Files, Directories, Git, and the Command Line

File management, directory structure, and git

Command Cheatsheet

A list of important commands with descriptions of useful options

cd

change directory

-c
: ouput a diagnostic for files processed **only** when a change is made

-R
: change files and directories recursively

-v
: output a diagnostic report for every file processed

cp

copy file

-a
: Preserves specified attributes such as ownership, timestamp etc. does not follow symbolic links in SOURCE.

-r
: copy directories recursively

-s
: create a symbolic link instead of actually copying

find

Look for a file or folder live.

syntax for finding a specific file
: 'find top-directory -name filename'

-P
: never follow symbolic links (default behaviour)

-L
: Follow symbolic links

ls

List contents of a directory

-a
: Do not ignore entries starting with .

--author
: When used with -l it prints author of each file

-g
: Like l but ignores owner

-G
: In long list, don’t print group names

-l
: Use long list format (show permissions etc)

-d
: List directories themselves, not their contents

mkdir

create a new directory

mv

Move or rename a file

--backup
: make a backup of each existing destination file

-u
: move only when the SOURCE file is newer than the destination file or when the destination file is missing

pwd

print working directory

-P

Avoid all symlinks

rm

Remove file

-d

Remove empty directories

-f

Force, don’t confirm

-i

Prompt before every removal

-I

Prompt once before moving 3 files

-r, -R

Remove directories and their contents recursively

chmod

change permissions on a file or directory.

- For a regular file, a dash appears infront of the 9 bit permission indicator. In this position, d (directory), l (symbolic link), b (block device), c (character device), s (socket), p (pipe)
- 9 bit permission indicator is ---(owner)---(group)---(others)

  Permission Codes

  read = 4, write = 2, execute = 1.

  - 7 (4 + 2 + 1) is full permission
  - 0 is no permission

- Examples

  `chmod 744 file`

  rwxr—r-- means you have rwx and everyone else has r--

  `chmod 755 file`

  rwxr-xr-x means you have rwx and everyone else has r-x

  `chmod 777 file`

  rwxrwxrwx everyone has total access to this file `chmod 644 file`

rw-r—r-- you can read and write but not execute, everyone else can just read

- Changing permissions with chmod letters
  - Use + and - to turn permissions on and off
  - u (user), g (group) o (other) a (all)
  - r (read), w (write), x (execute)
- Removing Permissions examples

`chmod a-w file`

r-xr-xr-x

`chmod o-x`

rwx-rwx-rw-

- Adding Permissions Examples

`chmod u+rw file`

rw-------

`chmod a+x`

--x—x—x

`chmod ug+rx file`

r-xr-x---

Flags

-R

change permissions recursively. Use this with chmod letters to change permissions for multiple files

- `chmod -R o-w $HOME/myapps` removes write permissions for others for all files under the myapps directory

Default permissions

Files for regular users are set to rw-rw-r-- and directories are set to rwxrwxr-x. Root users file and directory permissions are set to rw-r—r-- and rwxr-xr-x.

chown

change file or folder ownership

- Change the user `chown userName fileName`
- change the user and group `chown userName:groupName fileName`
- change just the group `chown :groupName fileName`

- change directory ownership `chown ownerName /directoryName`

Flags

-R
: change recursively. This is especially useful for when changing a folder's ownership. It will change ownership of all files within that folder.

Reading Directory Information

Extra information on how to read file permission information

Example

`drwxr-xr-x 2 owner group 1024 Jan24 12:17 test`

Breakdown

d
: directory

rwxr-xr-x
: permissions

  - r read
  - w write
  - x execute

owner
: owner of the files

group
: user group that the owner belongs to

jan24 12:17
: last accessed

test
: name of the file or folder

Metacharacters and Operators

Metacharacters can be used to match desired files more effeciently.

File Matching Characters

*****
: Matches any number of characters

?

Matches any one character

[...]

Matches any one character between the brackets. Includes hypen-separated range of letters or numbers. Will find anything that has whats in the [] in it. Use with * to find lots of files.

Example

`ls [a-g]*`

- This will list any files that begin with letters from a through g.

File direction metacharacters

Directory Management Best Practices

See information on course specific instructions in SAIT-wbdv directory notes following this section.

_ dirname

use the _ to denote special folders. anything with a _ to start will be at the top of the directory. Read only and important projects are good candidates for this.

uppercase naming

only use this for files and folders that really need to stand out. like a README.md

Git Notes

- Cannot clone a repo into a repo. This is why trying to clone a github account doesn't work.
- If a repo is stored too far up the directory tree, git will try to keep track of everything in that tree.

Commands and useful options

git add

Add file contents to the index

-A

all. Update index not only where the working tree has a file matching but where the index already has an entry. adds, modifies, and removes index entries to match the working tree.

-f

Allow added otherwise ignored files

--ignore errors

If some files couldn't be added because of errors indexing them, do not abort the operation **and** continue to add others.

-n

Don’t actually add the files, just show if they exist and/or will be ignored

—refresh

Don’t add the files, but only refresh their stat() information in the index.

-u

update index just where it has an entry matching

-v

show output

git checkout

Switch branches or restore working file tree.

- checkout will update HEAD to set the specified branch as the current branch `git checkout [branch]`

  -b

  Creates a new branch as if git-branch was called and then checksout to it

  -B

  Like -b but if the branch already exists it resets it to the start point. It is like running git branch -f.

  -m

  when switching branches, if local modifications to one or more files are different between current branch and the branch to which you are switching, the command refuses to switch branches in order to preserve your modifications.

  - helpful for stopping merge conflicts while hopping between Branches.

  -t

  when creating a new branc, set up "upstream" configuration.

git commit

Record changes to the repository. - TODO learn advanced features later.

-a

automatically stages files that have been modified and deleted

-m

use the message that follows the flag within "" as the commit message

git diff

Shows changes between commits, commit and working tree. TODO find some tutorials on how to do this in specific contexts.

`git diff [option] [--] [path]`

: view changes that you have made relative to the index(staging area for next commit)

`git diff [options] --no-index [--] <path> <path>`

: compare two given paths on a filesystem. --no-index can be ommitted when the command is being ran in a working tree controlled by git and at least one path pointing outside the working tree.

--color-words

: highlight changes by tokenizing added and removed linkes by whitespace

git ls-tree <branchName>

list files in a branch

git merge

Join two or more development histories together. Incorporates changes from named commits since the time their histories diverged from the current branch into the current branch.

--abort

: will abort the merge process and try to reconstruct the pre-merge state.

git push

update remote refs with local refs.

--all

: push all branches

--delete

: all listed refs are deleted from the remote repository

--dry-run

: Do everything except actually send the updates

-v

: run verbosely

git rm

Remove a file from the working tree. It will also remove it from the system that it is on.

-f

override the up-to-date check

-r

Allow recursive removal when a leading directory is given.

git status

shows the working tree status

-b

show the branch and tracking info

-b

give output in short-format

-u

show untracked files. Options are **no normal all**

-v

shows textural changes committed (like git diff --cached) as well as names of files

Definitions

Read-only Repo

A remote repository that you do not have permission to change

Repository (aka repo)

A collection of commits, branches and tags to identify commits. Tracked with git

Local Repository

A repo downloaded on your machine

Remote Repository

A repo located on another machine

SAIT-wbdv Directory Notes

default-dirs/assignments/

save local repositories for assignments in this folder. Each assignment should have it's own repo.

defaults-dirs/in-class

daily code. find starter code from each class here.

defaults-dirs/projects

a directory to store project repos that can't be categorized by course or day. Use this for portfolio projects.