Command Line Tips & Tricks

Johannes Alneberg 2023-09-29

Slides here: https://github.com/alneberg/command_line_tips_and_tricks

`cd`

```
cd ../
cd    # - without arguments
cd ~
cd -
```

Use tab!

`ls`

```
ls ../
ls /my/absolute/path
ls ../../inside/somewhere
```

Useful options: `-l` (`ll`), `-a` , `-lh`

Use tab!

Jump around

```
ctrl+a
ctrl+e
```

Inside screen:

```
ctr+a a
```

Jump around 2 - prerequisites

Map meta to opt:

- vscode:

```
  "terminal.integrated.macOptionIsMeta": true
```

- iterm: link
- terminal link

Jump around 2

Move word by word

```
opt+f
opt+b
```

`tr` - Translate characters

```
tr ',' '\t'
```

stdin

`tr` does not take a file input, only stdin

Common to use `cat file.tsv | tr '\t' ','`

But `< file.tsv` is slightly nicer:

```
tr '\t' ',' < file.tsv
```

stdout & stderr

Two output streams, one is intended for output and the other for essentially logging.

If not redirected, they are both printed on the terminal.

- stdout is redirected with pipes and to files with `>` or `>>`
- stderr can be redirected with `2>` or `2>>`
- redirect stderr to stdout `2>&1`
- redirect stderr & stdout to the same file `&>` or `&>>`

Case: tsv to csv convertion

Case: tsv to csv convertion

```
tr '\t' ',' < sandbox/file3.tsv > sandbox/file3.csv
```

Don't use the same file name!

`cut`

Extract columns from file

```
cut -c3-5
cut -f2 -d ','
```

`paste`

- merge files "side by side"

```
paste file1.csv file2.csv
```

|

"pipe" - Connects stdout of one command to the stdin of the following

```
paste file1.csv file2.csv | cut -f 3,5 -d ','
```

Case: reordering columns

file1.csv has column 2 and 3 re-ordered.

Case: reordering columns

file1.csv has column 2 and 3 re-ordered.

```
paste file1.csv file1.csv | cut -f 1,3,6,8- -d',' > file1_correct.csv
mv file1_correct.csv file1.csv
```

Explain Shell

explainshell.com

Case: Which wheel size is more popular?

`uniq -c` - report or omit repeated lines

`sort` - alphabetic sort

`sort -n` - numerical sort

```
cut -f 4 -d',' sandbox/*.csv | sort | uniq -c | sort -n
```

Search history

Soo useful for repeated commands

`ctrl+r`

For fuzzy searching `fzf` (thanks Matthias H)

`history` command is useful for investigating command history

dotfiles

- `.zsh_aliases`
- `.zprofile`
- `.zshrc`
- `.vimrc`

Keep them backed up somehow

aliases

example:

```
alias squiggle='ssh johannes.alneberg@ngi-squiggle.scilifelab.se'
alias docker_delete_all='docker rm $(docker ps -a -q); docker rmi $(docker images -q)'
alias miarka1='ssh alneberg@miarka1.uppmax.uu.se'
alias miarka2='ssh alneberg@miarka2.uppmax.uu.se'
alias miarka3='ssh alneberg@miarka3.uppmax.uu.se'
alias lims-prod='ssh johannes.alneberg@ngi-lims-prod.scilifelab.se'
alias lims-stage='ssh johannes.alneberg@ngi-lims-stage.scilifelab.se'
```

ssh configs:

Example:

```
Host ngi-preproc
    HostName ngi-preproc.scilifelab.se
    User johannes.alneberg

Host miarka
    HostName miarka1.uppmax.uu.se
    User alneberg
    ServerAliveInterval 30
```

Enables commands like:

```
scp ngi-preproc:/my_data/path .
```

`.zprofile`

example:

```
# Enable moving path levels as words
WORDCHARS=''
```

I also use powerlevel10k to customize the shell.

Other commands not covered

```
- which
- du
- df
- grep
- sed
- less
- bat
- whoami
- hostname
- zcat
- htop
- ps
- xargs
- bc
```

And many many more!

Thank you!

Questions?

Comments?

Corrections?