GitHub, GitHub Desktop, and You

This guide is Beginner Level Zero, i.e. starting from not even having a GitHub Account or any idea what it is and ending with having the app downloaded and a base-level knowledge of how it works and how to clone a repository (repo) from the GitHub website. There are tons of references out there, this is just one of many!

Contents (Ctrl+Click):

1) *What is GitHub?*
2) *Creating a GitHub account*
3) *Downloading GitHub Desktop*
4) *Using Your Shiny New Account*
    a. *Cloning a Repository (Repo) from GitHub to Your PC*
    b. *The GitHub Desktop App*
    c. *Committing and Pushing from Your PC to a Repo*
    d. *The Forked Repo*

Note: <> means to click, e.g. <Follow> would mean to click the Follow button

1) What is GitHub?

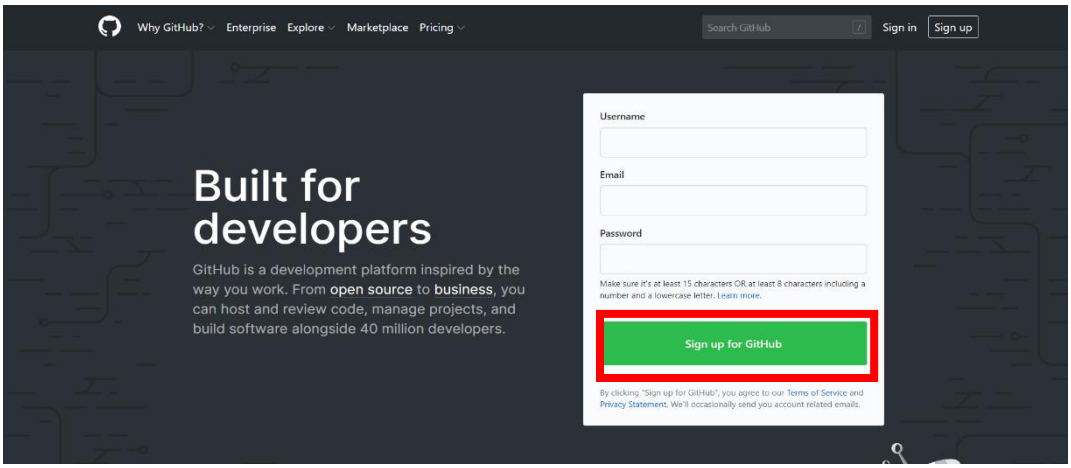
GitHub is the home-base website for the version control system "Git." There are plenty of resources out there explaining version control, why it's necessary, and how to go from no prior knowledge to functioning in the GitHub world a lot more eloquently than I can explain, and these are all laid out in the guide "R/RStudio & Git/GitHub Resources" in the "GitHub Resources" folder. But, since you're already here and eager to learn, I'll list a couple of my favorites:

- *Git for Humans:* A quick (~5-10 minute), accessible, and entertaining slideshow covering why Git is useful and how it can be used, focusing on basic tasks such as the ones our lab will be using. With great graphics and essential vocabulary, this introductory slideshow is a must-view.
- *Happy Git and GitHub for the useR*: ***The best*** resource to go from nothing to a completely linked system in ~45 minutes. A wonderful long-form resource. Great for **troubleshooting.**
- *Git and GitHub*: A thorough introduction to using R and Git/GitHub. Very R focused and shell-focused (don't fret if you have never heard "shell" in this context). Useful to **reference** certain commands.

2) Creating a GitHub Account

A step-by-step guide to creating your very own GitHub account:

1) Go to github.com
2) Enter your email address (.edu is preferable!), username, and password and press the green “sign up” button



3) Verify your account by completing the “Anti-Robo” puzzle.
4) Choose the “Free” Plan (Dr. Assal has the full plan for a lab subscription, we all can use the benefits through this).
5) Choose the appropriate personal selections (what kind of work you do, how much programming experience you have, etc.), press the blue Complete Setup button.
6) Log in with your new username and password! Congrats, you’ve made a GitHub account!

