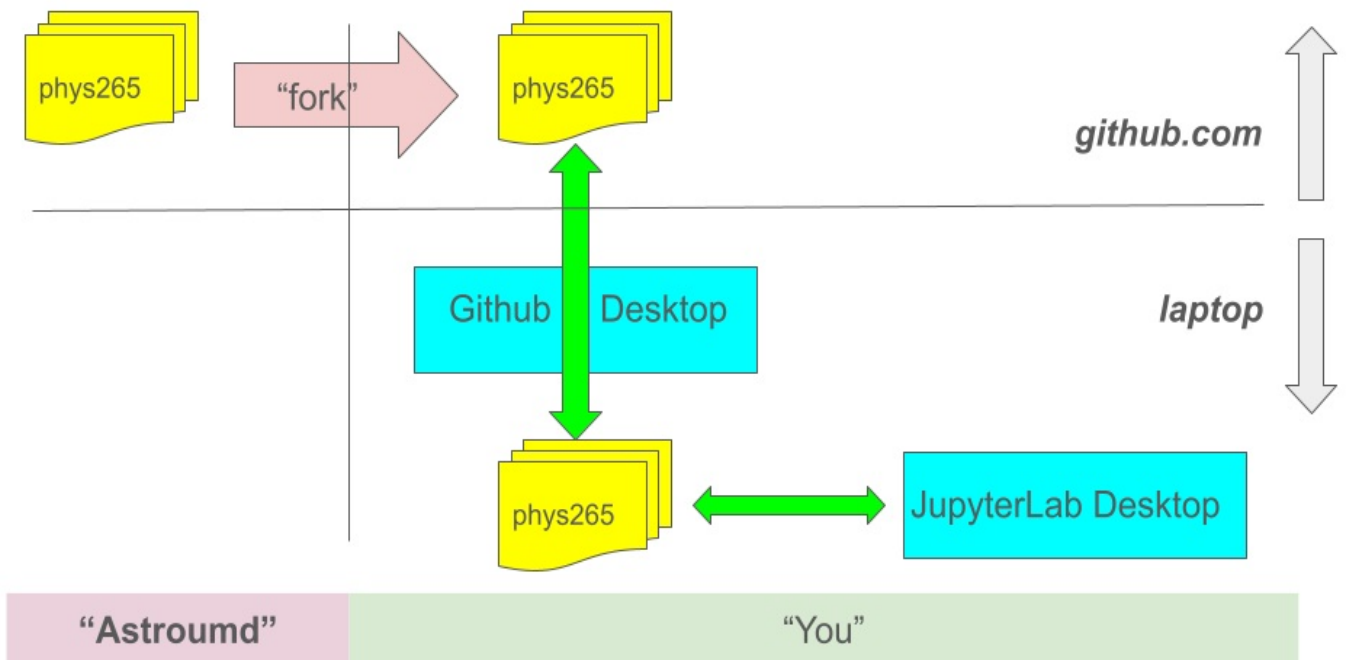


Github, Github Desktop and JupyterLab Desktop



git and github (2026)

This document originates here: <https://github.com/astroumd/PHYS265-spring26/blob/main/github.md>

git is arguably the most common Version Control System (VCS) on the market and very popular with Open Source Software. <https://github.com> is one of the [major contenders](#), although gitlab and bitbucket offer competitive interfaces to maintain your git repository. For private use you can also use git on personal laptop or a home network. Obviously for collaboration something like *github* is the way to go.

In this class you will learn git, and how to use github to maintain a git repository. We will use git to submit the three labs, not ELMS. Remember, after you graduate, there is no more ELMS. But will be github!

Even if these terms makes sense to you, we will **not** use git *branches*, or learn how to submit a *pull request*, or force *main branch protection*. In git terminology, all w/e need is: clone, add, commit, pull and push. Maybe fork.

Installation

GUI: GitHub Desktop

Github has a nice GUI frontend to git called *GitHub Desktop*. For Mac and Windows users you can download and install this from <https://desktop.github.com>. This application greatly simplifies working with Git.

