Notes on Using Git in this Class

Stephen Taylor, January 2022

If you are completely new to git, GitHub, and source control please watch this video: <https://www.youtube.com/watch?v=wpISo9TNjfU>

The files that exist on GitHub are in a “repo,” a repository of files. When you *clone* a repo to your machine, you’ve created a copy of the repo, what we call a local or working copy repo. Before GitHub existed, a large distributed group of folks could work on a huge project with Git repos around the world. If one of us makes a change, we only need to send the changes (the “delta”) to each other, and then our repos would all be the same again.

With GitHub, this distributed flow isn’t used as much, since instead of a web of repos all over, now there’s more of a hub-and-spoke: GitHub keeps the “main” repo, while each of us pushes and pulls changes down to our local repo.

## Folder “Bookkeeping”

With notebooks there are two folders we need to keep straight. The first is our Notebook folder, that’s the folder where Notebooks run. When Anaconda is installed, the default Notebook folder is usually your $HOME folder[[1]](#footnote-1) (e.g., in my case it might be /home/stta9820 or C:\Users\stta9820). Then I create a Notebook folder (entirely my made-up name, I could have called it “$HOME/cheesy-poof”. So long as it’s in the $HOME folder, it’ll work) to run my Notebooks.

A second folder is your Git repo (this is where you cloned from GitHub). I recommend keeping the two folders separate. A common mistake is to overwrite your repo (e.g., fill in a notebook, ready for turning in), but then need the original back. So if you’re not proficient at git commands, this can be a mess.

Better is to clone to a separate folder, then copy files over from your local repo into your notebook home.

I recommend flow from figure 1 to keep everything straight:

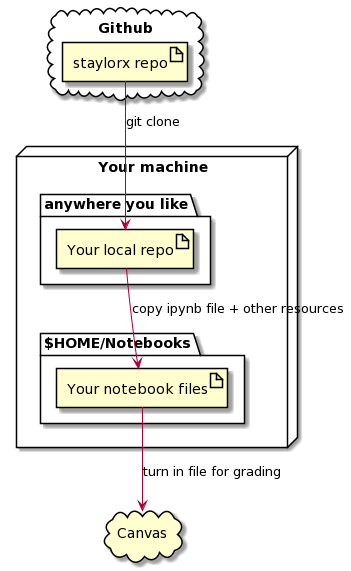


Figure – Where we download an entire repo to a new folder

## GIT CLONE

To clone the course code repository (“repo”) to your notebook folder, use your git client -- Windows folks, I like TortoiseGit but GitHub Desktop works pretty well I’m told — to clone the repo. If you clone the repo to a folder you don’t want, you can always simply move it to where you want it to go. All the guts of git live in a folder called .git (that’s got a dot in front of it). If you can’t see it, that’s fine (maybe better!). Point is don’t muck about in that folder. If you remove it your git repo turns into a plain ordinary folder and you’ll need to clone again.

From your browser, go to https://github.com/staylorx/cu-leeds-baim3220, click the green button that has “Code” on it, copy the https link (here: https://github.com/staylorx/cu-leeds-baim3220[[2]](#footnote-2)) and using your git client, clone that repo to any folder you like. Because Jupyter has a default notebook folder it runs it you just need to ensure the cu-leeds-baim3220 folder is in the right place. If it’s not? Just move it. Simple as that.

## GIT PULL

Every now and again I might make a change to the files on the GitHub-hosted repo. In that case, you’ll need to pull from your git client. That should pull down changes to the files. Caveat is that you’d probably be wise to make copies of the notebooks you’re working on. If you change a file, and I make one also, *and* I post that change up to the repo, you may not be able to pull changes down, especially if we’ve changed the same section of code. We call this a conflict, and those need to be resolved manually. It’s not impossible, but the easy fix is for you to copy your notebook to something else, then revert the changes on the original. Revert works great (okay, so that’s a third git command to maybe now), but it’s not so much dangerous as heartbreaking if you’ve built up a ton of work, then reverted it back to what came from GitHub. Make a copy of everything you’re going to work on and keep the original file as a reference (see the two-folder solution as described at the top of the doc).

An example of using “git pull” would look like figure 2 below. We can only a *pull* changes from a repo that we’ve already cloned. And if I’ve made no changes to the main (or “upstream” repo), there will be no changes. So unless you know I’ve changed a file at all, you rarely need to do this.

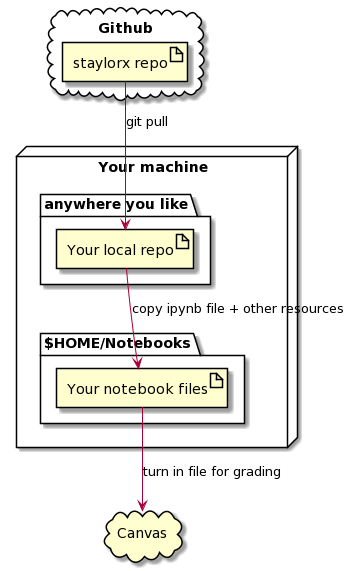


Figure – Where we pull changes to an existing local repo

1. The default notebook run folder can absolutely be changed, but it’s outside of the scope for us. If you know how to move the default run folder, and you’re comfortable doing so and keeping the folders straight, then this document probably isn’t for you anyway. [↑](#footnote-ref-1)
2. I’m using https in this example though I almost always use SSH professionally. If you agree, and know what I’m talking about, you almost certainly don’t need this document at all. [↑](#footnote-ref-2)