Note: *This is not a comprehensive reference by any stretch of the imagination!*

How to Read This

- Replace things in *italics* with actual filenames, URLs, etc.
- Anything in brackets `[]` is optional.
- *`file`* means one filename
- *`dir`* means a directory name
- *`name`* means a filename or directory name
- *`file [...]`* means "one or more files"
- **Note**: most commands can operate on multiple files, even if this sheet doesn't say so.

File Names and Globbing

Used with any of the commands, below.

`foo`	A file named `foo`
`foo*`	Files whose names begin with `foo`
`*foo`	Files ending with `foo`
`foo?`	Files starting with `foo` and ending with any single letter
`????`	Files made up of four letter names
`foo*bar`	Files that start with `foo` and end with `bar`
`1*2*3`	Files that start with `1`, end with `3`, and have `2` in between
`"foo bar"`	File with a space in the name
`foo.{gif,jpg}`	Expands to `foo.gif foo.jpg`

TAB completion

Start typing a filename or command name and hit **TAB** to complete it. Hit two times to see options if nothing happens.

Help

`man` *`command`*	Show manual page for a command
`help` *`builtin`*	Show Bash help for a built-in command, e.g. "help cd"

Exiting the Shell

`exit`	Exit the shell
CTRL-D	Send End-Of-File (EOF) to exit the shell

Getting Bearings

`ls`	Show directory listing

`ls -l`	Show long (detailed) directory listing
`ls -a`	Show all (including hidden) files
`ls -la`	Show all, long
`pwd`	Print Working Directory—where am I?
`cd`	Switch back to home directory

Permissions from `ls -l`

Arranged in triple of triples. Read, Write, and Execute permission for user, group, and other:

drwxrwxrwx	Leading d means "Directory"
d**rwx**rwxrwx	User permissions in bold
drwx**rwx**rwx	Group permissions in bold
drwxrwx**rwx**	Other permissions in bold
`drwxr-xr-x`	Directory. User can read, write, and enter. Group can read and enter. Other can read and enter.
`-rwxr-xr-x`	File. User can read, write, and execute (it's a program). Group can read and execute. Other can read and execute
`-rw-r-----`	File. User can read and write. Group can read. Other can do nothing.

See also: `chmod`

Moving Around and Directories

`cd` *`dirname`*	Change directory
`cd ..`	Change to parent directory
`cd -`	Change to previous directory
`cd`	Change to home directory
`cd ~`	Change to home directory, if you're ambitious
`mkdir` *`dirname`*	Make a new directory
`rmdir` *`dirname`*	Remove an empty directory
`pushd` *`dirname`*	Change to directory and push it on the directory stack
`popd`	Pop directory off directory stack and change to it
`dirs`	View the directory stack

Directory Names

`../foo`	Directory (or file) foo out of the parent directory
`./foo`	Directory (or file) foo in the current directory
`/`	The root directory of the filesystem
`~`	My home directory
`~/foo`	Directory (or file) foo out of my home directory

`-`	Previous directory (***cd command only!***)

File Manipulation

`rm` *`file`*	Remove (delete) a file
`rm -i` *`file`*	Ask for verification before delete
`rm -r` *`dir`*	Recursively remove a directory tree. ***Danger!***
`rm -rf` *`dir`*	Recursively remove a directory tree, force. ***Danger!***
`ls -l` *`file`*	List details about file
`ls -l` *`dir`*	List details about files in a directory
`ls -ld` *`dir`*	List details about a directory
`cp` *`file1 file2`*	Make a copy of file1
`mv` *`file1 file2`*	Rename file1 to file2
`mv` *`file dir`*	Move file into another directory
`stat` *`file`*	Print metadata about file
`locate` *`pattern`*	Locate all files with matching pattern
`file` *`file`*	Identify type of file from its contents

chmod

`chmod` *`mode file`*	Change permissions on a file
`chmod 755` *`file`*	User can RWX, group RX, other RX
`chmod 644` *`file`*	User can RW, group R, other R
`chmod 700` *`file`*	User can RWX
`chmod 600` *`file`*	User can RW
`chmod u+x` *`file`*	Add X permission to user
`chmod g-r` *`file`*	Remove R permission from group
`chmod o+w` *`file`*	Add W permission to other
`chmod a+rw` *`file`*	Give all users RW permission

Note that your directories need *at least* u+x (or 700) permission if you want to be able to read them yourself.

