Lecture 4 – Advanced git and Bash

Learning Objectives:

2. **Become familiar with the use of Bash, shell programming, and console editors**

   2.2 Understand the use of basic functions in Bash shell.

   2.3 Become proficient in the use of a console editor.

   2.4 Understand the syntax Bash commands.

   2.5 Understand the use of shell wildcards and regular expressions.

   2.6 Learn the use of advanced Bash commands (grep, awk).

4. **Produce code that is reproducible and produces results that are replicable.**

   4.1 Learn commands of git.

   4.4 Learn how to work together on a common repository.

Repository Options

Cloning a repository:

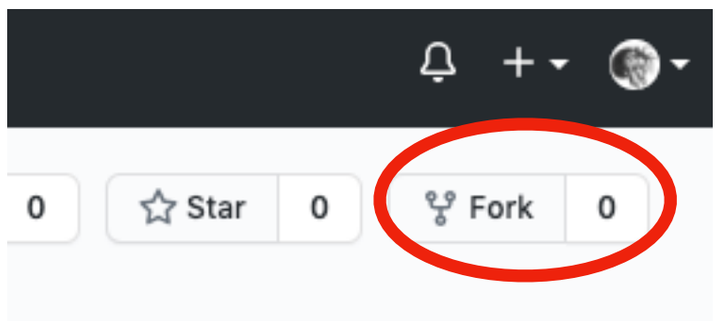
`git clone https://github.com/CS-510-Fall-2020/git-practice-repo.git`

- makes a copy of the repository
- can download (pull) updates from repository
- make changes (push) to main branch (depends on permissions)

Branching a repository: `git branch yourname-branch`

- creates a snapshot of repository at wherever HEAD pointer is
- can make commits, pull/push, independent of main branch
- can merge branch into main branch later!

Forking a repository:



- creates a copy of the repository into a separate repository, controlled by you
- can still get updates from main repository (sync), requires merge

Repository Options: branches

```
git log --graph --all
```

```
CPSC-WALDROP-MBP:git-practice-repo waldrop$ git hist
* 9945021 - Mon, 14 Sep 2020 09:17:35 -0700 (7 minutes ago) (test-branch)
|          Refrigerator pickle recipe added to test-branch - lindsaywaldrop
| * 2d235d0 - Fri, 11 Sep 2020 13:23:55 -0700 (3 days ago) (HEAD -> master)
|/           Adding another couple files for 04 Bash lecture - lindsaywaldrop
* 33fd874 - Thu, 10 Sep 2020 12:46:56 -0700 (4 days ago) (origin/master, origin/HEAD)
|          Adding some additional practice files, sample shell script - lindsaywaldrop
* 85262df - Thu, 10 Sep 2020 10:25:17 -0700 (4 days ago)
|          Adding some additonal items and practice files - lindsaywaldrop
* c243895 - Mon, 31 Aug 2020 09:21:18 -0700 (2 weeks ago)
|          Adding some notes on growing tomatoes - lindsaywaldrop
* dba5223 - Fri, 28 Aug 2020 18:50:19 -0700 (2 weeks ago)
|          adding some additional files - lindsaywaldrop
* f393629 - Fri, 28 Aug 2020 17:51:48 -0700 (2 weeks ago)
|          adding additional recipes, making new folder - lindsaywaldrop
* b5e14d2 - Fri, 28 Aug 2020 17:07:27 -0700 (2 weeks ago)
|          Adding some recipes, bread and waffles - lindsaywaldrop
* 62733d9 - Fri, 28 Aug 2020 16:48:49 -0700 (2 weeks ago)
           Initial commit - Lindsay Waldrop
CPSC-WALDROP-MBP:git-practice-repo waldrop$
```

What are the pointers telling you?

Why are there two HEADs?

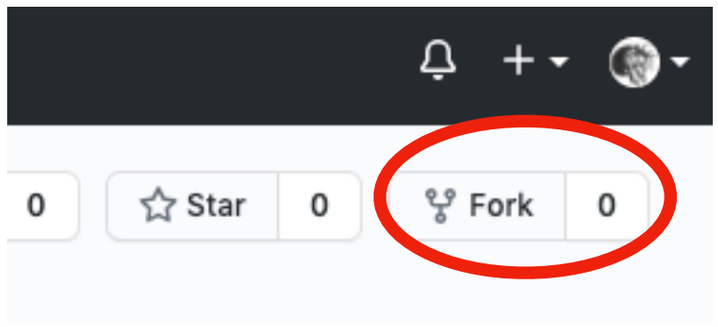
Merge a branch:

```
git checkout master

git merge test-branch
```

Repository Options: forks

Make a fork of the git-practice-repository on Github



I will push a commit to the original repo on Github.

