

Hugo

[Official Documentation](#)

[Hugo Docker Image](#) that is not maintained by Hugo, but it [is recommended by the official documentation](#)

Nice blog post on using Hugo with Docker [nodinrogers - Hugo in Docker](#)

Site Setup

```
sudo apt install docker-compose
mkdir /Code/Docker/hugo && cd /Code/Docker/hugo
vim docker-compose.yml
```

Paste the following into the `docker-compose.yml` -

```
server:
  image: klakegg/hugo:0.93.2
  command: server
  volumes:
    - "./src"
  ports:
    - "1313:1313"
```

Now we try to start the container, but there's an error; This is because I had not created a site previously, so hugo has nothing to serve.

```
docker-compose up
Starting hugo_server_1 ... done
Attaching to hugo_server_1
server_1 | Error: Unable to locate config file or config directory. Perhaps you need to
server_1 | create a new site.
server_1 | Run `hugo help new` for details.
server_1 |
hugo_server_1 exited with code 255
```

I don't have hugo installed locally, and one of the main attractions to Docker for me is avoiding installing service dependencies to my local system. So, I will use the following command to access

the `hugo` CLI using Docker, which will generate a new site for us. Note that the `/src/site` directory is relative to the container, and not our local system. This command will bind our CWD to the container's `/src` directory (`$(pwd):/src`), and then generate the new site at `/src/site` on the container. Since this is a bound directory, the generated site files will be available on our local system.

```
docker run --rm -it -v $(pwd):/src klakegg/hugo:0.93.2 new site /src/site
```

Congratulations! Your new Hugo site is created in `/src/site`.

Just a few more steps and you're ready to go:

1. Download a theme into the same-named folder.
Choose a theme from <https://themes.gohugo.io/> or create your own with the "hugo new theme <THEMENAME>" command.
2. Perhaps you want to add some content. You can add single files with "hugo new <SECTIONNAME>/<FILENAME>.<FORMAT>".
3. Start the built-in live server via "hugo server".

Visit <https://gohugo.io/> for quickstart guide and full documentation.

```
kapper@xps:~/Code/Docker/hugo/site$ tree -L 2
```

```
.
├─ archetypes
│   └─ default.md
├─ config.toml
├─ content
├─ data
├─ layouts
├─ public
│   └─ categories
│   └─ index.xml
│   └─ sitemap.xml
│   └─ tags
├─ resources
│   └─ _gen
├─ static
└─ themes
```

But when we run `docker-compose up` we still get the same error.

```
docker-compose
up

Starting hugo_server_1 ... done
Attaching to hugo_server_1
server_1 | Error: Unable to locate config file or config directory. Perhaps you need to
server_1 | create a new site.
server_1 | Run `hugo help new` for details.
server_1 |
hugo_server_1 exited with code 255
```

This is because the hugo container expects the `/src` volume to be bound to the directory that contains our site. We previously generated our site within the `site` subdirectory, so we need to modify our `docker-compose.yml` to reflect this under the `volumes` section -

```
server:
  image: klakegg/hugo:0.93.2
  command: server
  volumes:
    - "./site:/src"
  ports:
    - "1313:1313"
```

Now we can start our hugo site, and visit it in our browser at `localhost:1313`, assuming you are using the default ports above.

```
docker-compose up

Recreating hugo_server_1 ... done
Attaching to hugo_server_1
server_1 | Start building sites ...
server_1 | hugo v0.93.2-643B5AE9 linux/amd64 BuildDate=2022-03-04T12:21:49Z
server_1 | VendorInfo=gohugoio
server_1 | WARN 2022/06/03 14:58:20 found no layout file for "HTML" for kind "home": You
server_1 | should create a template file which matches Hugo Layouts Lookup Rules for this combination.
server_1 | WARN 2022/06/03 14:58:20 found no layout file for "HTML" for kind "taxonomy": You
server_1 | should create a template file which matches Hugo Layouts Lookup Rules for this combination.
server_1 | WARN 2022/06/03 14:58:20 found no layout file for "HTML" for kind "taxonomy": You
server_1 | should create a template file which matches Hugo Layouts Lookup Rules for this combination.
server_1 |
```

```

server_1 |                               | EN
server_1 | -----+-----
server_1 | Pages                | 3
server_1 | Paginator pages      | 0
server_1 | Non-page files       | 0
server_1 | Static files         | 0
server_1 | Processed images     | 0
server_1 | Aliases              | 0
server_1 | Sitemaps             | 1
server_1 | Cleaned              | 0
server_1 |
server_1 | Built in 1 ms
server_1 | Watching for changes in /src/{archetypes,content,data,layouts,static}
server_1 | Watching for config changes in /src/config.toml
server_1 | Environment: "DEV"
server_1 | Serving pages from memory
server_1 | Running in Fast Render Mode. For full rebuilds on change: hugo server --
server_1 | disableFastRender
server_1 | Web Server is available at http://localhost:1313/ (bind address 0.0.0.0)
server_1 | Press Ctrl+C to stop

