It's not ads, bandwidth limitations, or hidden fees though, it's the utilization of Jekyll plugins.

If you're not familiar with Jekyll or GitHubPages, check out the Quick-start guide (http://jekyllrb.com/docs/quickstart/) and their introduction to deploying sites to GitHub Pages (http://jekyllrb.com/docs/github-pages/).

Jekyll by default does more than enough to let you hit the ground running. You can write your posts in markdown, use the simple templating engine Liquid (//docs.shopify.com/themes/liquid-basics) to make your site look handsome, then generate static pages with `jekyll build` and *boom* you've got HTML files and assets ready to be deployed. Because these are static HTML files (and possibly some JavaScript) you can host them almost anywhere

1. Personal server
2. Shared hosting
3. Public dropbox folder
4. GitHub Pages (the obvious correct choice)

4. GitHub Pages (the obvious correct choice)

1 and 2 take time and money to maintain, and trying to host a site on Dropbox is probably not a good idea for anything more than testing. The GitHub Pages service (which is provided so users, organizations and projects can host sites about themselves, their organizations, and projects) runs on Jekyll; a tool they built and maintain because they dog food everything.

This is where the plugins come in (or rather, don't). It is pretty easy to get going with GitHub Pages even, if you're pretty new to Ruby and Git. But what they don't tell you is, when you `git push` your Jekyll repo to your user/gh-pages repo on Github and it builds your `_site` directory for public consumption **it doesn't run any Jekyll plugins that aren't a part of the official repo.**

This is because when Github creates your pages it does so with `jekyll build --safe`. The `--safe` parameter prevents any arbitrary code from being executed at build time, because GitHub obviously does not want you executing arbitrary code on their servers.

Fun fact: the liquid template engine is used in Jekyll specifically because it is safe and doesn't allow arbitrary code injection. This makes it safely "user editable".

There are lots of great Jekyll plugins, like archive and rss generators, that add highly desirable functionality. However, because they are not core Jekyll plugins, they will not run on GitHub Pages. Of course, you'll still want this functionality in your site, so how can we get around this (completely legitimate) fear of arbitrary code execution? There are a few different ways, from the drop-dead simple to the highly automated (and complicated). The good news is, the simple way will work for anyone.

GitHub Pages hosts your site, but it doesn't *technically* have to have been generated by Jekyll. This technicality means you can generate your site locally and then push the static files to GitHub. Github won't find anything to compile via a `jekyll build` so you end up with a repo of your site instead of a repo of your Jekyll files. You can make sure GitHub doesn't try to build your site by including a `.nojekyll` file in your repo.

The Simple Way

This is actually the method I used at first because I realized my plugins weren't working on GitHub Pages and it was far and away the easiest and quickest fix. That also means it requires a little bit of manual work each time you want to update your site. Here is what you need to do:

1. Rename the directory with your Jekyll site to `compiled_site`
2. Create a new directory (outside of the `compiled_site` directory) and name it `jekyll_site`
3. Copy all the contents of `compiled_site` into `jekyll_site`
4. Run `jekyll build` in your new `jekyll_site` directory
5. In your `compiled_site` directory run `git rm -r` to remove all files
6. Copy the entire contents of `jekyll_site/_site` into `compiled_site`
7. Create a `.nojekyll` file in `compiled_site`
8. Run `git add --all :/` in `compiled_site` to add all the static files into the git repo and remove the old ones
9. Run `git commit -m "Built site locally"`
10. Run `git push` to update GitHub

If you need plugins working on your site **now**, there isn't a faster way to get the content into GitHub Pages then this. Unfortunately, every time you write a new post, tweak your theme, or make any other change, you'll need to do a manual copy of the files built in your `_site` directory into the actual GitHub directory.

