

Introduction to Terminal

Qingyu Song

XMU

November 10, 2025

Credit to Jackie Baek, Computing in Optimization and Statistics 2017, MIT

Overview

Introduction & Motivation

Navigation commands

Files

- Basic file commands

- File path shortcuts

- Hidden Files

- `.bashrc` / `.bash_profile`

Redirection

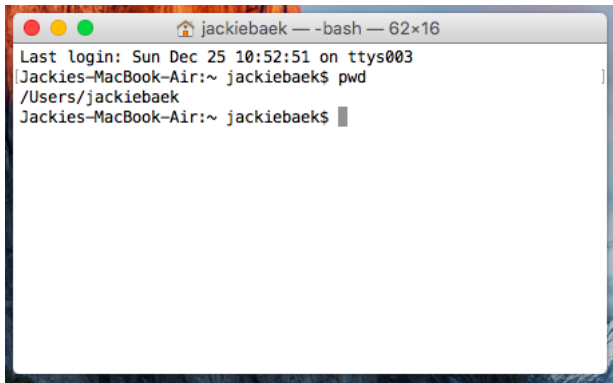
Documentation

Key Takeaways

What is the terminal?



What is the terminal?



```
jackiebaek — -bash — 62x16
Last login: Sun Dec 25 10:52:51 on ttys003
[Jackies-MacBook-Air:~ jackiebaek$ pwd
/Users/jackiebaek
Jackies-MacBook-Air:~ jackiebaek$
```

- ▶ The terminal is a text-based interface to interact with the computer.
- ▶ Alternate names: console, shell, command line, command prompt

Example

- ▶ Say you want to delete all files in a directory that end with .pyc

```
$ rm *.pyc
```

- ▶ This is possible to do without the terminal, but it requires much more effort.

Why should I learn it?

- ▶ You can do almost everything using just the terminal.
- ▶ It can do many tasks faster than using a graphic interface.
- ▶ It is sometimes the only option (e.g. accessing a client's server using SSH).
- ▶ It is universal.

Terminal Basics

- ▶ We will be using a **shell** called **bash**: a program that interprets and processes the commands you input into the terminal.
- ▶ The shell is always in a **working directory**.
- ▶ A typical command looks like:

```
$ command <argument1> <argument2> ...
```

Basic navigation commands

pwd: prints working directory.

```
$ pwd  
/Users/jackiebaek
```


Basic navigation commands

pwd: prints working directory.

```
$ pwd  
/Users/jackiebaek
```

ls: lists directory contents.

```
$ ls  
Applications      Movies  
Desktop           Music  
Documents         Pictures
```

Basic navigation commands

pwd: prints working directory.

```
$ pwd  
/Users/jackiebaek
```

ls: lists directory contents.

```
$ ls  
Applications                Movies  
Desktop                     Music  
Documents                   Pictures
```

cd <directory>: change working directory to new directory.

```
$ cd Documents  
$ pwd  
/Users/jackiebaek/Documents
```

Tab and arrow keys are your friends

- ▶ Use **tab** to autocomplete *commands* and *file paths*.
- ▶ Use ↑ and ↓ arrow keys to navigate through your command history.
- ▶ Use **clear** or **cmd-k** to clear screen.

What is a file?

A file is a container of data (0's and 1's).

What is a file?

A file is a container of data (0's and 1's).

A file name usually has an **extension** (e.g. .pdf, .doc, .csv), but these are just conventions.

What is a file?

A file is a container of data (0's and 1's).

A file name usually has an **extension** (e.g. .pdf, .doc, .csv), but these are just conventions.

A file is contained in a **directory** (folder). Files within the same directory have unique names.

What is a file?

A file is a container of data (0's and 1's).

A file name usually has an **extension** (e.g. .pdf, .doc, .csv), but these are just conventions.

A file is contained in a **directory** (folder). Files within the same directory have unique names.

Every file and directory has a unique location in the file system, called a **path**.

- ▶ **Absolute path:**

- /Users/jackiebaek/Dropbox/Documents/hello.txt*

- ▶ **Relative path** (if my current working directory is */Users/jackiebaek/Dropbox*): *Documents/hello.txt*

Working with files

Working with files

mkdir *directory_name*: create a new directory.

```
$ mkdir new_directory
```

Working with files

mkdir *directory_name*: create a new directory.

```
$ mkdir new_directory
```

touch *file*: create an empty file.

rm *file*: delete a file (**Careful!** Can't be undone!)

```
$ touch brand_new_file.txt
```

```
$ rm brand_new_file.txt
```

Working with files

mkdir *directory_name*: create a new directory.

```
$ mkdir new_directory
```

touch *file*: create an empty file.

rm *file*: delete a file (**Careful!** Can't be undone!)

```
$ touch brand_new_file.txt
```

```
$ rm brand_new_file.txt
```

nano *file*: edit contents of a file (many other editors exist).

```
$ nano helloworld.txt
```