```

At this point, the `-d` flag will be useful to us since we no longer need the debug output from the container. If you plan to have the container up for a while and won't be using this output, detach the container from your session -

```
docker-compose up -d
```

```
Starting hugo_server_1 ... done
```

Git Repository

You can track your entire site as a Git repository, if you want to. To do this, I would run the following commands, where the root of the repository ends up being the directory that contains the `docker-compose.yml` file. This bundles everything together so we can easily deploy the site on a new server if needed.

```
git init
```

```
Initialized empty Git repository in /home/kapper/Code/Docker/hugo/.git/
```

```
git status

On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    docker-compose.yml
    site/

nothing added to commit but untracked files present (use "git add" to track)
```

This output produces a warning telling us that we have an embedded repository, because I ran `git clone git@github.com:StaticMania/portio-hugo.git site/themes/portio` before initializing the repository. I wanted to track some test theme in the initial commit, but my hugo site is still not configured to use any theme. This way I can always restore the original configurations if I ever need them, or I could even just reference them on Github by looking in the repository history.

```
git add .
warning: adding embedded git repository: site/themes/portio
hint: You've added another git repository inside your current repository.
hint: Clones of the outer repository will not contain the contents of
hint: the embedded repository and will not know how to obtain it.
hint: If you meant to add a submodule, use:
hint:
hint:   git submodule add <url> site/themes/portio
hint:
hint: If you added this path by mistake, you can remove it from the
hint: index with:
hint:
hint:   git rm --cached site/themes/portio
hint:
hint: See "git help submodule" for more information.
```

The above output indicates that we could run `git submodule add git@github.com:StaticMania/portio-hugo.git site/themes/portio` to track the Portio theme as a submodule of our repository. For me, I won't - simply because Portio was the theme I used for my first attempt at configuring a Hugo theme. I plan to shop around for a theme that I like, and when I settle on one I will likely add that as a submodule.

Now, before we install our theme we could make a commit to track the site as it was newly generated by Hugo. Simply run `git commit` and type your initial commit message, then continue to work. You don't need to push to a remote until you're ready. I'll probably just track this project locally for a while and push it to a repo only when I go to deploy it. This way I will retain my repository history and have access to useful git commands to restore files or view diffs.

Themes

To start, let's look at the [Portio Hugo theme](#) on GitHub. This repo provides these [install instructions](#) which we will use to configure the theme below

```
cd themes/  
pwd  
~/Code/Docker/hugo/site/themes
```

```
git clone git@github.com:StaticMania/portio-hugo.git portio  
fatal: could not create work tree dir 'portio': Permission denied
```

To fix the above error, change ownership of these files to your current user -

```
kapper@xps:~/Code/Docker/hugo/site/themes$ cd ../../  
kapper@xps:~/Code/Docker/hugo$ sudo chown -R kapper:kapper site/  
kapper@xps:~/Code/Docker/hugo$ ll  
total 16  
drwxrwxr-x  3 kapper kapper 4096 Jun  3 11:29 ./  
drwxrwxr-x  3 kapper kapper 4096 Jun  3 10:13 ../  
-rw-rw-r--  1 kapper kapper  114 Jun  3 10:58 docker-compose.yml  
drwxr-xr-x 10 kapper kapper 4096 Jun  3 10:57 site/
```

```
git clone git@github.com:StaticMania/portio-hugo.git portio  
  
Cloning into 'portio'...  
remote: Enumerating objects: 1377, done.  
remote: Total 1377 (delta 0), reused 0 (delta 0), pack-reused 1377  
Receiving objects: 100% (1377/1377), 8.63 MiB | 16.73 MiB/s, done.  
Resolving deltas: 100% (502/502), done.
```

```
cp -r site/themes/portio-hugo/exampleSite/* site/
```

Now we can restart our docker container and see the theme in the browser!

