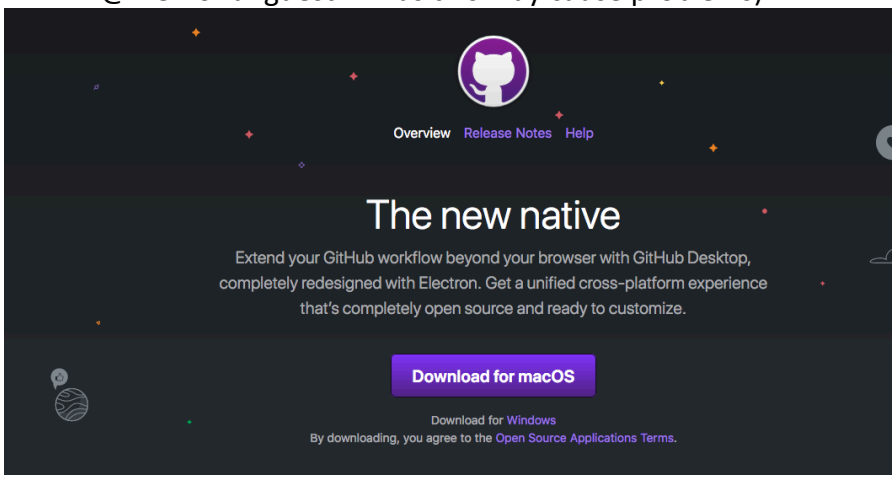
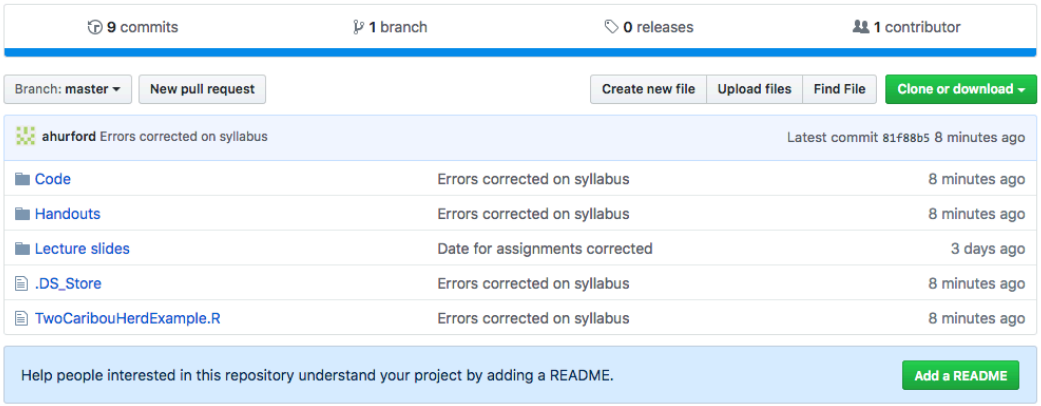
1. Installing Github Desktop

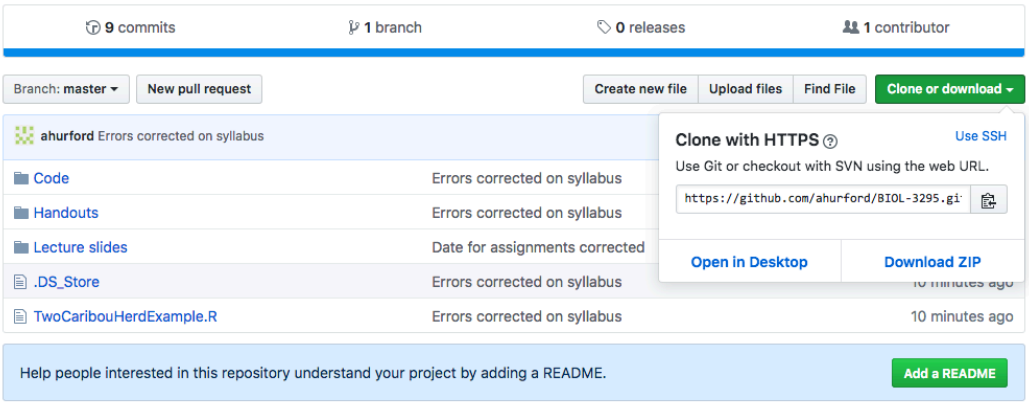
1. Install Github Desktop https://desktop.github.com/ (Note do not do with over the @memorial-guest wifi as this may cause problems)



