Pip'less, Brew'less ... and broke

Introduction

Hello and welcome to ... I have been here awhile and want to become more active and share my tinkering. I am interested in all things Sdf.org especially Gopher and later I will look at Gemini. I want to investigate possibilities of SDF as much as I can. With a little creativity I think there is a lot of potential.

The following steps where done with a MetaARPA account, I am almost certain these steps will NOT work with a lower account membership.

This is my first how-to if you want to call it that. The inspriation was my frustration with not having a few basic tools that I have used over the years on different systems, which lead to a decentralized kind of mess with no way in to my home network. I built multiple solutions over the years but nothing smoother than accessing a public system that I don't maintain and I can reach from anywhere. Using BBOARD or app request for every tiny little thing is cumbersome for all involved so I looked into a few tools that I like to use and seeing if I could get them loaded. I would caution to not get too crazy these are very small apps and don't pose any possible damage that I can see.

Name	Description
Safe Closet	A password vault of sorts written in rust
buku	CLI bookmark manager
Topydo	a powerful todo app
Calcurse	curses terminal calendar
** It turns out calcurse was already installed in meta! **	

App Manager	Info
pip(3)	not available use setup.py
Homebrew	locally instalable

Why consider Pip packages and Homebrew without superuser access?

For security reasons it makes sense to limit regular users and keep access restricted, but if you could install to your home directory without affecting anything else that would be awesome. A way to simplify my workflow on SDF.org, reducing the need to switch between multiple SSH sessions and accessable world wide.

Things covered in this article:

- How to install as a standard user (Meta ARPA).
- Method to install software using pip packages that contain setup.py and Homebrew in your home directory.
- Practical benefits and use of local home drive installations.
- References at the bottom of the article for further reading.

Let's see how you can enhance your SDF.org experience by using package managers without needing superuser rights. For the first example are not really using pip but we are working with the package itself and Python. Which still achieves the goal of installing the desired software.

Step-by-Step Guide: Installing TopyDo and Homebrew

Topydo is a powerful todo list CLI application using the todo.txt

1. Installing TopyDo

- **Locate the GitHub repository**: Find and clone the repository for TopyDo.

```
git clone https://github.com/topydo/topydo.git
```

- **Prepare the Installation**: Navigate to the installation directory, find `setup.py`, and make it executable (`chmod +x setup.py`).