Group work: How do you update your folk to include my new commit?

Bash command structure

Basic structure: `command` *`flags arguments`* `rm -r directory/`

- Command: command you wish to call. `rm -r directory/`
- Flags: options beyond command defaults. `rm -r directory/`
- Arguments: items you wish to act upon. `rm -r directory/`

Important features:

- **spaces**: spaces separate commands, flags, and arguments! They are really important!!
- **capitalization**: bash commands and options are case sensitive. The flags -a and -A may be completely different options!
- **working directory/path**: need to be specified, pay attention to where you are! Any argument that accepts a file will accept a path.

Other useful Bash commands

Command	Description	Example
`touch`	create a new, empty file	`touch example.txt`
`head`	print out first 10 lines of a file	`head allpara.txt`
`tail`	print out last 10 lines of a file	`tail allpara.txt`
`less`	read long files	`less allpara.txt`
`wc`	count number of lines, words, characters in a file	`wc allpara.txt`
`basename`	extracts base file/directory name from path	`basename $HOME`
`diff`	shows differences between two files	`diff allpara.txt allpara2.txt`

Other useful Bash commands

Command	Description	Example
`ps`	examine process information	`ps -ef`
`kill`	kill a process with processid	`kill processid`
`nohup`	run a process in the background	`nohup command &`
`chmod`	change file permissions	`chmod +x setparameters.sh` `./setparameters.sh`

Shell Wildcards

`?`	match any 1 alphanumeric character
`*`	match 0 to any number of alphanumeric characters
`[Bb]`	match character (ignore case)
`[0-9]`	match number sequence
`[A-Z]`	match letter sequence
`[A-Z,a-z]`	match letter sequence (ignore case)
`[^0-9]`	negate a match sequence

There are more!

Regular Expressions

NOTE: These are a little different than shell wildcards!

`.`	match any 1 alphanumeric character
`*`	match 0 to any number of the pervious alphanumeric character
`.*`	match 0 to any number of alphanumeric characters
`+`	match 1 or more of the pervious alphanumeric character
`[ ]`	square brackets work the same way as in shell
`^`	search at beginning of line
`$`	search at end of line
`\*`	escapes special character to interpret literally

There are more!

Advanced Bash Commands: grep

`grep` Find a specified pattern. Patterns can be literal or regular expressions. (Note: shell wildcards WILL NOT work in grep.)

Find lines with word “kale”: `grep “kale” kale.txt`

Include line numbers: `grep -n “kale” kale.txt`

Ignore case within pattern: `grep -i “kale” kale.txt`

Return only number of times: `grep -ni “kale” kale.txt | wc -l`

Specify an anchor using ^: `grep -ni “^grow” kale.txt`

Return only line number: `grep -ni “^grow” kale.txt | cut -d : -f 1`

Capture *n* lines after pattern: `grep -ni -A 10 “^grow” kale.txt`

How would you create a file with only growing instructions?

Advanced Bash Commands: grep

Group work: In the practicefiles folder, there is a csv file NW1.csv which is a 28 MB file with 132k lines! It's really large and contains two separate components of a recording stacked on top of each other. The column headers for each data set start with:

```
Frame,Sub Frame,Fx,Fy,Fz,Mx,My,Mz,Cx,Cy,Cz,Fx,Fy,Fz,Mx,My,Mz,Cx,Cy,Cz, …
```

```
Frame,Sub Frame,X,Y,Z,X,Y,Z,X,Y,Z,X,Y,Z,X,Y,Z,X,Y,Z,X,Y,Z,X,Y,Z,X,Y,Z, …
```

Break NW1.csv into two files: NW1-1.csv that contains the first data set and NW1-2.csv that contains the second data set.

NW1.csv

```
header info
Frame,Sub Frame,Fx,Fy,Fz,…
 numbers, numbers, numbers
 numbers, numbers, numbers
    ...
 numbers, numbers, numbers
Frame,Sub Frame,X,Y,Z, …
 numbers, numbers, numbers
 numbers, numbers, numbers
    ...
 numbers, numbers, numbers
```

NW1-1.csv

```
Frame,Sub Frame,Fx,Fy,Fz,…
 numbers, numbers, numbers
 numbers, numbers, numbers
    ...
 numbers, numbers, numbers
```

NW1-2.csv

```
Frame,Sub Frame,X,Y,Z, …
 numbers, numbers, numbers
 numbers, numbers, numbers
    ...
 numbers, numbers, numbers
```

More Information

More on Regular Expressions:
https://www.cyberciti.biz/faq/grep-regular-expressions/

More on grep:
https://www.cyberciti.biz/faq/howto-use-grep-command-in-linux-unix/