Linux users can get it from <https://github.com/shiftkey/desktop/releases>

Problems can be communally filed under <https://github.com/astroumd/PHYS265-spring26/issues>

CLI: git

With Linux, Mac or Window/WSL (Windows subsystem for Linux) git is already included as a command. Try the command **git --version** in a terminal and you should see something like 2.34.1 - the exact version does not matter for us very much. There is also an official "Github CLI", usually installed as the command **gh**. Nobody in class should need to use the CLI version, though it can be useful for power users with fast fingers.

Steps

1. Create an account on github.com and share your github username with the instructors. This was already part of your Homework 1. You will probably be asked to set up 2FA. I use "Duo Mobile".
2. Fork our existing repo from <https://github.com/astroumd/PHYS265-spring26> into your personal github space. Keep the same name! Look for the **fork** button in the top right portion of the window, and make sure you are logged into your github account. You only need the main branch here.

3. You should now be in your own https://github.com/Your_Github_Name/PHYS265-spring26 repository.
4. Notice the Lab1, Lab2, Lab3 and Project folders in the **Code** tab along the top. You submitted work will be in those folders.
5. Use the Github Desktop and use **File -> Clone Repository** to get a local copy on your laptop. Pick a **Local Path**, ideally somewhere in your personal class folder for PHYS265. Click on **CONTINUE**, as this potentially allows you to merge in updates from the **upstream**.
6. Modify the NAME file, and complete it by putting your name in it. You can do this from your JDL app, or any native operating system tool. Even emacs.
7. Look back in Github Desktop, you should see the NAME file changed. Modified files have an orange square with a circle in the middle. New files you created and will not be in github will show up with the green + symbol to the right of the file in your left pane.

Fill out a small description in the bottom left part. This is important to keep track of why/what you committed this change, for your future self and your collaborators. Go ahead and **commit to main** so it can be ready to push to github.
8. You now will see an option to push this change via a blue button in the right pane. **Push origin** it will say. Go ahead and push this to github now.
9. For new files, for example in your Lab1 folder, the same story, except new files will have a green square with a green + symbol inside. Only add the files that your grader needs to see. All other files will stay private on your laptop.
10. (optional) Create a public README.md file in your own github.com/yourname/yourname repository. For example look at my barebones one <https://github.com/teuben/teuben/> but there are many better examples online.

Authentication (advanced)

As of January 2024 github enforces 2FA (two-factor authentication). Although you will find everything you need to know to work with github on their website, there are numerous web pages summarizing this. One example is on <https://swcarpentry.github.io/git-novice/> which covers the all important [Installing Git](#) and [Creating a GitHub Account](#) to get your started on Linux, Mac, or Windows.

To summarize, there are two methods how to automate your authentication with github:

1. Personal Access Tokens

Settings -> Developer Setting -> Personal access tokens -> Tokens (classic) -> generate new token

<https://github.com/settings/tokens>

Typically you will get a token, something like ghp_b1ab1ab1ab1a that you will then use as a password. When it was generated, you also had to give it a lifetime. Pick one year, or anything you prefer.

2. SSH keys

Settings -> SSH and GPG keys -> new SSH key

See also <https://github.com/settings/keys>

```
ssh-keygen -t ed..... -C your_email@your.doman
```

Copy and paste the contents of your ~/.ssh/ed....pub file into the online field

There could be more here on how keys are generated with **ssh-keygen** and **ssh-copy-id**

Upstream Merge (advanced)

If the upstream (the repo you forked from) was updated, you can merge them into your repo. Of course this assumes you didn't change those original files, that would be a conflict. Here is the recipe to *merge the upstream* as the lingo says:

1. Switch to your default branch (main): from the top menu click **Current Branch** and pick main.
2. Fetch changes from all remotes (including upstream). This is the top right button.
3. Top menu bar: **Branch > Merge into current branch**. Pick upstream/main and create the merge commit.

Class Repository

Our public class repository is available via two different style links: http and git:

```
https://github.com/astroumd/PHYS265-spring26
git@github.com:astroumd/PHYS265-spring26.git
```