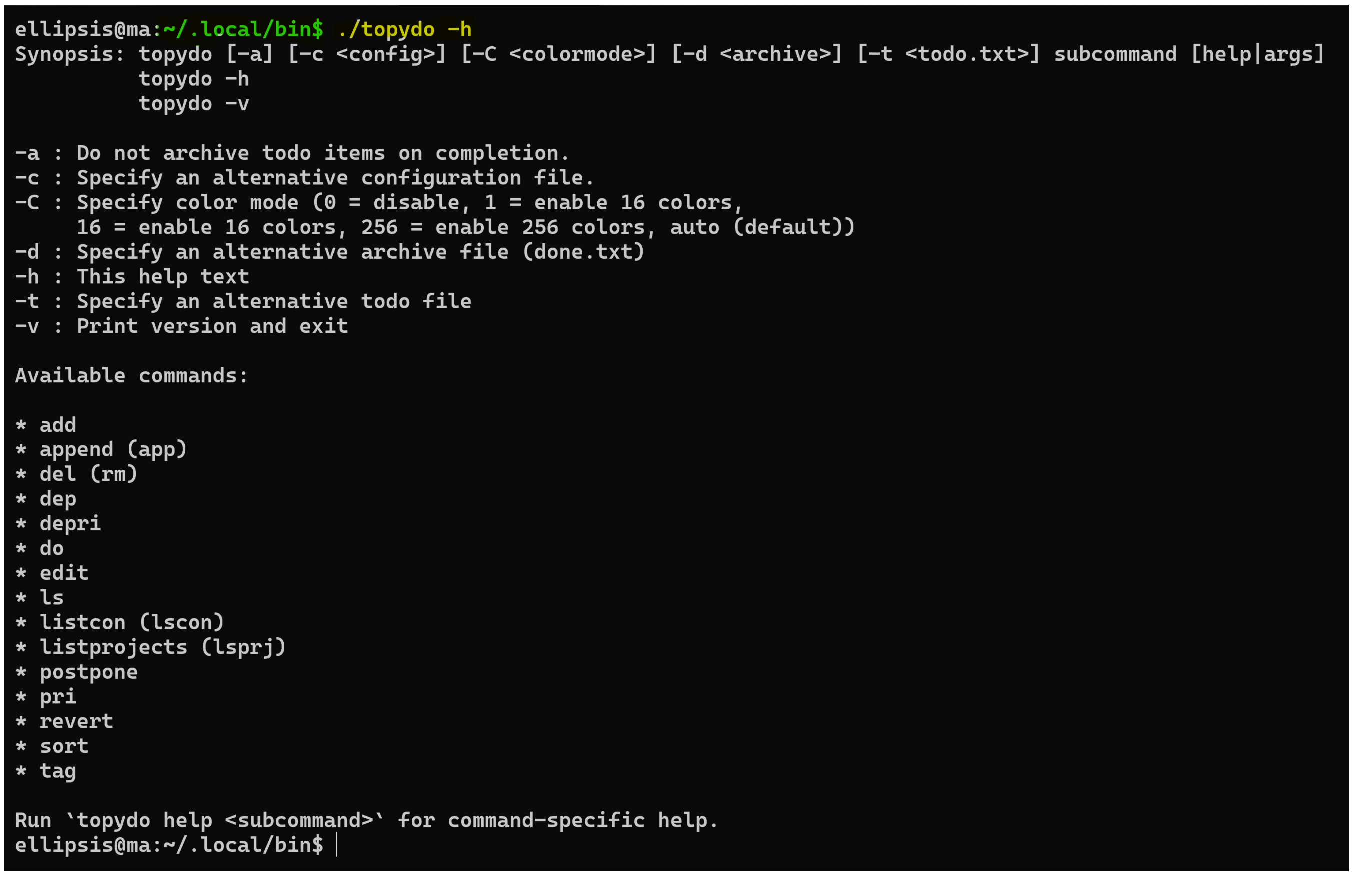
```
cd topydo
chmod +x setup.py
```

- **Install TopyDo**: Run the installation with user-level permissions to avoid needing superuser access:

```
python setup.py install --user
```

- **Verify the Installation**: Test TopyDo by running basic commands to ensure it's functioning correctly.

```
# My result installed into the the ~/.local/bin directory
cd ~/.local/bin

./topydo -h
./topydo --v
```