3) Downloading GitHub Desktop

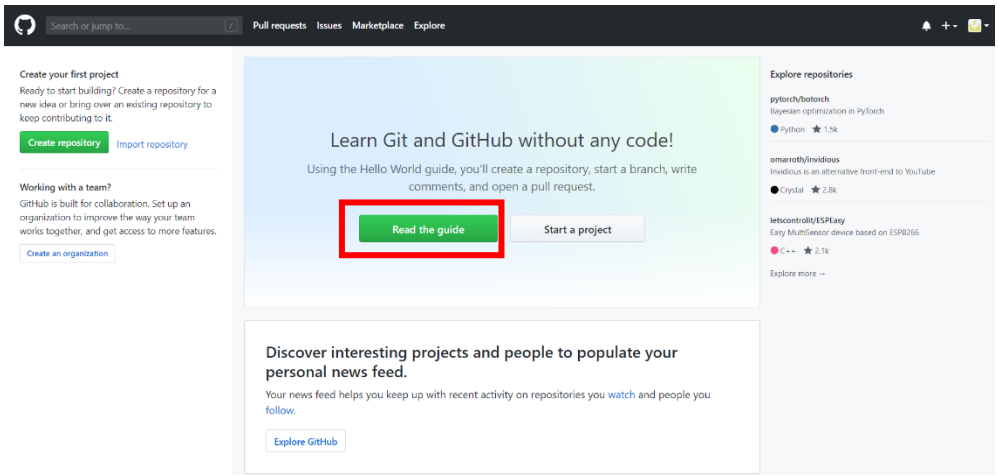
Before we start delving into the wonderful world of GitHub, I’m going to share a pro-tip that will make your life a whole lot easier. Download GitHub Desktop! Steps on how to do so follow:

1) Go to https://desktop.github.com/
2) <Download for Windows> (or whatever Operating System you use)
3) Open the downloaded installer file and follow the indicated steps
4) Congratulations! This should be all you need to do. There is a “Help” tab on the website above for help with troubleshooting.

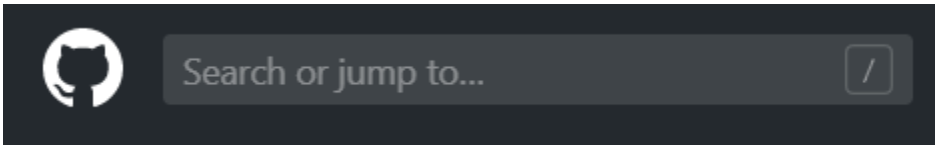
4) Using Your Shiny New Account

4a.) Cloning a Repository (Repo) from GitHub to Your PC

1) Making an account is easy, but getting the hang of using the program is a little more challenging. A link to a helpful guide appears on your home page (red square below); I highly advise you to read this guide. I will be describing how to pull down one of Dr. Assal's lessons, but to *fully understand* I advise you to read the official GitHub Guide (it's a quick read). After that, continue to step 2 where we work through an example of a pull.



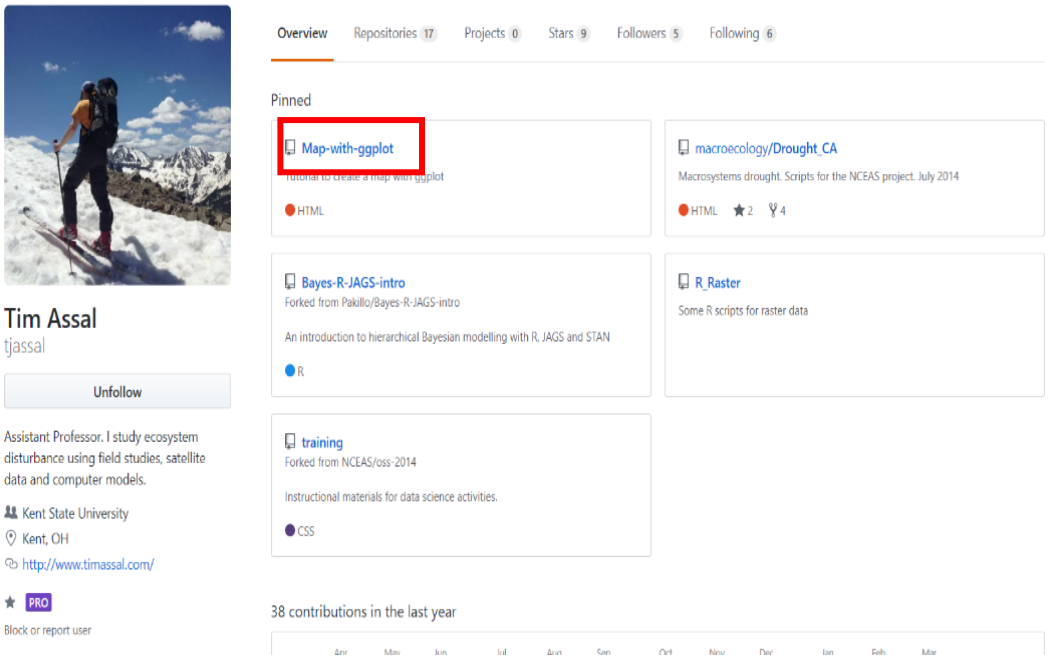
2) In the top left hand corner of the screen, near the GitHub logo, find the search bar.



