Git Workflow

Overview

Git is a system that lets you:

- Store versions of your files at key points that you want to be able to get back to
- Revert to previous versions of your files if need be
- Collaborate with other people making changes to the same files you are working on
- Share your work with the world

Your code (or other work) is kept in a **repository**. There are typically at least two copies of this repository:

1. One on a server on the internet; in this class we will keep a copy of our code on GitHub
2. One on the computer where you are editing the files; in this class I will suggest that you keep these files on Gryd.
3. If other people are also working on these files, they may have their own copies of the respository.

There are two main steps to using Git. Each has sub-steps that we'll outline below.

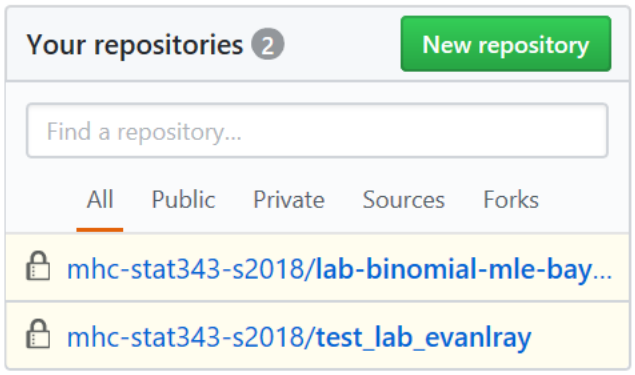
1. Initialize a git repository. In this class, I will set up starter repositories for you on GitHub, and you will **clone** these to the computer where you will be working on the files.
2. Ongoing updates. Make changes to files, **add** them to a staging area, **commit** them to a local repository, and **push** them to a remote repository

Note that these steps work if you are the only person working on a project. The workflow for collaborating on projects and sharing code via Git is more involved; we will get there later in the semester.

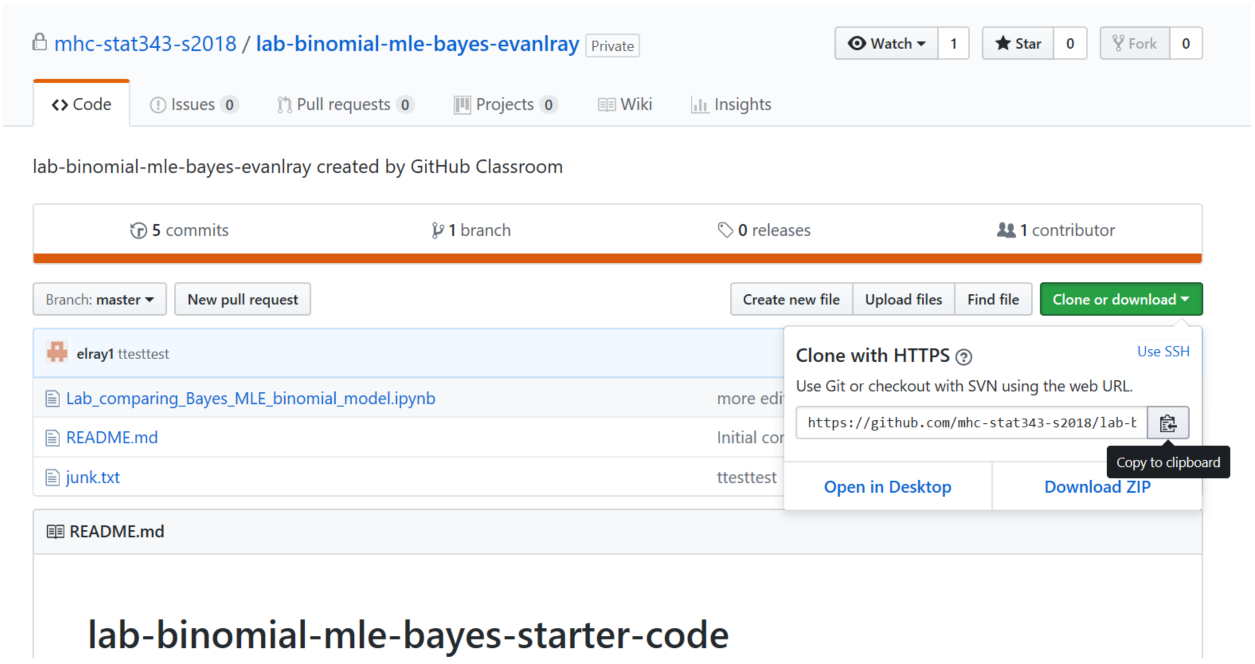
Initial Set-Up: Cloning a Repository from GitHub