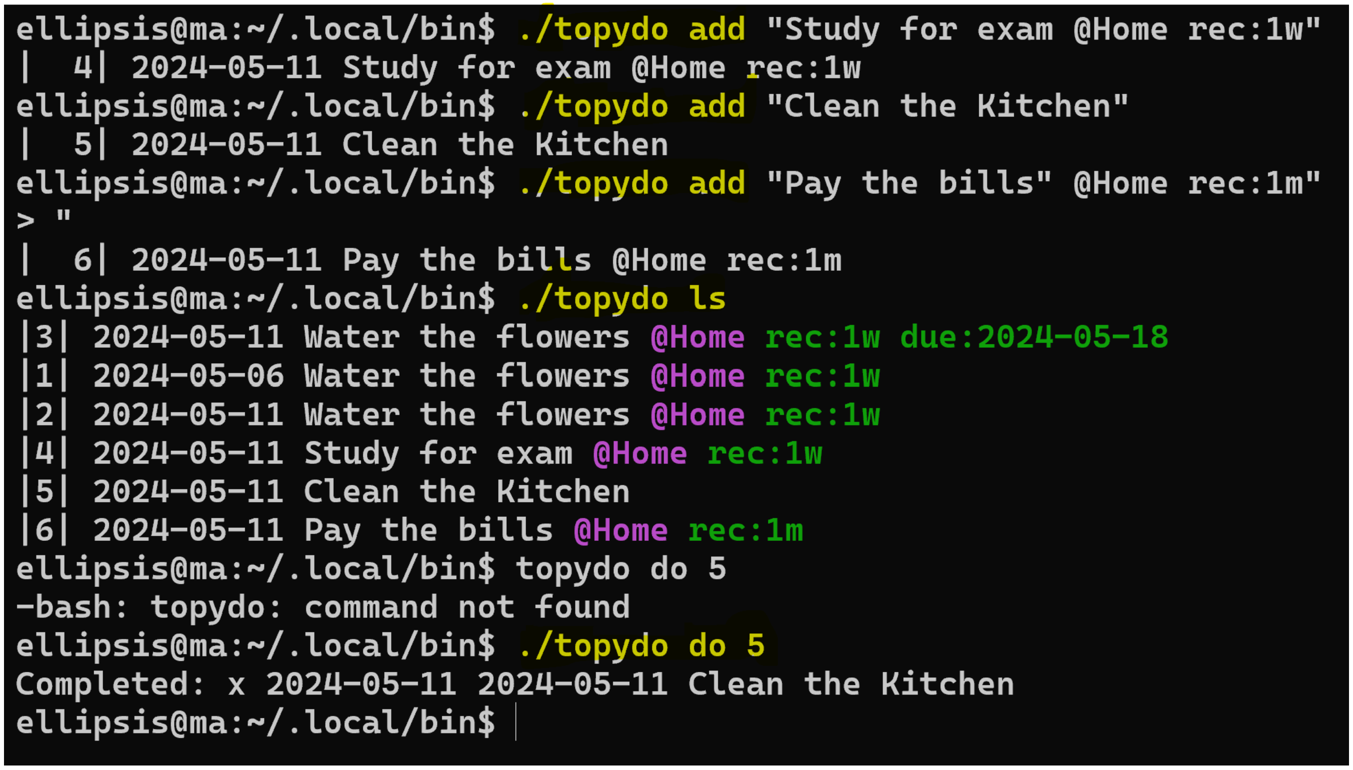


```
ellipsis@ma:~/.local/bin$ ./topydo -h
Synopsis: topydo [-a] [-c <config>] [-C <colormode>] [-d <archive>] [-t <todo.txt>] subcommand [help|args]
          topydo -h
          topydo -v

-a : Do not archive todo items on completion.
-c : Specify an alternative configuration file.
-C : Specify color mode (0 = disable, 1 = enable 16 colors,
     16 = enable 16 colors, 256 = enable 256 colors, auto (default))
-d : Specify an alternative archive file (done.txt)
-h : This help text
-t : Specify an alternative todo file
-v : Print version and exit

Available commands:

* add
* append (app)
* del (rm)
* dep
* depri
* do
* edit
* ls
* listcon (lscon)
* listprojects (lsprj)
* postpone
* pri
* revert
* sort
* tag

Run `topydo help <subcommand>` for command-specific help.
ellipsis@ma:~/.local/bin$
```

Add your first todo entry and it creates a todo.txt

```
topydo add "Water the flowers @Home rec:1w"
```

