chezmoi Dotfile manager

START HERE

`chezmoi init <user>`
Clone your dotfiles repo

`chezmoi update`
Pull and apply

`chezmoi add <file>`
Start managing a file

`chezmoi cd`
Shell into the source dir

`chezmoi edit <file>`
Edit the source copy

`chezmoi diff`
Preview changes

`chezmoi apply`
Write changes to $HOME

`chezmoi status`
What would change?

Source dir ~/.local/share/chezmoi → `templates + data` → **Target state** computed per machine → `chezmoi apply` → **Destination** your home dir → `chezmoi add / re-add` → **Source dir** back into the repo

CONCEPTS

chezmoi keeps a source directory (a git repo) and copies it into your home directory; your real dotfiles are never symlinks unless you ask.

Always run chezmoi diff (or apply -n -v) before apply on a new machine: apply overwrites files in $HOME.

SETUP

`chezmoi init`
Create an empty source directory

`chezmoi init <user>`
Clone github.com/<user>/dotfiles

`chezmoi init --apply <user>`
Clone and apply in one step

`chezmoi cd`
Open a shell in the source dir

MANAGE FILES

`chezmoi add <file>`
Start managing a file or dir

MANAGE FILES (cont.)

`chezmoi add --template <file>`
Add as a template

`chezmoi re-add`
Pull edited $HOME files back into source

`chezmoi edit <file>`
Edit a managed file's source

`chezmoi edit --apply <file>`
Edit and apply on exit

INSPECT AND APPLY

`chezmoi diff`
Show what apply would change

`chezmoi apply`
Make $HOME match the source

`chezmoi apply -n -v`
Dry run with verbose output

`chezmoi status`
Two-column change summary

First column: changed since chezmoi last wrote it. Second: what apply would do.

INSPECT AND APPLY (cont.)

`chezmoi managed`
List managed targets

`chezmoi update`
Pull the repo, then apply

`chezmoi git -- <args>`
Run git in the source dir

Put git flags after -- so chezmoi does not parse them.

TEMPLATES AND DATA

`chezmoi data`
Show template data as JSON

`chezmoi execute-template '<tmpl>'`
Render a template string

Template variables

Variable	Value
`.chezmoi.os`	linux, darwin, windows
`.chezmoi.arch`	amd64, arm64
`.chezmoi.hostname`	Short hostname
`.chezmoi.username`	Current user
`.chezmoi.homeDir`	Home directory
`.chezmoi.sourceDir`	Source directory

TEMPLATES AND DATA (cont.)

OS-specific block

```
{{ if eq .chezmoi.os "darwin" -}}
export BROWSER=open
{{ else -}}
export BROWSER=xdg-open
{{ end -}}
```

SOURCE NAMING

Source prefixes

Name	Effect
`dot_zshrc`	Target .zshrc
`private_`	No group/world access
`executable_`	Mark executable
`readonly_`	Remove write bits
`symlink_`	Create a symlink
`exact_`	Delete unmanaged children
`encrypted_`	Stored encrypted
`create_`	Write only if missing
`modify_`	Script that edits a file
`remove_`	Delete the target

SOURCE NAMING (cont.)

Suffixes and scripts

Name	Effect
`file.tmpl`	Contents are a template
`run_<name>`	Run script on every apply
`run_once_<name>`	Run once per content
`run_onchange_<name>`	Run when content changes
`before_ / after_`	Script ordering vs files

`.chezmoiignore`
Targets to skip (template, per machine)

SECRETS AND EXTERNALS

Config must set encryption = "age" (or "gpg") at the top of chezmoi.toml before --encrypt works.

GLOBAL FLAGS

`-n / --dry-run`
Never modify anything

GLOBAL FLAGS (cont.)

`-v / --verbose`
Show diffs and commands

RECIPES

New machine in one line

```
chezmoi init --apply <user>
chezmoi status
# later: pick up changes
chezmoi update
```

Track a new dotfile

```
chezmoi add ~/.config/nvim
chezmoi edit ~/.config/nvim/init.lua
chezmoi diff
chezmoi apply
```

Commit and push the source

```
chezmoi cd
git add -A
git commit -m "Update dotfiles"
git push
```

Per-machine git identity

```
# dot_gitconfig.tmpl
[user]
  name = {{ .name | quote }}
  email = {{ .email | quote }}
```