In this class, I will set up repositories with starter code on GitHub. Your first step is to **clone** that repository, which downloads it from GitHub to the computer where you will be editing the files. In this class, we will edit our files on the website Gryd, so we'll be cloning the files to Gryd.

1. Sign into GitHub.com in your web browser.
2. Find the repository you want to clone. If you don't have a link to the repository's page, there is a search box to find repositories on the main page of GitHub:



3. In this example, I will work with a repository called "lab-binomial-mle-bayes-evanlray". On the GitHub page for the repository you want to clone, click the green "Clone or download" button, and copy the web address there:



mhc-stat343-s2018 / lab-binomial-mle-bayes-evanlray Private

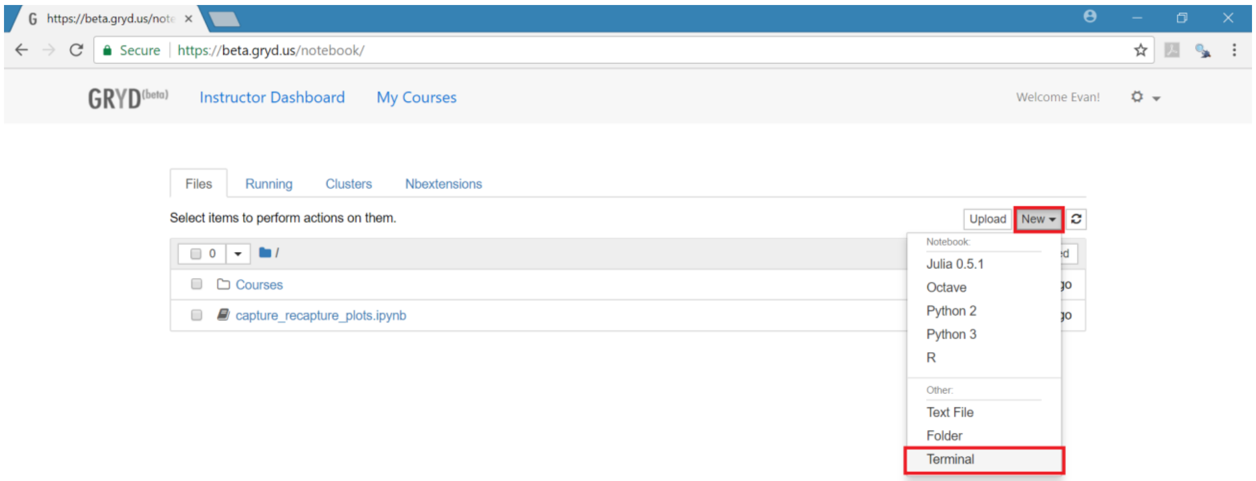
lab-binomial-mle-bayes-evanlray created by GitHub Classroom

Clone with HTTPS

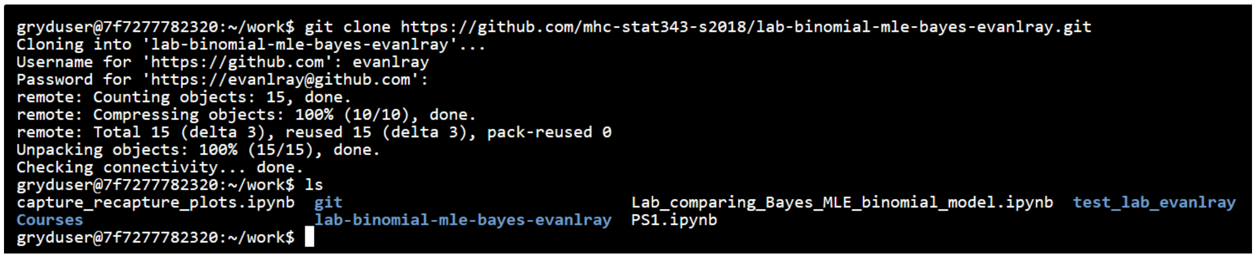
Use Git or checkout with SVN using the web URL.

```
https://github.com/mhc-stat343-s2018/lab-b
```