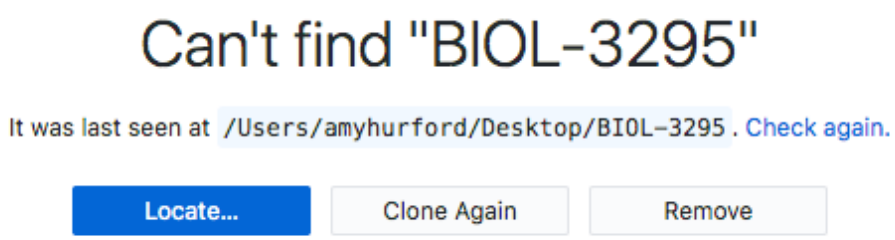
2. Go to the class github repository https://github.com/ahurford/BIOL-3295
3. Click on the green **'Clone or download'** button



4. Click the left option **'Open in Desktop'** (if you have not installed Github Desktop as per step 1, then choose Download ZIP or just view the files on the Github website).



5. My computer then gives a 'Launch in Application' window, and I select 'GitHub Desktop' and 'Open link', but you may get something different depending on your computer.
6. My Github Desktop now displays the following screen:

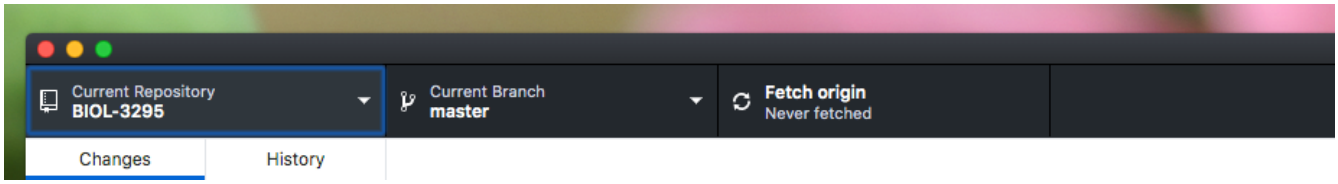


This is because I deleted my BIOL-3295 folder for the purpose of this example. If you haven't downloaded before you'll likely have a different screen.

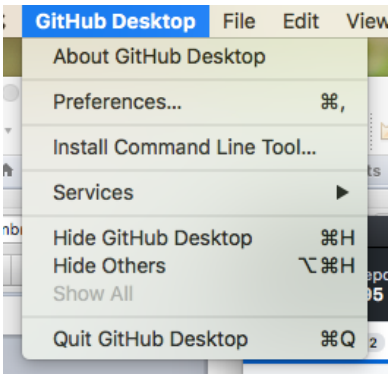
- "Locate…" means the software suspects that I may have moved the "BIOL-3295" folder from my Desktop (where this folder was last seen), and asks if I would like to guide it to the new directory where I moved it. In my case, I put the file in the trash, so it won't help to provide a location.
- **"Clone Again"** is the option I will choose. This will copy all the folders from the Github repository to a folder on my Desktop (I believe I had previously specified Desktop as the location that I wanted these files to appear in and I'm not asked for the location again. If you're having trouble locating where files are downloaded to perform a search on your computer to find the files just as you would any other lost files).

7. After I click **"Clone Again"** a folder "BIOL-3295" appears on my Desktop and this folder contains all the latest files for the class.