File Manipulation II

`head` *`file`*	Show first 10 lines of a file
`head -23` *`file`*	Show first 23 lines of a file
`tail -n +7` *`file`*	Show end of file starting from 7th line from beginning
`tail -n 13` *`file`*	Show last 13 lines of file
`tail -f` *`file`*	Show last 10 lines of file, then show more as file is updated
`more` *`file`*	Page through a long file
`less` *`file`*	Page through a long file, improved
`most` *`file`*	Page through a long time, improved more
`sort` *`file`*	Sort a file a line at a time

`wc` *file*	Word count: lines, words, characters
`wc -w` *file*	Word count: words only
`wc -l` *file*	Word count: lines only
`sort -u` *file*	Sort a file (unique), collapse duplicate lines into a single line
`cat` *file […]*	Display file(s) on the screen
`cut -d' ' -f 5` *file*	Cut the 5th space-delimited field from each line
`cut -d' ' -f 6,7` *file*	Cut the 6th and 7th space-delimited fields from each line
sed	Powerful stream editor
`sed 's/foo/bar/g'`	Replace all occurrences of `foo` with `bar`
awk	Powerful text file processing language
`ls -l | tail +2 | grep -v ^d | awk 'BEGIN {t=0} {t+=$5} END {print t}'`	Add up size in bytes of all files in the current directory

grep

`grep -E` *pattern file*	Use extended regular expressions
`egrep`	Short for `grep -E` (people tend to use this instead of `grep`)
`grep` *pattern file*	Search for pattern in file
`grep -v` *pattern file*	Search for not-pattern in file
`grep -c` *pattern file*	Count the number of times pattern appears in file
`grep -i` *pattern file*	Case-insensitive grep
`grep -l` *pattern file*	Show matching file names only ("minus ell")
`grep '^Hello'` *file*	Show all lines beginning with "Hello" in a file
`grep 'Bye$'` *file*	Show all lines ending with "Bye" in a file

find

Multiple arguments can be specified at once to, for example, find all regular files ending in .mp4 that are larger than 1000 MB.

`find . -name` *pattern*	Find files from current directory matching pattern
`find . -name \*foo\*`	Find files with "foo" anywhere in the name
`find . -size +100M`	Find files larger than 100 MB
`find . -type d`	Find files that are directories
`find . -type f`	Find files that are regular files
`find . -type f -exec grep -li` *pattern* `{} \;`	Show names of files containing a pattern

Editors

`vim` *file*	Run the vim text editor
`nano` *file*	Run the nano text editor
`emacs` *file*	Run the Emacs text editor

Command History

UP or **CTRL-P**	Previous command in history (left/right to move cursor)
DOWN or **CTRL-N**	Next command in history
CTRL-R *text*	Search for a previous command containing text
`history`	Look at the command history
`!`*number*	Substitute command number from history here
`!!`	Substitute previous command here
`!^`	Substitute first argument of previous command here
`!$`	Substitute last argument of previous command here
`!*`	Substitute all arguments of previous command here
`!`*command*	Substitute last command beginning with given command
`set -o vi`	Set command line editing mode to vi (vim) mode
`unset HISTFILE`	Don't save history from this bash session

Output

`echo` *text*	Show text on screen, followed by a newline
`printf` *text*	Show text on screen, no newline
`printf "`*text*`\n"`	Show text on screen with a newline

Redirection

command `>` *file*	Redirect output of command into file
command `<` *file*	Redirect input of command from file
command `2>` *file*	Redirect stderr of command to file
command `>` *file* `2>&1`	Redirect both output and stderr of command to file
command `>>` *file*	Append output of command to a file
command `2>>` *file*	Append standard error output to a file

Pipes

command1 `|` *command2*	Pipe output of *command1* into input of *command2*
command1 `2>&1 |` *command2*	Pipe standard error output of *command1* into *command2*

Networking

`ssh` *user@hostname*	SSH to a remote machine
`lftp` *user@hostname*	LFTP to a remote machine
`ftp` *hostname*	FTP (older client) to a remote machine
`telnet` *hostname*	Telnet to a remote machine
`lynx` *url*	Run the Lynx text-based web browser
`links` *url*	Run the Links text-based web browser
`ping` *hostname*	See if a host is reachable over the network

`traceroute` *`hostname`*	Trace all hops a packet takes to reach a host

Process Management

CTRL-Z	Suspend a running foreground job
`jobs`	Show all jobs
`fg`	Put last suspended job in foreground
`fg %2`	Put a specific job in foreground
`bg`	Put last suspended job in background
`ps`	Show all processes running in this terminal (tty)
`ps -u`	Show all processes running for this user attached to a tty
`ps -ux`	Show all processes for this user
`kill` *`pid`*	Kill process (terminate signal)
`kill -9` *`pid`*	Kill process (kill signal—process cannot ignore)
`top`	Text GUI presentation of currently running processes

