**`git status`** - find out what's going on in the current directory (assuming it is tracked by git, of course); this is kind of the Git equivalent of constantly typing “ls” and “pwd” all the time in the CLI

When you switch machines, you’ll need to get the code from the remote repository to your local repo. If this is your first time at a new machine, you **clone** to add a new local copy of the repo. If you’ve got changes in the cloud that you want to add to an *existing* local repo, you **pull** them down (*before* you start working on changes!):

**`git clone [repo url from github/bitbucket]`** - grab the repository and put it on your machine in a directory, tracked – your first step when you start working with existing code on a new machine

**`git pull origin [branch_name]`** - pull down any changes that are in the repository stored in the cloud, in the [branch_name] branch; usually, git will ask you to commit any changes after you do this – unless you get fancy about using branches (good practice, but not required for DAT-119 or 129), you’ll always type **`git pull origin master`**

After you’ve made changes in your local copy of the repo, here is how you push it back up to the cloud (do all three steps, in order):

**`git add [filename]`** - add this file to the next commit (add all changed files with **`git add *`**)

**`git commit -m "commit message goes here"`** - commit the changes you've made; you can make multiple commits before you push! also, try to leave yourself helpful messages when you make commits (if you like to do really verbose commit messages, you can leave off the -m flag and the message, and it will pull up whatever your default editor is on the current machine; if it's vi/vim and you aren't a vi user, **`:q!`** is how you quit without saving, and **`git config --global core.editor "nano"`** will set nano as your default text editor ;))

**`git push origin [branch_name]`** - push any commits you've made up to the repo in the cloud; you can specify any branch, but by default you’ll use **`git push origin master`**

If you are comfortable with the above and want to branch out (hehe), you can start saving your solid code in some kind of main branch (“master” or “deployment” or whatever you want to call it) and code that’s under development, maybe with new features or bugfixes under other branches (“development” and “feature” are common choices):

**`git branch [branch_name]`** - make a branch with name [branch_name]

**`git checkout [branch_name]`** - switch to branch [branch_name]