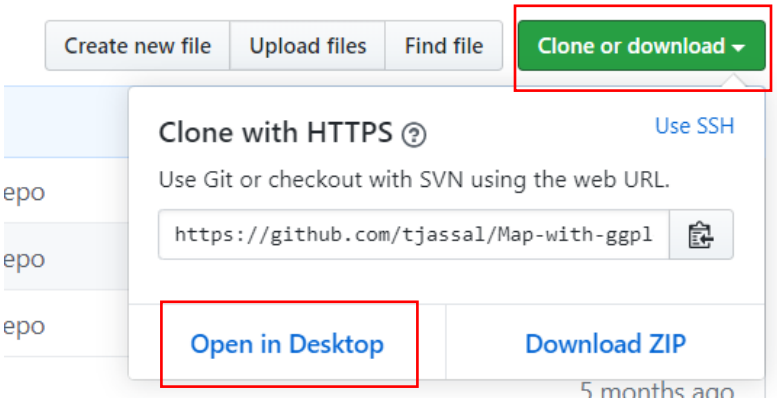
Enter Dr. Assal's username "tjassal", then <Users> then <Tim Assal> then <Follow>.

a. Quick Note: Once you send a request and are accepted, you will see our lab icon under a new "Organizations" Header. It's a cool leaf graphic with a digital green background behind it; it's pretty official.

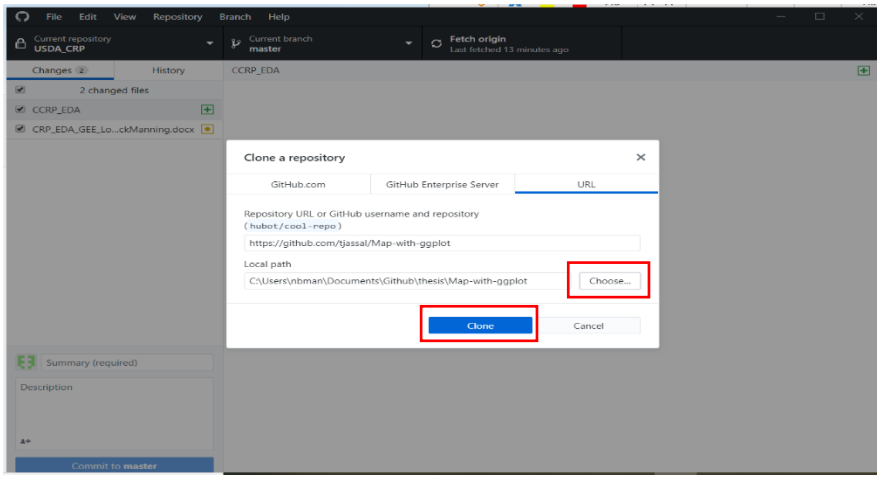
3) From this page, <Map-with-ggplot> in the “Pinned” section