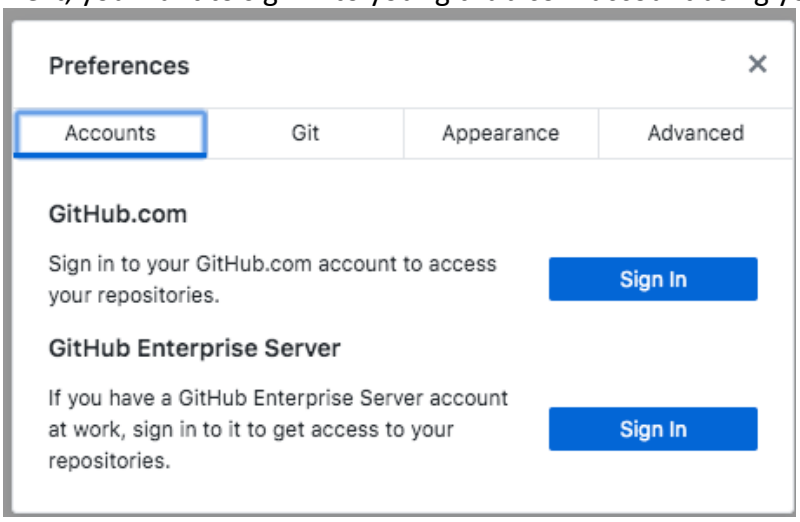
8. The usefulness of this will become apparent later, as it is very easy to get the most recent course material using Github Desktop. With the current repository selected as "BIOL-3295" and the Current Branch as "master", click on **"Fetch origin"** and any new files will be downloaded to your BIOL-3295 folder.



9. The final step is to link your Github account to your Github Desktop. Sign-up for Github here: https://github.com/join. Next in the **"Github Desktop"** menu, select **"Preferences"**.



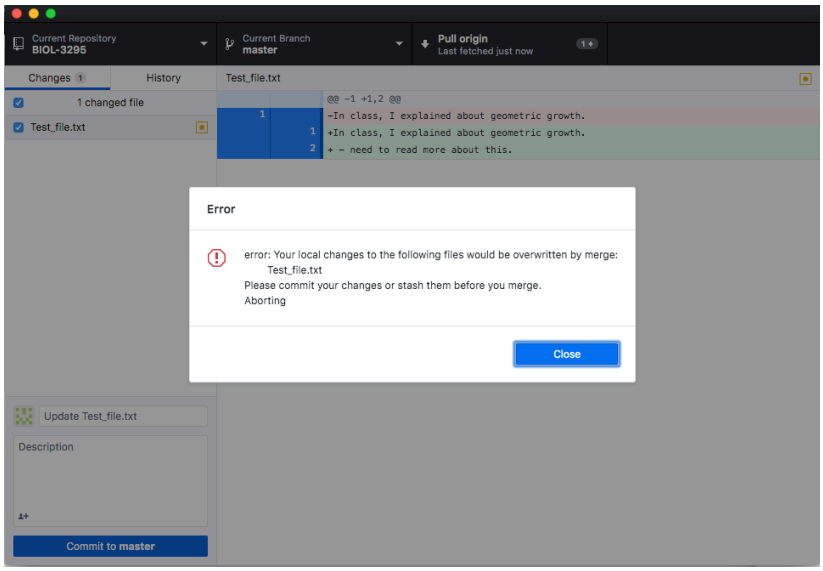
10. Next, you want to sign in to your github.com account using your newly created credentials:



