Hugo Static site generator

START HERE

`hugo new site <name>`
Create a project

`hugo config`
Show merged config

`hugo new content posts/<name>.md`
Add a page

`hugo mod get -u`
Update modules

`hugo server -D`
Preview with drafts

`hugo list drafts`
Find unpublished pages

`hugo`
Build into public/

`hugo --minify`
Build, smaller output

content/ + hugo.toml Markdown and config → `hugo server` → **Live preview** localhost:1313, reloads → `hugo` → **public/** static HTML, CSS, JS → `deploy` / `upload` → **Web host** any static host

CONCEPTS

Hugo is one binary: content in content/, layouts in a theme or layouts/, config in hugo.toml. The build writes plain files to public/.

Pages with draft: true, a future date or a past expiryDate are skipped unless you pass -D, -F or -E.

Hugo comes in standard, extended and extended/deploy editions. Some themes need extended for Sass; hugo version shows which you have.

PROJECT

`hugo new site <name>`
Create a new site skeleton

`hugo new site <name> --format yaml`
Use YAML (or json) config

`hugo new site . --force`
Initialise in a non-empty folder

`hugo new theme <name>`
Scaffold a theme in themes/

`git submodule add <url> themes/<name>`
Install a theme as submodule
Then set theme = '<name>' in hugo.toml.

`hugo version`
Show the installed version

PROJECT (cont.)

`hugo env`
Show version and Go/OS info

`hugo config`
Print the merged site config

`hugo config --format yaml`
Print config as yaml or json

`hugo config mounts`
Print the file mounts

CONTENT

`hugo new content <path>`
Create a page from an archetype

`hugo new content posts/<name>.md`
Add a post

`hugo new content -k <kind> <path>`
Force an archetype kind

`hugo new content --editor <ed> <path>`
Open it in an editor

`hugo list drafts`
List draft pages

`hugo list future`
List pages with a future date

`hugo list expired`
List expired pages

`hugo list published`
List pages that will be built

`hugo list all`
List all pages, any state

CONTENT (cont.)

`hugo convert toYAML`
Convert front matter to YAML
Also toTOML and toJSON. Writes in place unless -o is set; commit first.

`hugo convert toTOML -o <dir>`
Convert into another directory

DEV SERVER

`hugo server`
Serve and live reload

`hugo server -D`
Include drafts

`hugo server -F`
Include future-dated pages

`hugo server -p <port>`
Listen on another port

`hugo server --bind 0.0.0.0`
Reachable from other devices

`hugo server --bind 0.0.0.0 -b http://<ip>:1313`
Phone test with matching baseURL

`hugo server -N`
Jump to the changed page

`hugo server -O`
Open the browser on start

`hugo server --disableFastRender`
Full rebuild on each change

DEV SERVER (cont.)

`hugo server --navigateToChanged --noHTTPCache`
Avoid stale browser cache

`hugo server --renderToMemory`
Do not write to disk

`hugo server --poll 700ms`
Poll files (network drives)

`hugo server --tlsAuto`
Serve HTTPS locally
Run hugo server trust once to install the local CA.

BUILD

`hugo`
Build the site into public/

`hugo --minify`
Minify HTML, CSS, JS, XML

`hugo -d <dir>`
Write to another directory

`hugo -b <url>`
Set the base URL

`hugo -s <dir>`
Build a site elsewhere

`hugo --gc`
Remove unused cache files

`hugo --cleanDestinationDir`
Remove stale files from output

`hugo -D`
Build drafts too

`hugo -e <env>`
Pick the environment

BUILD (cont.)

`hugo --enableGitInfo`
Use git for lastmod and author

`hugo --logLevel info`
Set log verbosity

`hugo --quiet`
Hide non-error output

`hugo --panicOnWarning`
Fail on the first warning

`hugo --templateMetrics`
Show template timings

`hugo --printPathWarnings`
Warn on duplicate target paths

`hugo --printUnusedTemplates`
Warn about unused templates

MODULES

`hugo mod init <path>`
Make the site a Hugo module

`hugo mod get <path>`
Add or fetch a module

`hugo mod get <path>@<ver>`
Pin a module version

`hugo mod get -u`
Update all modules

`hugo mod tidy`
Prune go.mod and go.sum

Modules need Go and git installed. Import one with [[module.imports]] path = '<path>' in the config.

DEPLOY AND EXTRAS

`hugo deploy`
Upload public/ to a configured target
Needs the deploy edition and a [deployment] block in the config.

`hugo deploy --dryRun`
Show what would change

`hugo deploy --target <name>`
Choose a deployment target

RECIPES

New site with a theme

```
hugo new site blog
cd blog
git init
git submodule add <theme-url>
themes/<theme>
echo "theme = '<theme>'" >>
hugo.toml
hugo server -D
```

Write and publish a post

```
hugo new content posts/first.md
# edit it, set draft = false
hugo list drafts
hugo --minify --gc --
cleanDestinationDir
```

Clean production build

```
hugo --gc --minify --
cleanDestinationDir \
  -b https://example.com/ -d
public
```

Preview on a phone

```
hugo server --bind 0.0.0.0 -D \
  -b http://192.168.1.20:1313
--appendPort=false
```