Working with files

mkdir *directory_name*: create a new directory.

```
$ mkdir new_directory
```

touch *file*: create an empty file.

rm *file*: delete a file (**Careful!** Can't be undone!)

```
$ touch brand_new_file.txt
```

```
$ rm brand_new_file.txt
```

nano *file*: edit contents of a file (many other editors exist).

```
$ nano helloworld.txt
```

cat *file*: prints contents of a file.

```
$ cat helloworld.txt
```

```
Hello, World!
```

Working with files

mkdir *directory_name*: create a new directory.

```
$ mkdir new_directory
```

touch *file*: create an empty file.

rm *file*: delete a file (**Careful!** Can't be undone!)

```
$ touch brand_new_file.txt
```

```
$ rm brand_new_file.txt
```

nano *file*: edit contents of a file (many other editors exist).

```
$ nano helloworld.txt
```

cat *file*: prints contents of a file.

```
$ cat helloworld.txt
```

```
Hello, World!
```

cp *source target*: copy.

mv *source target*: move/rename.

```
$ cp helloworld.txt helloworld_copy.txt
```

```
$ mv helloworld.txt goodbyeworld.txt
```

File path shortcuts

. is current directory.

.. is parent directory.

- ▶ ../*file.txt* references a file named *file.txt* in the parent directory.

File path shortcuts

`.` is current directory.

`..` is parent directory.

- ▶ `../file.txt` references a file named `file.txt` in the parent directory.

`~` is home.

- ▶ expands to `/Users/<username>` (or wherever *home* is on that machine).
- ▶ `~/Documents` → `/Users/jackiebaek/Documents`
- ▶ The command **cd** (without any arguments) takes you to `~`.

Hidden Files

- ▶ Files that start with a dot (.) are called **hidden** files.
- ▶ Used for storing preferences, config, settings.
- ▶ Use `ls -a` to list all files.

```
$ ls
github_notes.md  presentation  scripts
```

```
$ ls -a
.          .git          github_notes.md  scripts
..         .gitignore    presentation
```


.bashrc / .bash_profile

- ▶ There is a hidden file in `~` directory called `.bashrc` or `.bash_profile`.
- ▶ This file is a bash script that runs at the beginning of each session (i.e. when you open the terminal).

.bashrc / .bash_profile

- ▶ There is a hidden file in `~` directory called `.bashrc` or `.bash_profile`.
- ▶ This file is a bash script that runs at the beginning of each session (i.e. when you open the terminal).
- ▶ This file can be used to set variables or to declare **aliases**.
- ▶ **alias** *new_command=command*

```
$ alias athena="ssh baek@athena.dialup.mit.edu"
```

Redirection

> redirects output to a file, *overwriting* if file already exists.

```
$ ls > out.txt
```

>> redirects output to a file, *appending* if file already exists.

```
$ python fetch_data.py >> output.csv
```

Redirection

> redirects output to a file, *overwriting* if file already exists.

```
$ ls > out.txt
```

>> redirects output to a file, *appending* if file already exists.

```
$ python fetch_data.py >> output.csv
```

< uses contents of file as STDIN (standard input) to the command.

```
$ python process_stuff.py < input.txt
```

Documentation

- ▶ To look up documentation for a particular command, use '**man** *command*'. (man = manual)

```
$ man mkdir
```

NAME

```
mkdir -- make directories
```

SYNOPSIS

```
mkdir [-pv] [-m mode] directory_name ...
```

DESCRIPTION

```
The mkdir utility creates the directories named as operands ...
```

- ▶ *d* for down, *u* for up, *q* to quit.
- ▶ Commands can have required and/or optional arguments.
- ▶ Optional arguments usually come first, and are indicated by a hyphen (-). These are called **flags**.

Key Takeaways

Key Takeaways

- ▶ Basic commands: ls, cd, pwd, cat, cp, mv, rm, mkdir

Key Takeaways

- ▶ Basic commands: ls, cd, pwd, cat, cp, mv, rm, mkdir
- ▶ Google is your friend.

Key Takeaways

- ▶ Basic commands: `ls`, `cd`, `pwd`, `cat`, `cp`, `mv`, `rm`, `mkdir`
- ▶ Google is your friend.
- ▶ So is *tab* for autocomplete, *arrow keys* for history.

Key Takeaways

- ▶ Basic commands: `ls`, `cd`, `pwd`, `cat`, `cp`, `mv`, `rm`, `mkdir`
- ▶ Google is your friend.
- ▶ So is *tab* for autocomplete, *arrow keys* for history.
- ▶ Be careful with *rm*.

Key Takeaways

- ▶ Basic commands: `ls`, `cd`, `pwd`, `cat`, `cp`, `mv`, `rm`, `mkdir`
- ▶ Google is your friend.
- ▶ So is *tab* for autocomplete, *arrow keys* for history.
- ▶ Be careful with *rm*.
- ▶ Getting comfortable with the terminal can be daunting at first, but it has the potential to greatly boost your efficiency!

Thank you!