4. Sign in to Gryd.us in a new browser tab, and open a new terminal session:



5. In the terminal type “git clone ”, and paste in the web address for the GitHub repository that you copied in step 3. Hit enter, and enter your username and password for GitHub at the prompts (as you type your password, no characters will appear).

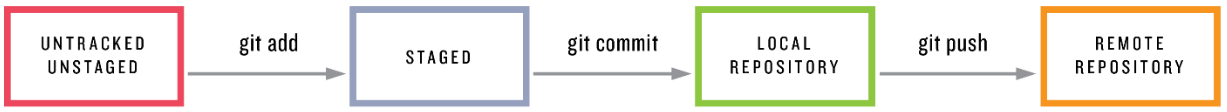


```
gryduser@7f7277782320:~/work$ git clone https://github.com/mhc-stat343-s2018/lab-binomial-mle-bayes-evanlray.git
Cloning into 'lab-binomial-mle-bayes-evanlray'...
Username for 'https://github.com': evanlray
Password for 'https://evanlray@github.com':
remote: Counting objects: 15, done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 15 (delta 3), reused 15 (delta 3), pack-reused 0
Unpacking objects: 100% (15/15), done.
Checking connectivity... done.
gryduser@7f7277782320:~/work$ ls
capture_recapture_plots.ipynb  git                              Lab_comparing_Bayes_MLE_binomial_model.ipynb  test_lab_evanlray
Courses                        lab-binomial-mle-bayes-evanlray  PS1.ipynb
gryduser@7f7277782320:~/work$
```

6. If you want to verify that the files are there, you can type “ls” to list the files and folders in your present working directory, as in the screenshot above.

Ongoing Updates: Add, Commit, and Push to GitHub

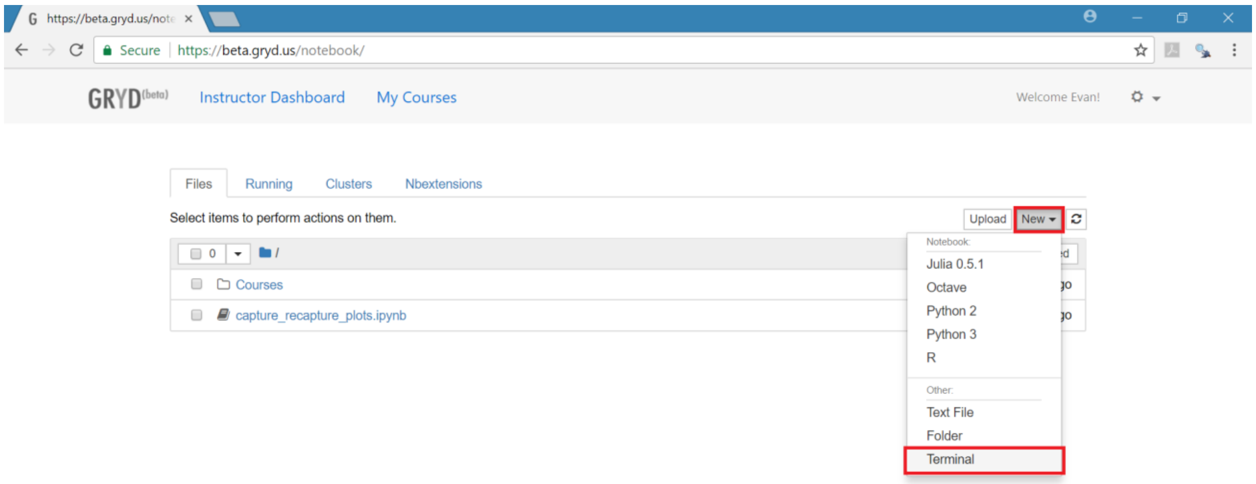
The graphic below illustrates the four phases involved in committing changes you make to your files to your local repository and pushing them to github:



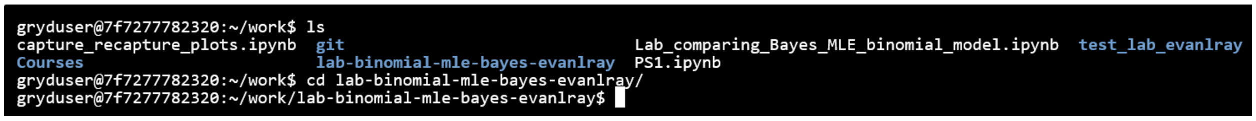
When you create a new file or edit an existing file on your local computer (or, in our class, on Gryd), those changes are not backed up and are not available for other people to see. At this point, these files are **unstaged**. When you have reached a point where you are ready to save some changes you have made, you will **add** them to a staging area, **commit** them to a local repository, and **push** that commit to a remote repository. You will usually want do this after you have finished a particular programming task that is more or less self-contained.

These steps are outlined in more detail below. In this example, I have edited the file named "Lab_comparing_Bayes_MLE_binomial_model.ipynb", which is in a repository named "lab-binomial-mle-bayes-evanlray". I now want to commit those edits to the repository.

1. Open up a terminal on Gryd:

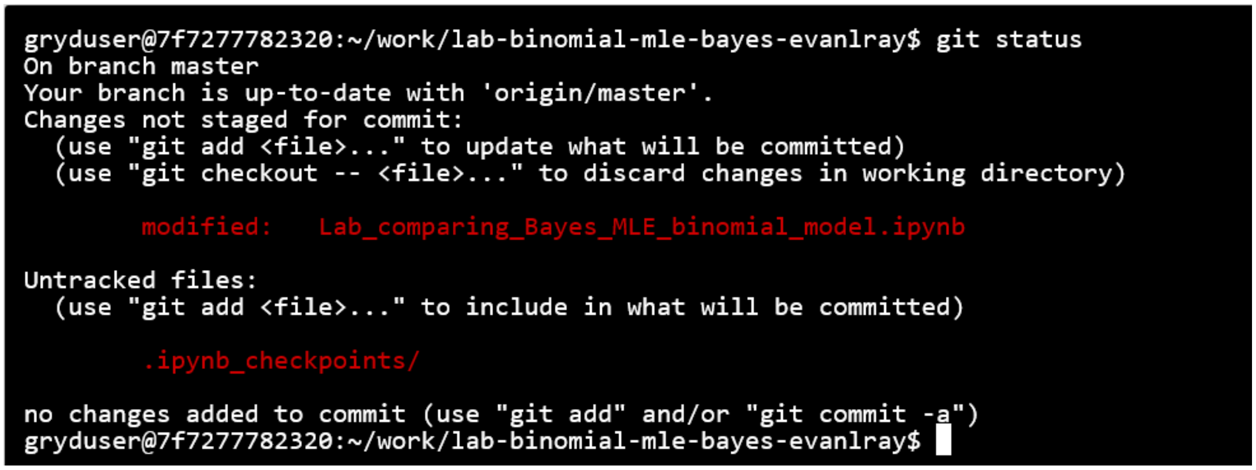


2. Navigate to the directory for the repository you are working with (in this example, "lab-binomial-mle-bayes-evanlray"). At any point in the terminal, you can:
   - use the **ls** command to list files and directories in your present working directory
   - use the **cd** command to change directories