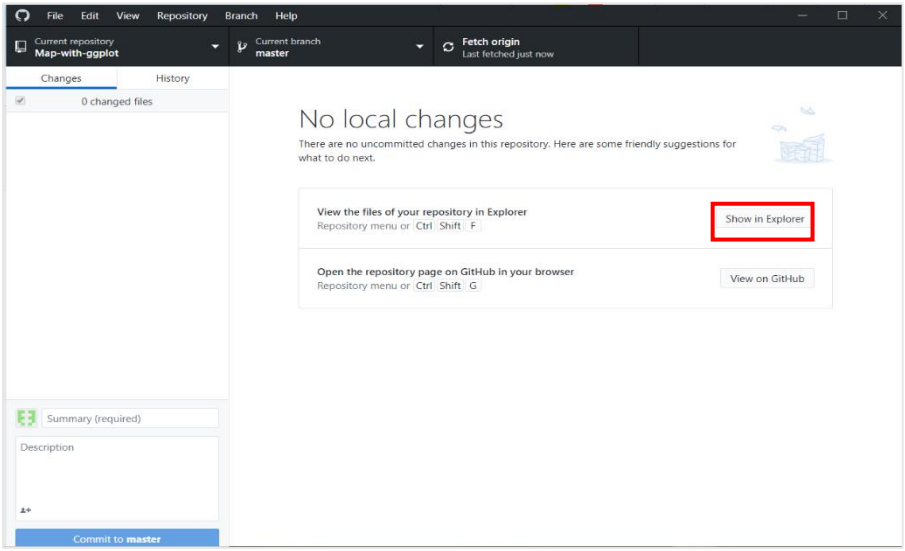
4) <Clone or Download> then <Open in Desktop>



5) This should automatically open the newly downloaded GitHub Desktop app and load the following screen. Choose a location to save the newly cloned repo to (preferably somewhere within the automatically generated “Github” folder (usually in your Documents section)) by <Choose…> then <Clone>



6) Explore the repo you downloaded by opening it in the explorer: <Show in Explorer>



This should give you all of the access to the files downloaded in any specific repo! You can access files, code, anything saved to the repo. Next, we will work on pushing from your personal PC to a specific repo on the GitHub website. The Desktop App is not a necessity to use GitHub, but it definitely makes the process run a lot smoother and enhances user interface accessibility. The next section will take a quick tour of the GitHub Desktop App and highlight some of the features, but there is no substitute for personal exploration. Explore the app on your own to get a more in-depth sense of what it can do, especially the tabs at the top (File, Edit, View, etc.).

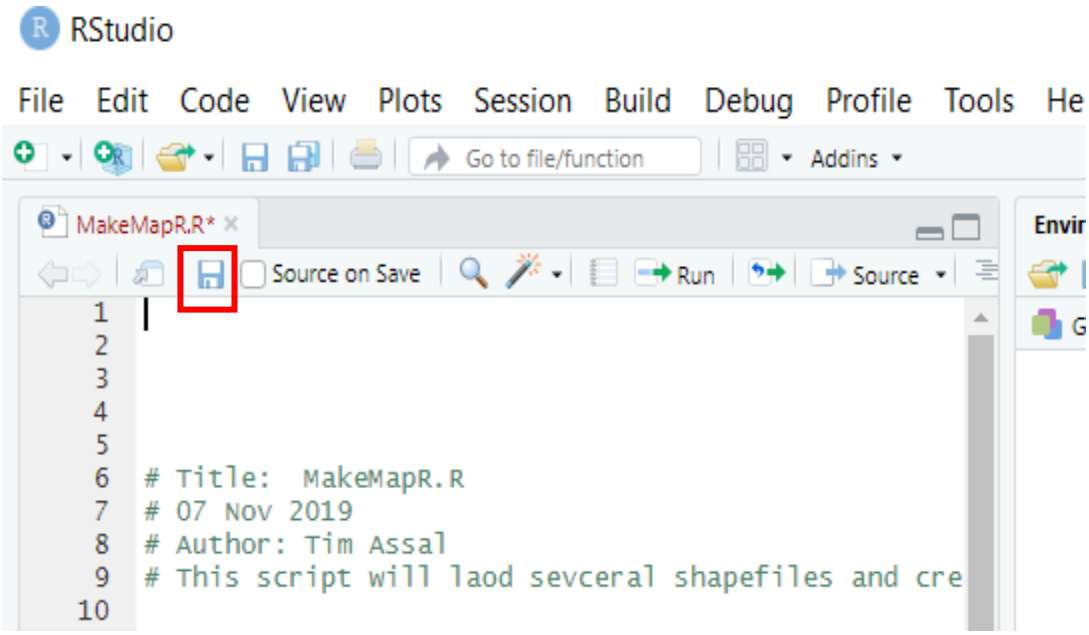
4b.) The GitHub Desktop App

While the app has a wonderful Getting Started Guide (also available by clicking on the **Help** tab in the top toolbar), here are some immediately visible features:

4c.) Committing and Pushing from Your PC to a Repo

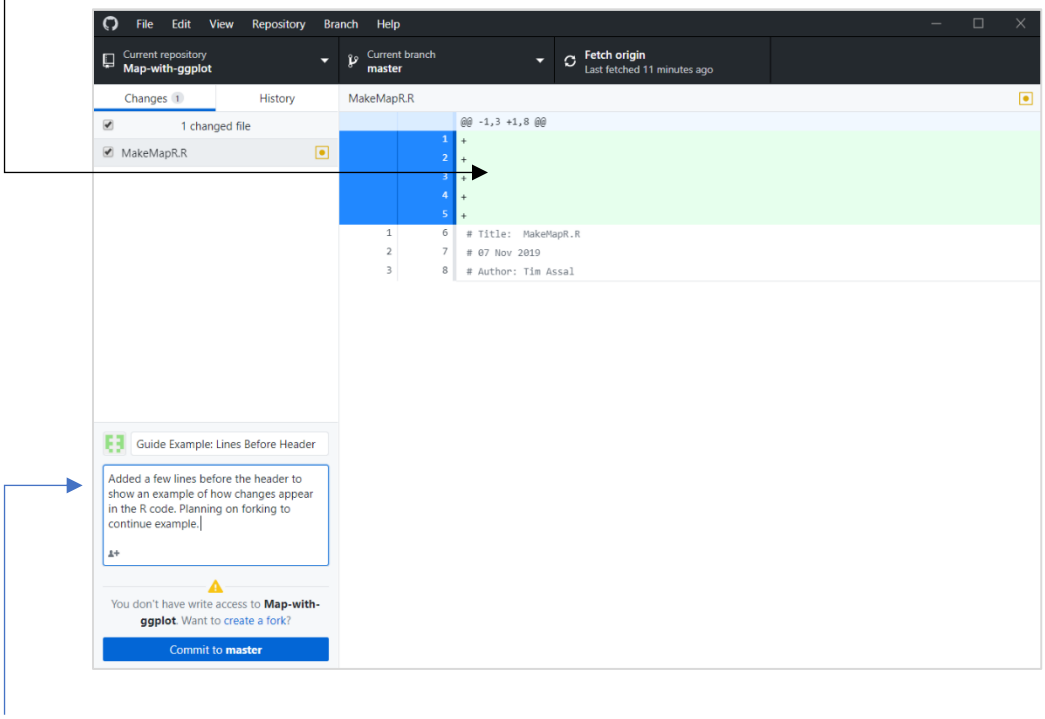
*Now that we've mastered cloning a project from GitHub to your own PC, let's say you wanted to make some changes to a code or collaborate with a colleague. This will involve sending/submitting/saving the work you did in a way that is accessible by your colleague. This involves **committing** then **pushing** your changes. Note: this section will require you to have R (and RStudio) already downloaded.*

1) For convenience's sake, let's continue to use the "Map-with-ggplot" repo. In the File Explorer, double-click the R file titled "MakeMapR", opening it in R/RStudio.
2) It doesn't matter which change we make to this file, but since we don't want to ruin a perfectly good script, simply press enter a few times (five if you want to follow exactly, six if you're independent) at the top of the file to add a few blank lines of code.
3) Click the save icon in the top left corner



4) Open the GitHub Desktop app.

5) The changes you've made should be visible (the green rows added means you've added something, these rows would be red if you were to delete something).