[illegible]

Manual file copying? The Jekyll motto is “blog like a hacker” and this manual file management doesn’t seem very hacker-esque. Fortunately, we have rake . We can make a simple rake task to automate the process outlined above. If your Jekyll blog isn’t already using a Rakefile , create one and put it into the jekyll_site directory created earlier. Add the following into this Rakefile:

```
GH_PAGES_DIR = "compiled_site"

desc "Build Jekyll site and copy files"
task :build do
  system "jekyll build"
  system "rm -r ../#{GH_PAGES_DIR}/*" unless Dir['../#{GH_PAGES_DIR}/*'].empty?
  system "cp -r _site/* ../#{GH_PAGES_DIR}/"
end
```

Before you run your Rakefile make sure to exclude it from building into the _site/ by adding an exclude line to your _config.yml . The exclude line requires an array, even if there is just one file.

```
exclude: [Rakefile]
```

Now instead of using jekyll build to get your site ready you can use rake build . This Rakefile will

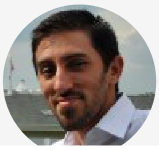
- run jekyll build for you
- empty your destination folder if it has contents (in case you deleted files/posts/assets since your last site build)
- copy the contents of the newly built jekyll_site/_site directory to your compiled_site git repo directory

Now you can cd into compiled_site and run

```
git add --all
git commit -m "Updated my site!"
git push origin master
```

You can expand your Rakefile to handle the git add , git commit , and git push commands as well, if you want to get even fancier. I eventually expanded my Rakefile to manage the whole thing for me via git branches instead of multiple directories (a post for another day!) However, with this setup you can use all the Jekyll plugins you want, build your site locally, and still host your site on the fantastic (and free!) GitHub Pages easily.

Are you blogging like a hacker? Do you use GitHub pages to host those Jekyll files? Let me know in the comments what automations you build into your process, and the Jekyll plugins you’ll be able to use now.



(http://www.sitepoint.com/author/dlyons/)

David Lyons (http://www.sitepoint.com/author/dlyons/)

David is an Instructional Technologist happily hacking away in higher ed. When not behind a keyboard he can be found behind a controller, handlebars, or in the mountains.

(https://twitter.com/lyonsinbeta) (https://github.com/lyonsinbeta)
(http://google.com/+DavidLyons)

You might also like

Getting Started with Jekyll Collections (http://www.sitepoint.com/getting-started-jekyll-collections/)

Book: Jump Start Rails (https://learnable.com/books/jump-start-rails/?utm_source=sitepoint&utm_medium=related-items&utm_content=getting-started-ruby)

Getting Started with Vim (http://www.sitepoint.com/getting-started-vim/)

4 Comments SitePoint Login ▾

Sort by Best ▾ Share Favorite ★

Join the discussion…

wejrowski • 8 months ago

An alternate solution I'm doing:
- .gitignore _site in master
- git init in /_site
- git add remote url of the same repo
- Then create gh-pages branch and just stay on that.

Then all you really need to do whenever you want to publish is jekyll build, cd into _site, commit and push. A bit simpler than doing a bunch of copying.

^ | v • Reply • Share ›

Andrew Tongen → wejrowski • 8 months ago

You can even move the .git directory out of _site so it doesn't get in jekyll's way: http://www.andrewtongen.net/20...

^ | v • Reply • Share ›

Harry Moreno • 9 months ago

I'm using the yeoman generator jekyllrb which let's you use bower and grunt to handle your Jekyll project. My blog is at morenoh149.Github.io

^ | v • Reply • Share ›

Fernando Basso • 9 months ago

Great tip! I set up a github page but haven't had time to play with it yet. Bookmarked the post.

^ | v • Reply • Share ›

Subscribe Add Disqus to your site Privacy DISQUS

About

About us (/about-us/)
Advertise (/advertising)
Press Room (/press)
Legals (/legals)
Feedback (mailto:feedback@sitepoint.com)
Write for Us (/write-for-us)

Our Sites

Learnable (https://learnable.com)
Reference (http://reference.sitepoint.com)
Web Foundations (/web-foundations/)

Connect

(/feed) (/newsletter) (https://www.facebook.com/sitepoint) (http://twitter.com/sitepointdotcom) (https://plus.google.com/+sitepoint)