The example does not do the tool justice, its good tool and maybe I will write a follow up to show more. The topydo github page has a good demo.



```
ellipsis@ma:~/.local/bin$ ./topydo add "Study for exam @Home rec:1w"
|  4| 2024-05-11 Study for exam @Home rec:1w
ellipsis@ma:~/.local/bin$ ./topydo add "Clean the Kitchen"
|  5| 2024-05-11 Clean the Kitchen
ellipsis@ma:~/.local/bin$ ./topydo add "Pay the bills" @Home rec:1m"
> "
|  6| 2024-05-11 Pay the bills @Home rec:1m
ellipsis@ma:~/.local/bin$ ./topydo ls
|3| 2024-05-11 Water the flowers @Home rec:1w due:2024-05-18
|1| 2024-05-06 Water the flowers @Home rec:1w
|2| 2024-05-11 Water the flowers @Home rec:1w
|4| 2024-05-11 Study for exam @Home rec:1w
|5| 2024-05-11 Clean the Kitchen
|6| 2024-05-11 Pay the bills @Home rec:1m
ellipsis@ma:~/.local/bin$ topydo do 5
-bash: topydo: command not found
ellipsis@ma:~/.local/bin$ ./topydo do 5
Completed: x 2024-05-11 2024-05-11 Clean the Kitchen
ellipsis@ma:~/.local/bin$
```

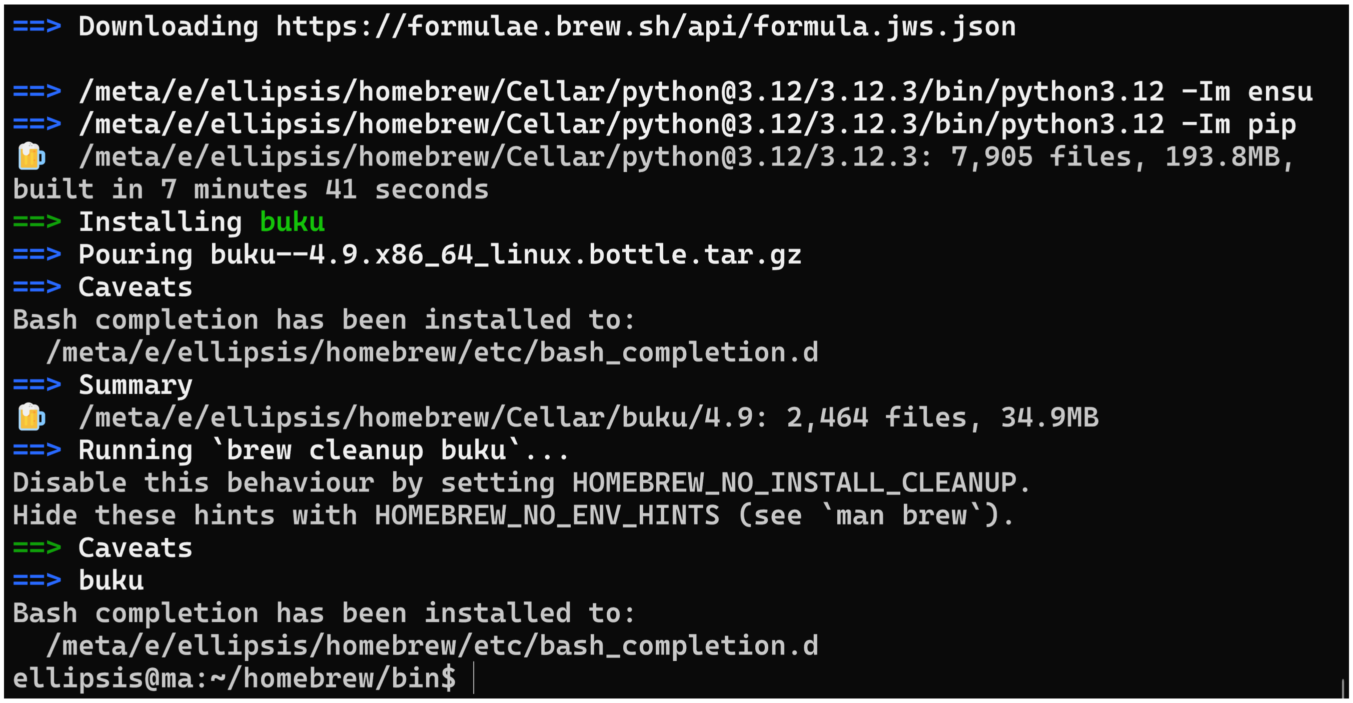
2. Installing Homebrew

- **Create a Homebrew Directory and Download**: Set up a directory for Homebrew and download the installation script:

```
mkdir homebrew && curl -L https://github.com/Homebrew/brew/tarball/master | tar xz --
strip 1 -C homebrew
```