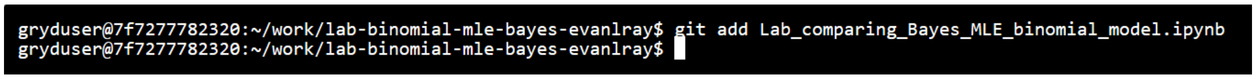
```
gryduser@7f7277782320:~/work$ ls
capture_recapture_plots.ipynb  git                              Lab_comparing_Bayes_MLE_binomial_model.ipynb  test_lab_evanlray
Courses                        lab-binomial-mle-bayes-evanlray  PS1.ipynb
gryduser@7f7277782320:~/work$ cd lab-binomial-mle-bayes-evanlray/
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$
```

3. You can check the current status of files in the repository with **git status**. At this point, you will see something like the output below. The top section is showing one file ("Lab_comparing_Bayes_MLE_binomial_model.ipynb") that has been modified. A previous version of that file was committed to the repository, but new changes have been made to that file and have not yet been committed. The lower section shows a second file (".ipynb_checkpoints/") that is "Untracked" – that is, it was never in the repository.



```
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   Lab_comparing_Bayes_MLE_binomial_model.ipynb

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        .ipynb_checkpoints/

no changes added to commit (use "git add" and/or "git commit -a")
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$
```

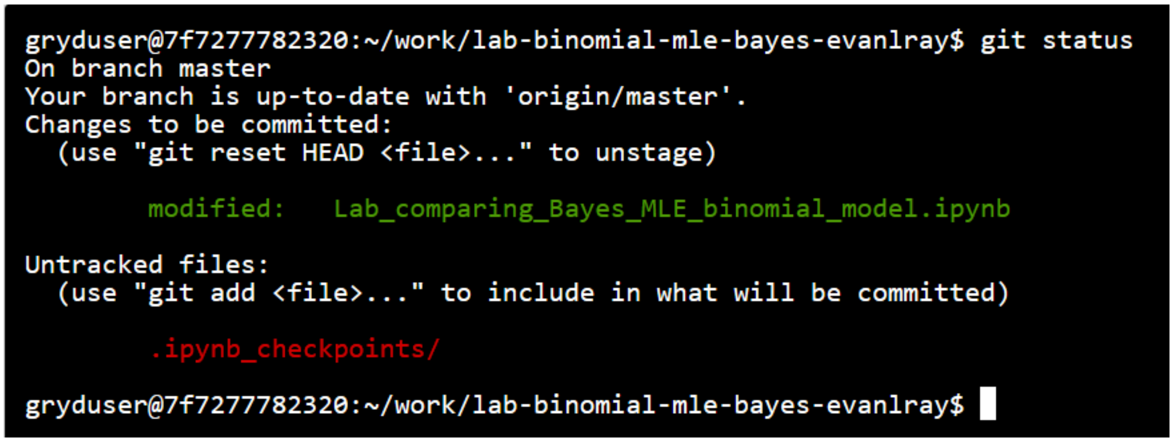
4. To add the Lab_comparing_Bayes_MLE_binomial_model.ipynb file to the staging area, use **git add Lab_comparing_Bayes_MLE_binomial_model.ipynb**



```
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$ git add Lab_comparing_Bayes_MLE_binomial_model.ipynb
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$
```

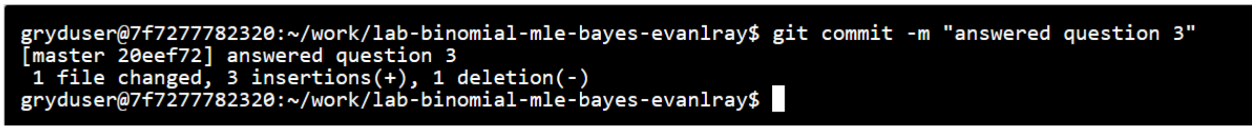
You can add multiple files to the staging area if you have made changes to several files that you want to save together.