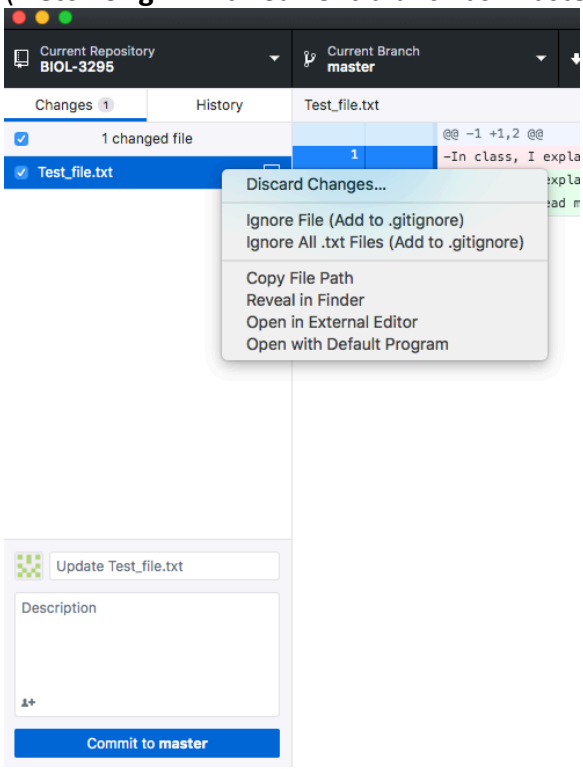
I'm not certain that at this stage of the class this is necessary, but for Labs 5 and 8 you will need to submit your own files to the class repository, and you will need to be logged in for this.

2. When you make changes to your local files (which causes conflicts)

You might add notes to your local copy of the lecture slides, or you might edit an R script while working on a lab. This may cause conflicts as the version of the files on the remote server doesn't match your local version.



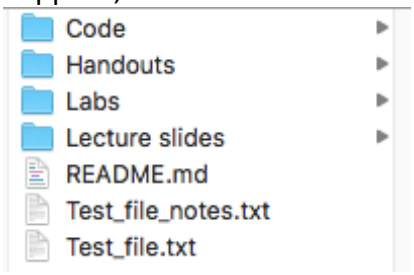
On the left hand side we see that there are conflicts on the file 'Test_file.txt'. We can right click (control click) on the yellow square with the circle inside and then select **'Discard changes…'**. This will discard all you local changes, removing the conflict, and you can now sync to the latest class material (**'Fetch origin'** with Current branch as 'master').



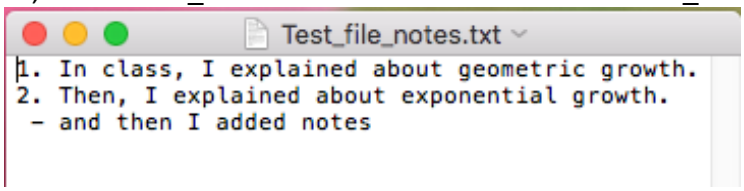
But clearly this isn't desirable since you have to delete the work you had done locally to resolve the conflict.

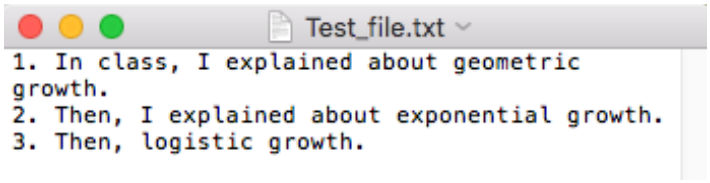
2a. Avoiding conflicts by renaming files

Suppose, we instead rename the local file with our edits. This is one option that appears to work.



Here, 'Test_file_notes.txt' is my local copy of 'Test_file.txt', which I have added some personal notes to, while 'Test_file.txt' is the remote version of 'Test_file.txt', which the instructor may modify.



