chezmoi Dotfile manager

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.

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

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

`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

INSPECT AND APPLY (cont.)

`chezmoi apply`
Make $HOME match the source

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

`chezmoi status`
Two-column change summary

`chezmoi managed`
List managed targets

`chezmoi update`
Pull the repo, then apply

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

TEMPLATES AND DATA

`chezmoi data`
Show template data as JSON

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

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

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

GLOBAL FLAGS

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

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