Symlinks ("symbolic links") and Hard Links

`ln -s` *`file1 file2`*	Make *`file2`* a symlink to *`file1`*
`ln -s` *`/some/path name`*	Make *`name`* a symlink to a file or directory
`ln` *`file1 file2`*	Make *`file2`* a hard link to *`file1`*

Users

`w`	Show users on system
`who`	Show users on system (alternate)
`whoami`	Show your current user name
`last`	Show list of last logged-in users
`su` *`user`*	Switch to another user
`su -`	Run a superuser/root shell
`sudo` *`command`*	Run a command as superuser

Date and Time

`date`	Show date and time
`cal`	Show calendar for this month
`cal 7 1999`	Show calendar for July, 1999
`cal 2015`	Show for 2015

System Info

`uname`	Show OS/system info
`uname -a`	Some more complete OS/system info

Archives, Compression

`tar xvf` *`file`*`.tar`	Extract an uncompressed `tar` archive
`tar xvf` *`file`*`.tar.gz`	Extract a `tar`/`gzip` archive
`tar xvf` *`file`*`.tgz`	Extract a `tar`/`gzip` archive, alternate extension
`tar xvf` *`file`*`.tar.xz`	Extract a `tar`/`xz` archive
`tar cvf` *`file`*`.tar` *`file1`* `[...]`	Create an uncompressed `tar` archive
`tar cavf` *`file`*`.tgz` *`file1`* `[...]`	Create a gzip-compressed `tar` archive
`tar cavf` *`file`*`.tar.xz` *`file1`* `[...]`	Create an `xz`-compressed `tar` archive
`zip -r` *`file`*`.zip` *`file1`* `[...]`	Create a ZIP archive
`unzip` *`file`*`.zip`	Extract a ZIP archive
`gzip` *`file`*	Create a compressed (`.gz`) version of this file
`gunzip` *`file`*`.gz`	Uncompress this file
`zgrep [options]` *`file`*`.gz`	`grep` (see above) a Gzipped file
`zmore` *`file`*`.gz`	`more` (see above) a Gzipped file

Disk Information

`df`	Show all mounted drives and information
`du` *`name`*	Show disk usage for file or directory
`du -k` *`name`*	Show disk usage in KB
`du -m` *`name`*	Show disk usage in MB
`du -h` *`name`*	Show disk usage in a human-readable form
`du -s` *`name`*	Show summary disk usage for a directory
`du -sh` *`name`*	Show summary/Human-readable disk usage

Aliases

`alias ls='ls -l'`	Make it so when you type `ls`, it turns into `ls -l`
`alias p='ping'`	Add `p` as shorthand for `ping`
`unalias ls`	Remove previously-defined `ls` alias.

Variables and Substitutions

Exported variables tend to be capitalized by convention.

`set`	Show all set variables
`VAR``=`*`value`*	Set variable to value
`export` *`VAR`*	Mark variable to be exported to subprocesses
`export` *`VAR`*`=`*`value`*	Set variable, and mark as exported to subprocesses
`$`*`VAR`*	Show the value stored in the variable
`PATH=$PATH:`*`/some/path`*	Append a path to the PATH variable
`vim $(find . -name \*.txt)`	Open in vim all files in all subdirectories with `.txt` extension

`read x`	Read from standard input into variable x
`export PS1='beej$ '`	Set main prompt to `beej$`

Shell Initialization Scripts

`~/.bash_profile`	Contains commands to be executed on the first shell you log in from ("login shell").
`~/.bashrc`	Contains commands to be executed in any interactive non-login shell.
`source ~/.bashrc`	Rerun `.bashrc` in this shell (e.g. after you've made changes to it).
`. ~/.bashrc`	Shorthand for `source`.

Often people just use their .bashrc, and put some code in their `.bash_profile` to run `.bashrc`:

```
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi
```

Starting a Shell

`bash`	Run another Bourne Again shell inside this one
`sh`	Run a Bourne shell
`csh`	Run a C shell
`zsh`	Run a Z shell

More Reading

The Linux Command Line (free PDF) http://linuxcommand.org/tlcl.php

sed one-liners http://sed.sourceforge.net/sed1line.txt

Awk http://www.grymoire.com/Unix/Awk.html
https://www.gnu.org/software/gawk/manual/gawk.html

Prompt customization https://wiki.archlinux.org/index.php/Bash/Prompt_customization

`grep` https://www.gnu.org/savannah-checkouts/gnu/grep/manual/grep.html