Bash for Beginners

useful commands and reference sheets

General Tips

- `Ctrl + C (or ⌘ + C)` will stop the currently running command
- Pressing the up arrow key will retrieve previously used commands
- Typing `sudo` before a command will run it as administrator, provided the logged in user has permission to do so. *Be careful with this!*
- Pressing [tab] will autocomplete the rest of the command or filepath, if unambiguous
- Passing the parameter `--help` to almost any command will give details on how to use it. `man [command]` will give a more in-depth guide.
- The file `~/.bashrc` is automatically executed when the shell is launched. It contains some aliases and environment variables, and you can customize it to add your own.
- Running `source ~/.bashrc` will reload bash with the current changes.

Directory and File Operations

- `pwd` show current directory
- `mkdir [dir]` make directory *dir*
- `rmdir [dir]` delete directory *dir*, if it is empty
- `cd [dir]` change directory to dir
- `cd ..` go up a directory
- `cd -` go back to previous directory
- `mv [file1] [file2]` move *file1* to location *file2*
- `cp [file1] [file2]` copy *file1* to location *file2*
- `rm [file]` delete *file*
- `rm -rf [folder]` delete all contents of folder. *Use with caution!*
- `du -sh ./*` view total disk space used by directories in current directory
- `ls` list files
  - `-a` show all (including hidden)
  - `-S` sort by file size
  - `-l` use long listing format (refer to next slide)
  - `-h` use human readable file sizes (when using -l)

Long Listing Format (and File Permissions Overview)

```
$ ls -l
-rw-rw-r-- 1 hamaon hamaon   15204 Aug 13 22:43 '*_useful code blocks.ipynb'
-rwxrwxrwx 1 ubuntu ubuntu    9784 Jul 11 02:12 '20200709 convert to tx.ipynb'
drwxrwxr-x 2 hamaon hamaon    8192 Jul 24 02:33  pickles
12--3--4-- 5 6----- 7-----    8--- 9----------  10--------------------------
```

1. `-` for file, `d` for directory, `l` for link
2. `r`ead, `w`rite, and e`x`ecute permissions for owner
   `-` means they do not have that permission
3. `rwx` permissions for group
4. `rwx` permissions for everybody else
5. Number of links or directories in a directory
6. The user that owns this file or directory
7. The group that owns this file or directory
8. Filesize in bytes, by default. (See `-h` option)
9. Date of last modification
10. Name of the file or directory

Modifying Ownership and Permissions

- `chown [options] [user] [file]` change the owner of *file* to *user*. Must be run from root or with `sudo`.
  - `[user]` can also specify a group, as `[user:group]` or just `[:group]`

  `-R` apply chown to subdirectories recursively
- `chmod [options] [mode] [file]` set the file permissions of *file* to *mode.*

The shortest way to define `[symbols]` is to use octal notation to encode permissions for each access group in a single number. The access groups in order are: user, group, and others.

For example, `$ chmod 750 output.txt` sets the permissions of *output.txt* to `rwxr-x---` i.e. full permissions for the owner, read/execute for the group, and none for everyone else.

r/w/x	binary	octal
---	000	0
--x	001	1
-w-	010	2
-wx	011	3
r--	100	4
r-x	101	5
rw-	110	6
rwx	111	7

IO Redirection

- `cmd < file` input of *cmd* from *file*
- `cmd > file` output of *cmd* to *file*
- `cmd 1> file` standard output (stdout) of *cmd* to *file*
- `cmd 2> file` error output (stderr) of *cmd* to *file*
- `cmd >> file` append output of *cmd* to *file*
- `cmd1 | cmd2` pipe stdout of *cmd1* to *cmd2*'s input

Searching (for) Files

- `grep [options] [pattern] [files]` search for *pattern* in *files.*
  - `-i` case insensitive search
  - `-r` recursive search
  - `-o` show matched part of file only
- `find [/dir/] -name [name]` find files starting with *name* in *dir*
- `find [/dir/] -regextype sed -regex [pattern]` find files matching *pattern* in *dir*

Patterns are defined using regex, which allows for rather sophisticated searches.

- `^` start of string or line
- `$` end of string or line
- `\` escape character
- `.` any single character
- `\s` any whitespace character
- `\w` letter, number, or underscore
- `*` zero or more of preceding character

A more thorough guide can be found here https://www.computerhope.com/unix/regex-quickref.htm

Screen usage

Screen makes multiple terminals that can be detached and reattached to from within the same shell. This is useful when connecting to a remote computer, as if a disconnection happens the session can be resumed without interrupting running commands.

- `screen -S [name]` start a new screen with session *name*
- `screen -ls` list running sessions/screens
- `screen -r [name]` attach to a running session with *name*.
- `screen -X -S [name] quit` kill session with *name*
- Ctrl + A + D (or ⌘ + A + D) detach from current session