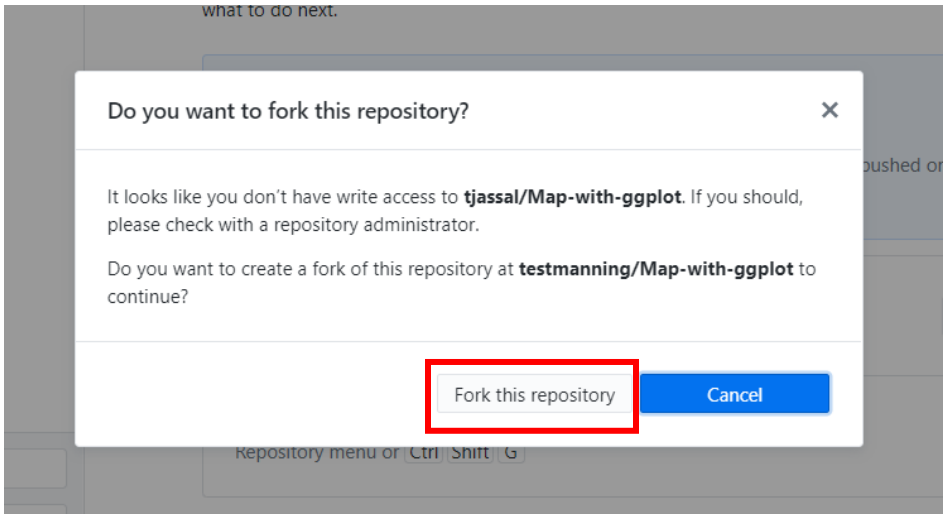


6) This section is where you will **commit** the changes you've made, and make some notes about what you've changed. The top line is for the summary, which usually includes a quick synopsis of what you changed and the overall purpose of the changes. The box below the summary box is the description box, where you will talk more in depth about what you changed, why you did it, and what your future goals are. *Note:* the button with the shadow and the plus sign allows you to add a co-author to the changes you've made in this commit.

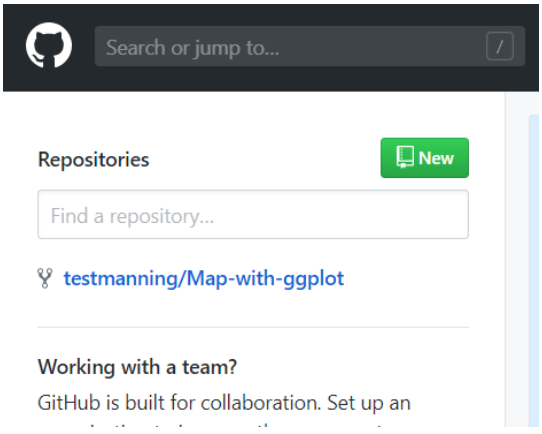
7) <Commit to **master**> *Note:* there can be many commits per one push. Pushing your local changes is more of a final gesture that submits *all* of the local changes in *all* of the edited files. For example, if we changed the code as we did above and changed a Word Doc in the "Map-with-ggplot" repo on separate commits, we could push both of these commits at the same time. Commits are useful because they contain a description of what exactly was changed and for what purpose, as opposed to saving a sequence of files titled "MakeMapR", "MakeMapR_Spaces", "MakeMapR_Final", "MakeMapR_TrueFinal", "MakeMapR_ReallyTrueFinal" (which we're all guilty of from time to time).

8) <Push Origin>

9) A box will appear asking you to **fork** this repository. You should be familiar with forking from the resources at the top of this guide (which I'm confident you did, but take a second to re-visit if you need a refresher, especially the [quick presentation]).
10) <Fork this Repository>