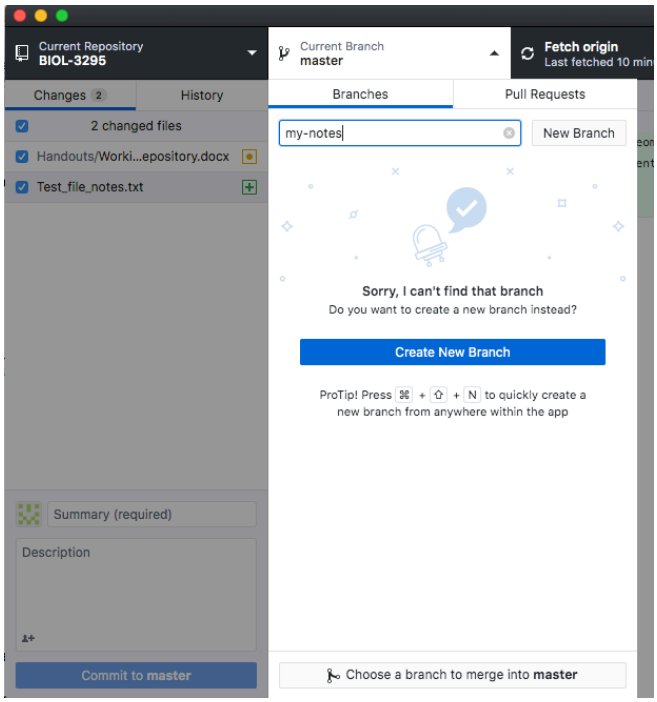


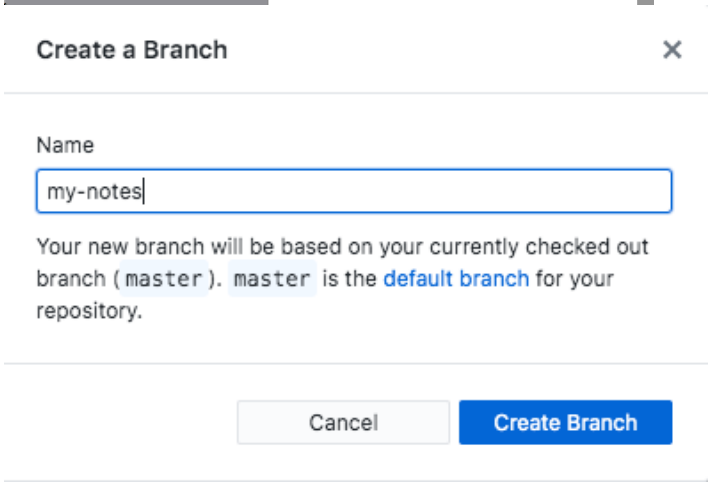
And so we can see that the remote (and local) version (Test_file.txt) has had “3. Then, logistic growth.” added since you wrote your note “- and then I added notes”, which only appears on the file ‘Test_file_notes.txt’

2b. Avoiding conflicts by creating a New Branch

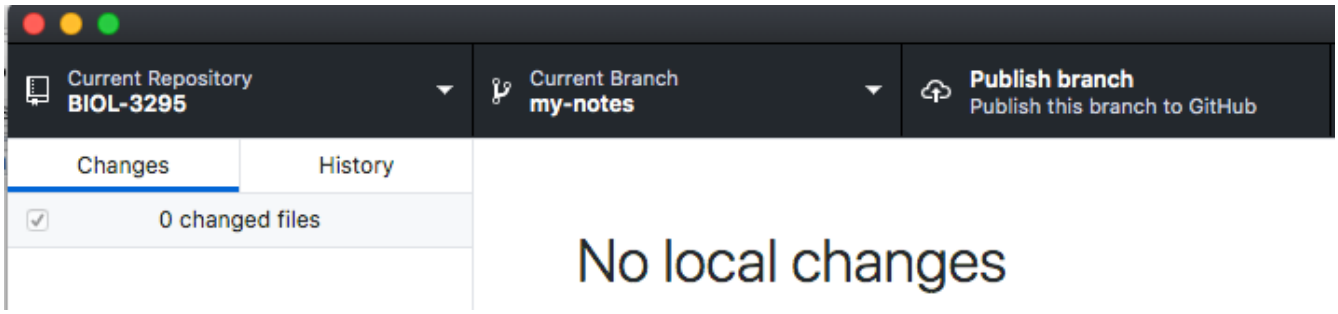
Instead of renaming files you can fork the repository by **creating a branch**.