5. To check that all of the correct files are in the staging area before you commit them, use **git status**
The files to be committed will appear in green text under a heading of "Changes to be committed:"



```
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   Lab_comparing_Bayes_MLE_binomial_model.ipynb

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        .ipynb_checkpoints/

gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$
```

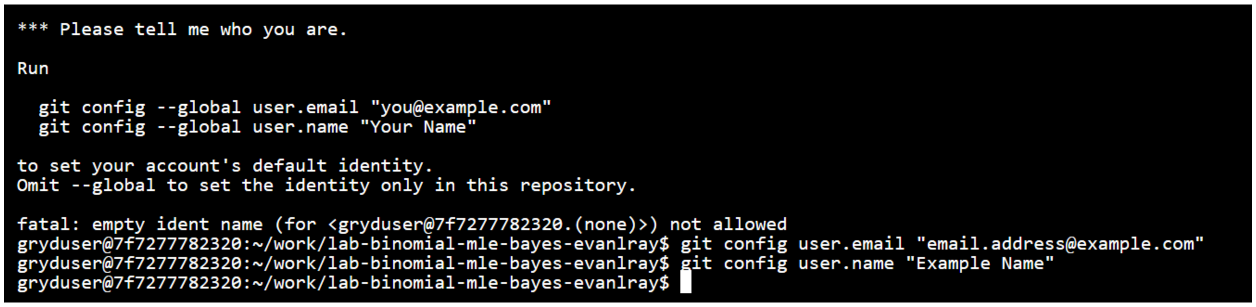
6. To **commit** the files in the staging area to your local repository, enter the following:
   git commit –m "description of the changes you have made"



```
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$ git commit -m "answered question 3"
[master 20eef72] answered question 3
 1 file changed, 3 insertions(+), 1 deletion(-)
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$
```

After entering this command you may see an error saying "Please tell me who you are." If you see that message, you will have to enter the following commands, substituting in the email address you used when you registered for GitHub and your name:

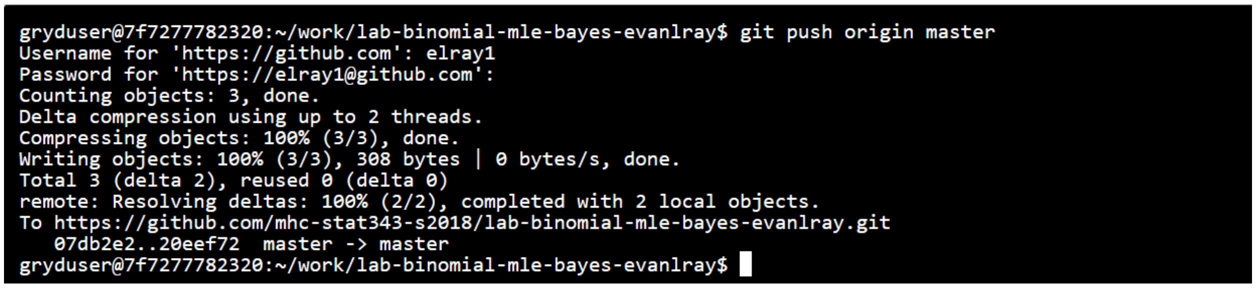
git config user.email "email.address@example.com"
git config user.name "Example Name"



```
*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: empty ident name (for <gryduser@7f7277782320.(none)>) not allowed
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$ git config user.email "email.address@example.com"
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$ git config user.name "Example Name"
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$
```

You will then have to re-run the **git commit –m "description of the changes you have made"** command above.

7. To push the commits in your local repository to the remote repository, enter the following:
   git push origin master



```
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$ git push origin master
Username for 'https://github.com': elray1
Password for 'https://elray1@github.com':
Counting objects: 3, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 308 bytes | 0 bytes/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/mhc-stat343-s2018/lab-binomial-mle-bayes-evanlray.git
   07db2e2..20eef72  master -> master
gryduser@7f7277782320:~/work/lab-binomial-mle-bayes-evanlray$
```

In this command, "origin" is a key word that has been configured to tell git where the remote repository is, and how to communicate it. "master" is the name of the branch where your changes have been made. A branch in git is a way to organize development of new features or code that you're not sure you want to keep permanently yet. We will talk about branches in git later in the semester. The default is to work on the "master" branch, so for now the exact command above will work.