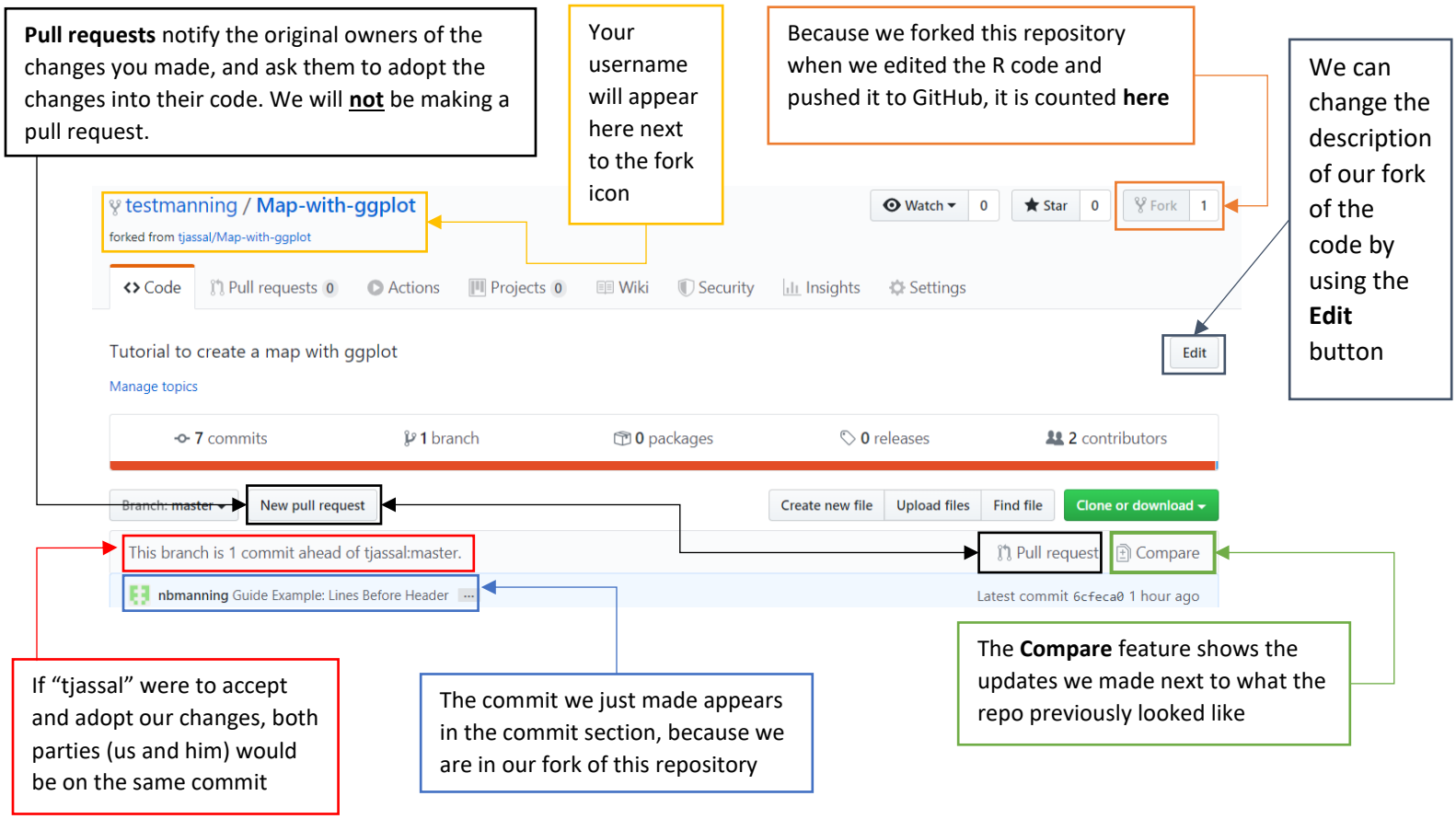
11) <Push Origin>
12) What you have now is a fork from the main repo. If you have "write" access to a repo (or you created a repo, or have any partial ownership), this will not happen, and you will be able to push directly to GitHub.
13) Go to https://github.com/ to access your GitHub online account, where you should find the "Map-with-ggplot" repo under the "Repositories" header



14) The icon to the left of the repo title indicates a **forked repo**. Click on the repo title to take you to the specific forked repo page

4d.) The Forked Repo

Any time you make changes to a repo you do not have at least partial ownership of, you create a **fork.** This section further explains what you can do from your forked repo after it is pushed back to GitHub.



Each time a new commit is made, it will appear here. It will appear as a forked repo if it was forked, or simply as a new commit if you have at least partial ownership of the repo.

You made it! I hope this was a helpful guide to GitHub Desktop. There are many more references out there on the World Wide Web (and in the Help button in the GitHub Desktop app), but I hope this worked well as a quick introduction to GitHub Desktop and cloning a repository. Good luck!