- **Install Software with Homebrew**: Use Homebrew to install additional software, such as Buku, a command-line bookmark manager:

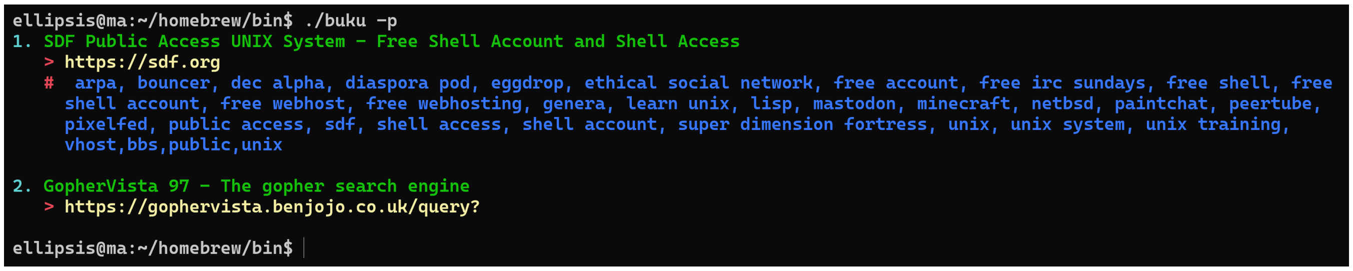
```
brew install buku
```



```
==> Downloading https://formulae.brew.sh/api/formula.jws.json

==> /meta/e/ellipsis/homebrew/Cellar/python@3.12/3.12.3/bin/python3.12 -Im ensu
==> /meta/e/ellipsis/homebrew/Cellar/python@3.12/3.12.3/bin/python3.12 -Im pip
🍺  /meta/e/ellipsis/homebrew/Cellar/python@3.12/3.12.3: 7,905 files, 193.8MB,
built in 7 minutes 41 seconds
==> Installing buku
==> Pouring buku--4.9.x86_64_linux.bottle.tar.gz
==> Caveats
Bash completion has been installed to:
  /meta/e/ellipsis/homebrew/etc/bash_completion.d
==> Summary
🍺  /meta/e/ellipsis/homebrew/Cellar/buku/4.9: 2,464 files, 34.9MB
==> Running `brew cleanup buku`...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see `man brew`).
==> Caveats
==> buku
Bash completion has been installed to:
  /meta/e/ellipsis/homebrew/etc/bash_completion.d
ellipsis@ma:~/homebrew/bin$
```

- **Verify Installation**: Ensure Buku is correctly installed in `~/homebrew/bin`.

```
# Add a bookmark (Buku tries to add a description
./buku --add https://sdf.org unix,public

#Add a few more then list them with -p
./buku -p
```