Click on the Branch tab:

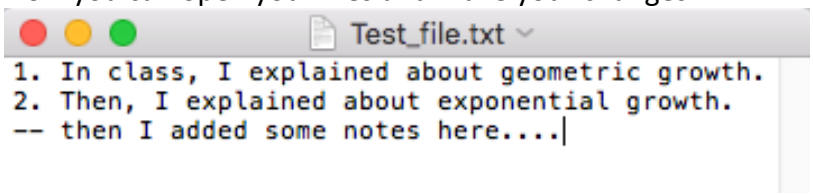




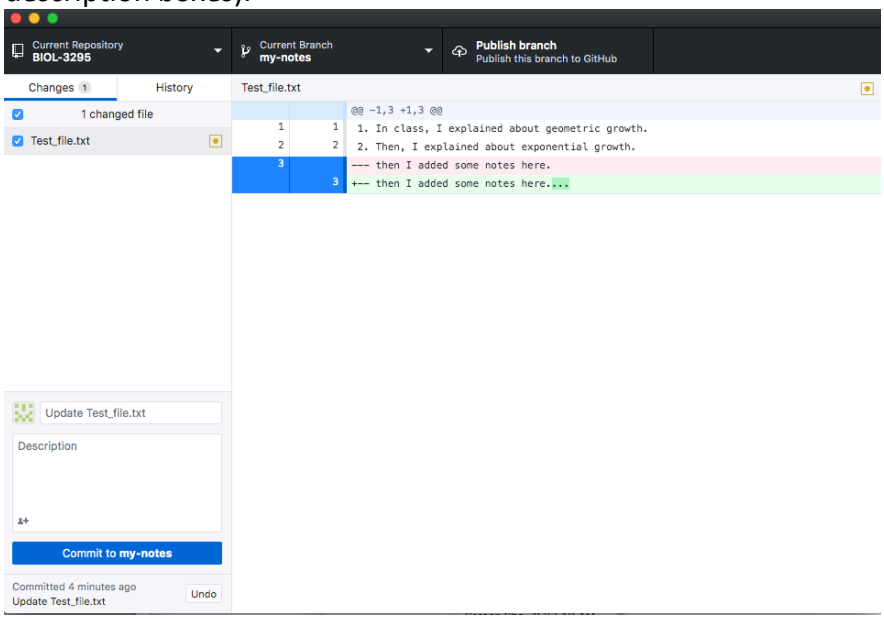
After creating your branch, make sure that your Current Branch is ‘my-notes’ when you start editing your local files. See the centre tab below:



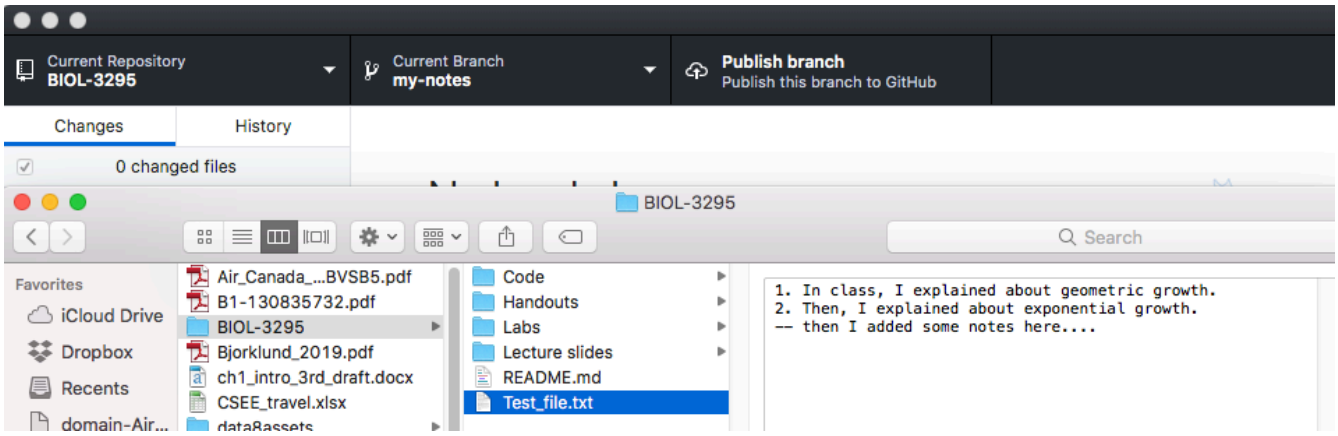
Now you can open your files and make your changes:

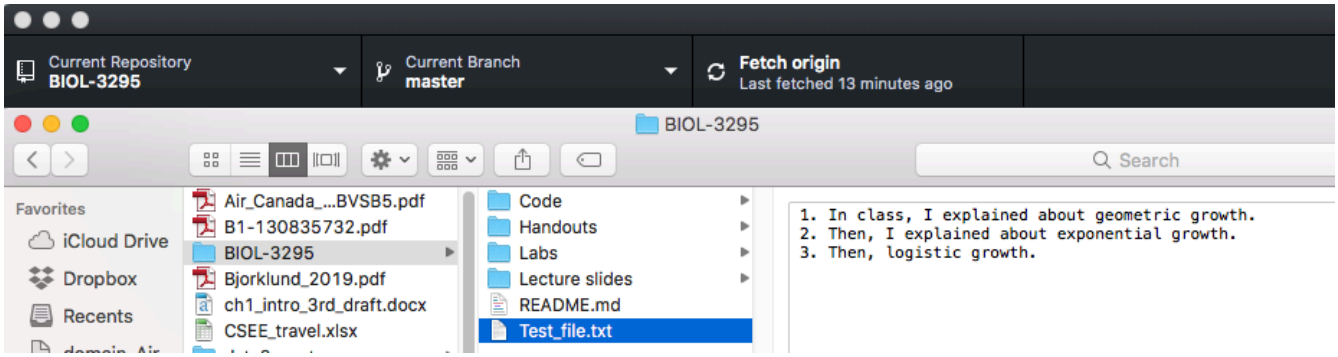


To save these changes to your branch, you want to commit them, by clicking the blue **"Commit to my-notes"** button in Github Desktop. (If this isn't available to click, you may need to fill in the title and description boxes).



Now you have 'forked' the repository - you have two local versions (or branches): 'master', which is an exact copy of the remote version (i.e. the instructor provided version) and 'my-notes' which is your personal copy with your notes added. To view either use the middle tab "Current Branch" in GitDesktop.





See how in the two photos above the version of Test_file.txt is different depending on the branch I have open in the middle tab.

To summarize you have two ways to add notes to your course files: 1) rename the file with the notes, or 2) create a branch and commit the changes.

But perhaps you are still wondering “but why don’t I just add my notes to the master branch?” a) this will create conflicts where, when you try to upload new course material, there will be confusion about which version should take presidence (your local copy with your notes, or the newer remote version) and you will have to resolve this conflict before proceeding.

Or perhaps you are wondering “why don’t I just commit by changes to master after I add my notes?” I suspect you, as a student in the course, don’t have this option because I am the administrator of “master”. Instead, you will be forced to create a branch with your changes, and then you can make a “pull request” to have your changes be considered for “master”. As the administrator, I would be notified of the “pull request” and would have to approve your changes before they could be added to master. Therefore, “master” is the version that is everyone’s copy, and so it doesn’t make sense for you personally to change the master branch, you only want to change your own copy, so you make a branch and commit your changes on it.