Git Class Note

Check if you have git in your computer

```
git --version
```

See the commands

```
git help
```

Configuration

```
git config --global user.name “<name>”
git config --global user.email “<email>”
git config --global core.editor “<text editor>”
git config --list
git config --global --edit
```

Associating text editors with Git

Using Sublime Text as your editor

- Modify path to subl.exe according to your computer!

```
git config --global core.editor "'C:\Program Files\Sublime Text\subl.exe' -w"
git config --global core.editor "notepad"
```

Delete individual git config

```
git config --system --unset <key>
git config --global --unset user.name
git config --system --unset user.email
git config --system --list
```

Create a local repo

```
git init
```

Get the content of a remote repo

```
git clone <url>
```

See the status of your repo

```
git status
```

Stage files

```
git add <filename>
git add .
```

Record the current state

- The git stash command takes your uncommitted changes (both staged and unstaged), saves them away for later use, and then reverts them from your working copy.
- At this point you're free to make changes, create new commits, switch branches, and perform any other Git operations; then come back and re-apply your stash when you're ready.

```
git stash
git stash list
git stash pop
git stash apply <stash>
git stash clear
git stash drop <stash>
```

Commit

```
git commit
git commit -m “message”
git commit -am “message”
git commit --amend
```

See commit logs

```
git log
git log --oneline
git log --since=<date>
git log --since=2.weeks
git log --since="2008-01-15"
git log --since="2 years 1 day 3 minutes"
git log --until=<date>
git log -- <file>
```

See code differences

```
git diff
```

Clean

```
git rm
git rm --cached
git checkout
```

Branches

Rename the current branch

```
git branch -m <branch>
```

Rename the default branch

```
git config --global init.defaultBranch main
```

Create a new branch

```
git branch <branchname>
```

Switch to a branch

```
git checkout <branchname>
```

Create and switch to a branch

```
git checkout -b <branchname>
```

Delete a local branch

```
git branch -d <branchname>
git branch -D <branchname>
```

- Use -D instead if you want to force the branch to be deleted, even if it hasn't been pushed or merged yet.

See branches

```
git branch
git branch -r
git branch -a
```

Merge two branches

```
git merge
```

Remote Ops

Connect to remote repo

```
git remote add origin <url>
```

Get the latest version

```
git fetch
git pull
```

Upload your commit

```
git push -u origin master/main
git push
```

Delete a remote branch

```
git push origin --delete <branchname>
git push origin :<branchname>
```

gitignore

- A good documentation about gitignore