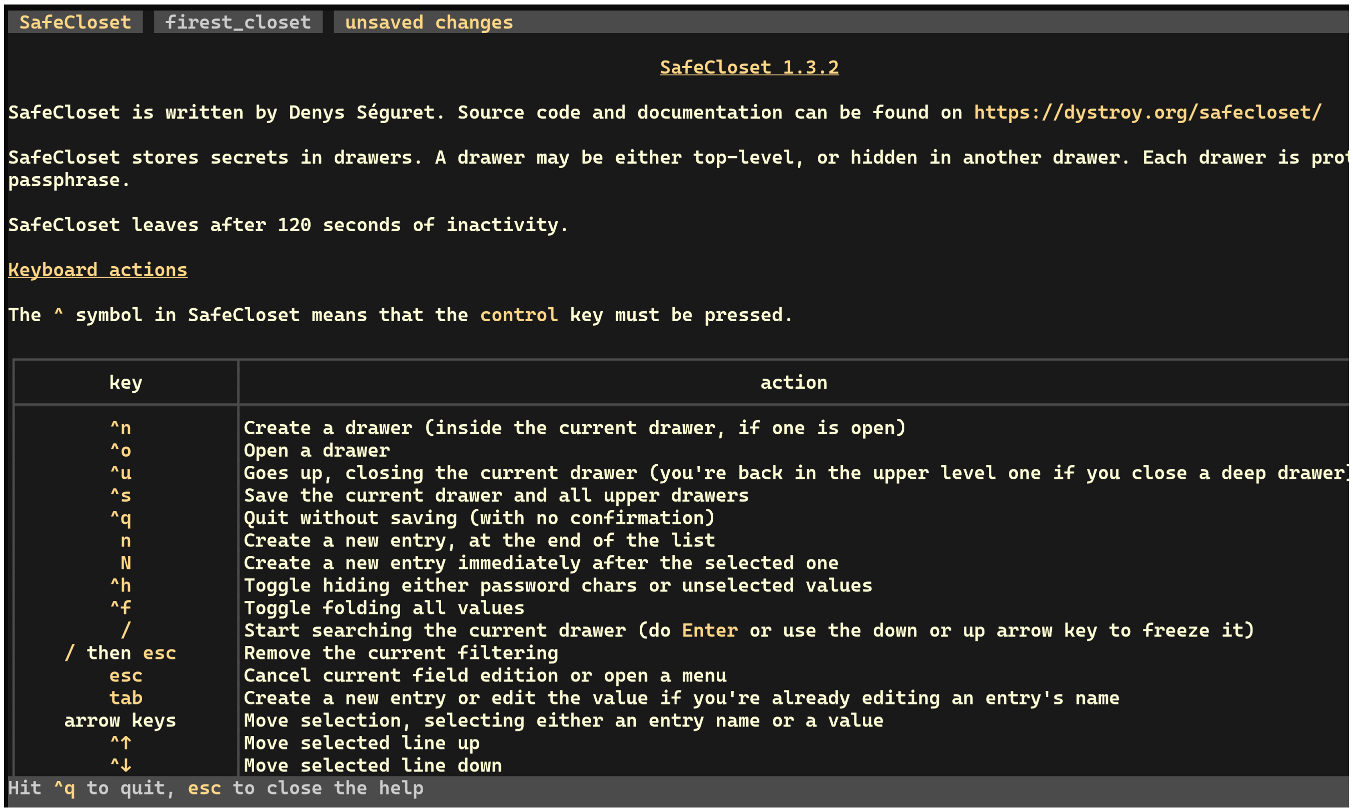


```
ellipsis@ma:~/homebrew/bin$ ./buku -p
1. SDF Public Access UNIX System - Free Shell Account and Shell Access
   > https://sdf.org
   #  arpa, bouncer, dec alpha, diaspora pod, eggdrop, ethical social network, free account, free irc sundays, free shell, free
      shell account, free webhost, free webhosting, genera, learn unix, lisp, mastodon, minecraft, netbsd, paintchat, peertube,
      pixelfed, public access, sdf, shell access, shell account, super dimension fortress, unix, unix system, unix training,
      vhost,bbs,public,unix

2. GopherVista 97 - The gopher search engine
   > https://gophervista.benjojo.co.uk/query?

ellipsis@ma:~/homebrew/bin$
```

3. Installing Safecloset

- **Download and Prepare the Installation**: Obtain the precompiled package of Safecloset, a tool for securely managing passwords and secrets, from its official GitHub release page. https://github.com/Canop/safecloset/releases
- **Make Executable and Test**: Change the permissions to make the file executable (`chmod +x`) and test to confirm it functions as intended.

```
chmod +x safecloset

./safecloset -o myFirst_safeCloset
```

