

stownd symlink manager

Ralph Seichter

1.2.0, 2025-02-27

Table of Contents

1. About	1
2. Installing	1
2.1. Requirements	1
2.2. All platforms	1
2.3. Arch Linux	1
3. Command line options	1
4. Usage example	2
5. Strategy	2
5.1. Overview	2
5.2. Example scenario	3
6. Environment variables	4
7. Support	4
8. License	5

1. About

"stown" manages file system object mapping via symlinks. It was inspired by GNU Stow, which I found very useful but too unwieldy for my personal use. GNU Stow relies on a number of Perl modules and can be a hassle to install on minimalistic systems. In contrast, stown requires only Python 3.9 or newer, without any additional dependencies.

If you are looking for a lightweight tool instead of a full-fledged symlink farm manager, stown might be for you.

Documentation is provided in [HTML](#) and [PDF](#) format. Source code is hosted [here](#).

2. Installing

2.1. Requirements

stown requires Python version 3.9 or newer. That's all.

2.2. All platforms

The first option is to install the [PyPI package](#) using Python's pip utility, available for all platforms.

```
pip install stown
```

Depending on your OS, [pip](#) may need to be installed first, but it is often available by default. On Debian family systems, the matching package is typically called `python3-pip`. If your environment is bare-bones and lacks a package manager, there is also Python's built-in [ensurepip](#) module you can use.

2.3. Arch Linux

Arch users can install the [AUR package](#) as a convenient alternative to the PyPI package.

3. Command line options

```
usage: stown [-h] [-a {link,unlink}] [-b] [-d] [-f] [-i RE] [-l LEVEL] [-n]
           [-o RE] [-D DEPTH] [-L PATH]
           target source [source ...]
```

Manage file system object mapping via symlinks

positional arguments:

target	action target (links are created here)
source	action sources (links point here)

options:

-h, --help	show this help message and exit
-a, --action {link,unlink}	action to take [link]
-b, --absolute	create links using absolute paths
-d, --dry-run	log operations but do not modify
-f, --force	force action (overwrite permission)
-i, --ignore RE	ignore sources matching regex
-l, --loglevel LEVEL	log level [WARNING]
-n, --no-dot	disable dot-prefix treatment
-o, --override RE	override targets matching regex
-D, --depth DEPTH	maximum recursion depth [10]
-L, --logpath PATH	log data destination

stown version 1.2.0+9fb82aa Copyright © 2025 Ralph Seichter

4. Usage example

The following example simulates linking the contents of a dotfile repository to a user's home directory. The `--dry-run` flag causes the necessary steps to be printed only; no changes will be made.

```
stown --dry-run $HOME /path/to/dotfiles
```

5. Strategy

Terminology: In stown parlance, *source* means any existing file object (regular file, directory or symlink). *Targets* are paths in the file system where stown is meant to create symlinks pointing to sources.

5.1. Overview

When used with default options, stown will routinely abort operations in an attempt to protect existing targets. This can however lead to half-finished jobs. Using `--dry-run` prior to any live operation is recommended to lower that risk. Note that stown limits its recursive approach to managing links to a maximum depth of 10, which is typically sufficient. The `--depth` option can be used to

change the limit.

WARNING | Always create backups before using stown, because here be monsters!

Using the `--force` flag gives stown explicit permission to overwrite existing file objects. Sources are processed in the order specified by the user. Already existing targets may not be overwritten, unless permission is granted via either the `--force` or `--override` options. Override, a feature introduced in stown 1.1.0, matches target names against a user-defined [regular expression](#) (RE).

```
# Use quotes to protect RE from shell interpolation
stown --override '\.(json|toml)$' sometarget somesource
```

RE are powerful, but they can also lead to surprises for those unfamiliar with their use: For instance, the regular expression `.c` matches "foo.c" but also ".conf", "sketch" and countless other strings.

5.2. Example scenario

As a more in-depth example, let us imagine a directory `mike` originally containing the following:

```
mike
├── alpha
│   ├── bar
│   ├── bar.txt
│   └── foo
├── bravo
│   ├── dot-editorconfig
│   ├── foo
│   └── quux
└── zulu
```

This is the content after `stow --force --ignore '\.txt$' zulu alpha bravo`:

```

mike
├── alpha
│   ├── bar
│   ├── bar.txt
│   └── foo
├── bravo
│   ├── dot-editorconfig
│   ├── foo
│   └── quux
└── zulu
    ├── .editorconfig -> ../bravo/dot-editorconfig
    ├── bar -> ../alpha/bar
    ├── foo -> ../bravo/foo
    └── quux -> ../bravo/quux

```

Symlinks are created using relative paths with stown's default settings. Using the `--absolute` flag changes the path type. Links with relative paths can make changing the directory structure easier. For example, renaming the parent folder `mike` to `november` would not break the relative links shown above.

Unless disabled via the `--no-dot` option, the prefix `dot-` in source names is converted to a leading dot character in symlinks, as shown with `zulu/.editorconfig` pointing to `bravo/dot-editorconfig`. This makes handling dotfiles easier, disabling their respective special effects and lifting their "hidden" status.

Like the `--override` option, `--ignore` takes a regular expression argument. However, while `override` matches targets, the `ignore` option matches sources. `stown` uses a small but reasonable default ignore pattern. If the outcome of a run is not what you expected, try the `INFO` loglevel to see if certain sources are being ignored, possibly requiring you to specify a custom ignore argument. An empty string RE pattern turns the ignore feature off.

6. Environment variables

- `STOWN_LOGLEVEL` can be used like the `--loglevel` option. Supported values are the Python's predefined logging levels `CRITICAL`, `ERROR`, `WARNING` (the default), `INFO` or `DEBUG`. Values are case insensitive. `CRITICAL` will effectively silence `stown`.
- `STOWN_LOGPATH` corresponds to the `--logpath` option. As per convention, a standalone dash `-` is interpreted as an alias for the standard output stream, which is also the default.

7. Support

For questions regarding `stown` and its use, please use the [discussions](#) section. If you are experiencing technical problems, you can [file an issue](#) instead. Both methods allow other users to participate. If privacy is a concern, you can use [email](#) to contact me.

Note that my support offer is strictly voluntary.

8. License

Copyright © 2025 Ralph Seichter

This file is part of "stown".

stown is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

stown is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with stown. If not, see <https://www.gnu.org/licenses/>.