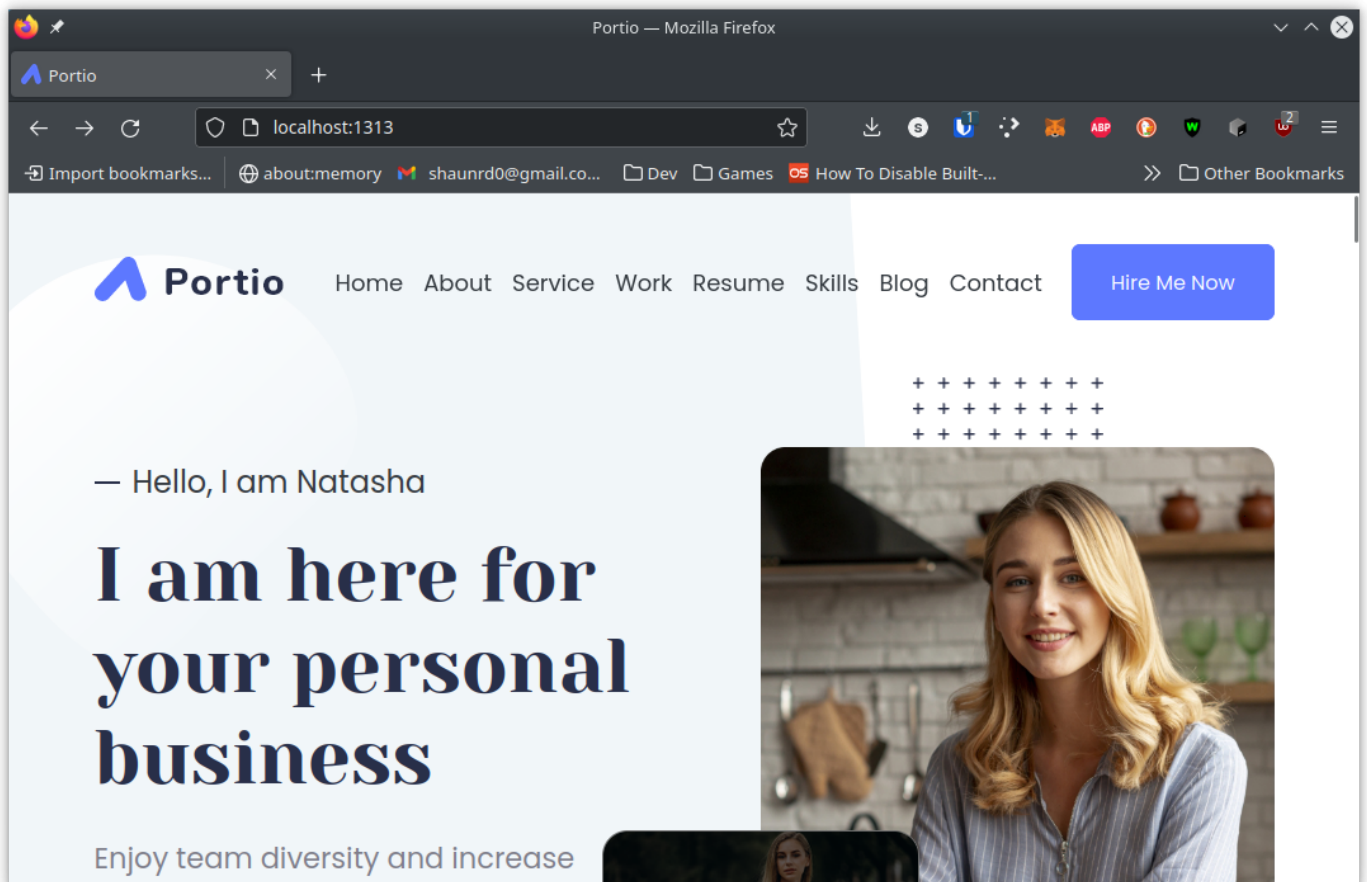
```
docker-compose down && docker-compose up -d`
```

Unfortunately at this point the theme did not render correctly for me. This was because I was targeting the `klakegg/hugo:0.93.2` docker image in my `docker-compose.yml`, and this theme requires Hugo Extended to render SASS/SCSS styling. So, we modify the `docker-compose.yml` to contain the below `ext-ubuntu` image instead -

```
server:
  image: klakegg/hugo:ext-ubuntu
  command: server
  volumes:
    - "./site:/src"
  ports:
    - "1313:1313"
```

And when we restart the container with the following command once more, everything is working and the theme is rendered correctly!

```
docker-compose down && docker-compose up -d`
```



Creating Posts

In your `config.toml`, set the following option -

```
contentDir = 'content/'
```

Then we can create our `posts` directory, and run `hugo new content/posts/post-name.md` to create a new post.

Revision #2

Created 2022-06-03 09:03:34 UTC by Shaun Reed

Updated 2022-06-06 08:14:07 UTC by Shaun Reed