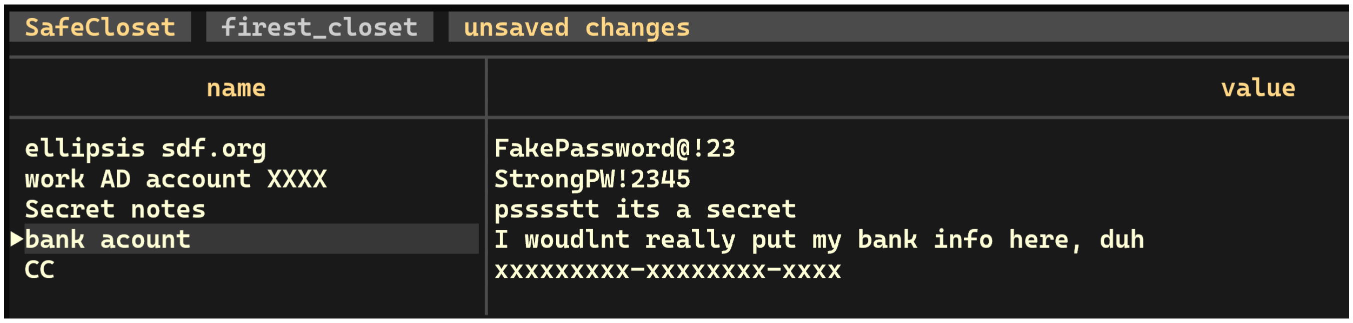


```
 SafeCloset   firest_closet   unsaved changes

                                    SafeCloset 1.3.2

SafeCloset is written by Denys Séguret. Source code and documentation can be found on https://dystroy.org/safecloset/

SafeCloset stores secrets in drawers. A drawer may be either top-level, or hidden in another drawer. Each drawer is prot
passphrase.

SafeCloset leaves after 120 seconds of inactivity.

Keyboard actions

The ^ symbol in SafeCloset means that the control key must be pressed.
```

key	action
^n	Create a drawer (inside the current drawer, if one is open)
^o	Open a drawer
^u	Goes up, closing the current drawer (you're back in the upper level one if you close a deep drawer)
^s	Save the current drawer and all upper drawers
^q	Quit without saving (with no confirmation)
n	Create a new entry, at the end of the list
N	Create a new entry immediately after the selected one
^h	Toggle hiding either password chars or unselected values
^f	Toggle folding all values
/	Start searching the current drawer (do Enter or use the down or up arrow key to freeze it)
/ then esc	Remove the current filtering
esc	Cancel current field edition or open a menu
tab	Create a new entry or edit the value if you're already editing an entry's name
arrow keys	Move selection, selecting either an entry name or a value
^↑	Move selected line up
^↓	Move selected line down

```
Hit ^q to quit, esc to close the help
```

A few secret test entries:



```
 SafeCloset   firest_closet   unsaved changes
```

name	value
ellipsis sdf.org	FakePassword@!23
work AD account XXXX	StrongPW!2345
Secret notes	psssstt its a secret
▶bank acount	I woudlnt really put my bank info here, duh
CC	xxxxxxxxx-xxxxxxxx-xxxx

Conclusion

This exploration on SDF.org reveals that user-level restrictions don't need to limit your software capabilities. By adapting installation procedures and utilizing tools like Homebrew and Python's setup tools, you can still install some of your desired software and streamline your workflow, all within the constraints of regular user permissions. I believe these practical examples are steps that help enrich the experience at sdf and also encourage further experimentation and discovery in a shared Unix environment.

P.S. the "broke" part in the title had nothing to do with anything technically. Im just broke atm :)

References

TOPY DO

- https://github.com/topydo/topydo
- https://github.com/todotxt/todo.txt

Homebrew

- https://docs.brew.sh/Installation
- https://en.wikipedia.org/wiki/Getting_Things_Done

BUKU

- https://github.com/jarun/Buku/wiki

SafeCloset

- https://github.com/Canop/safecloset