APCS
Lesson 3B
Using GitHub and BlueJ

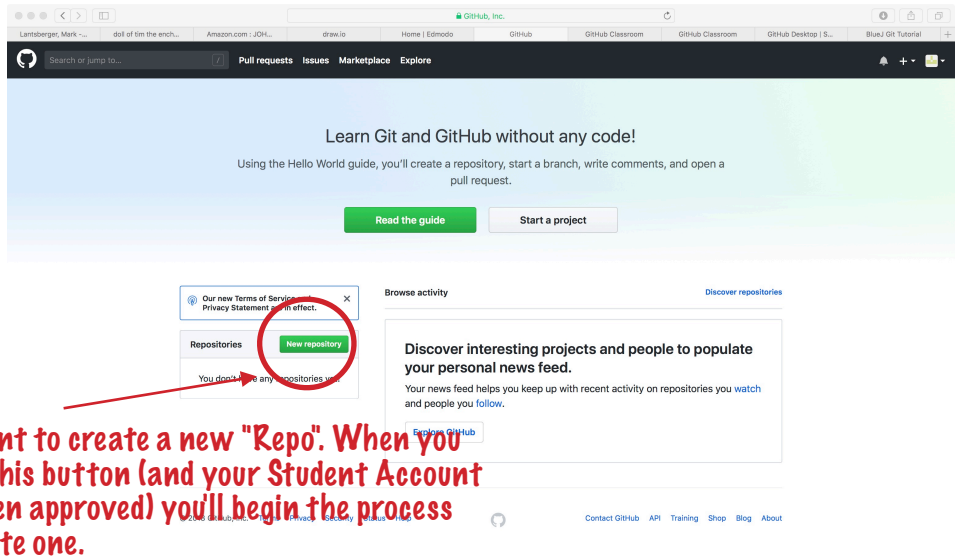
I. Creating a New Project

Creating a GitHub Repository (Repo)

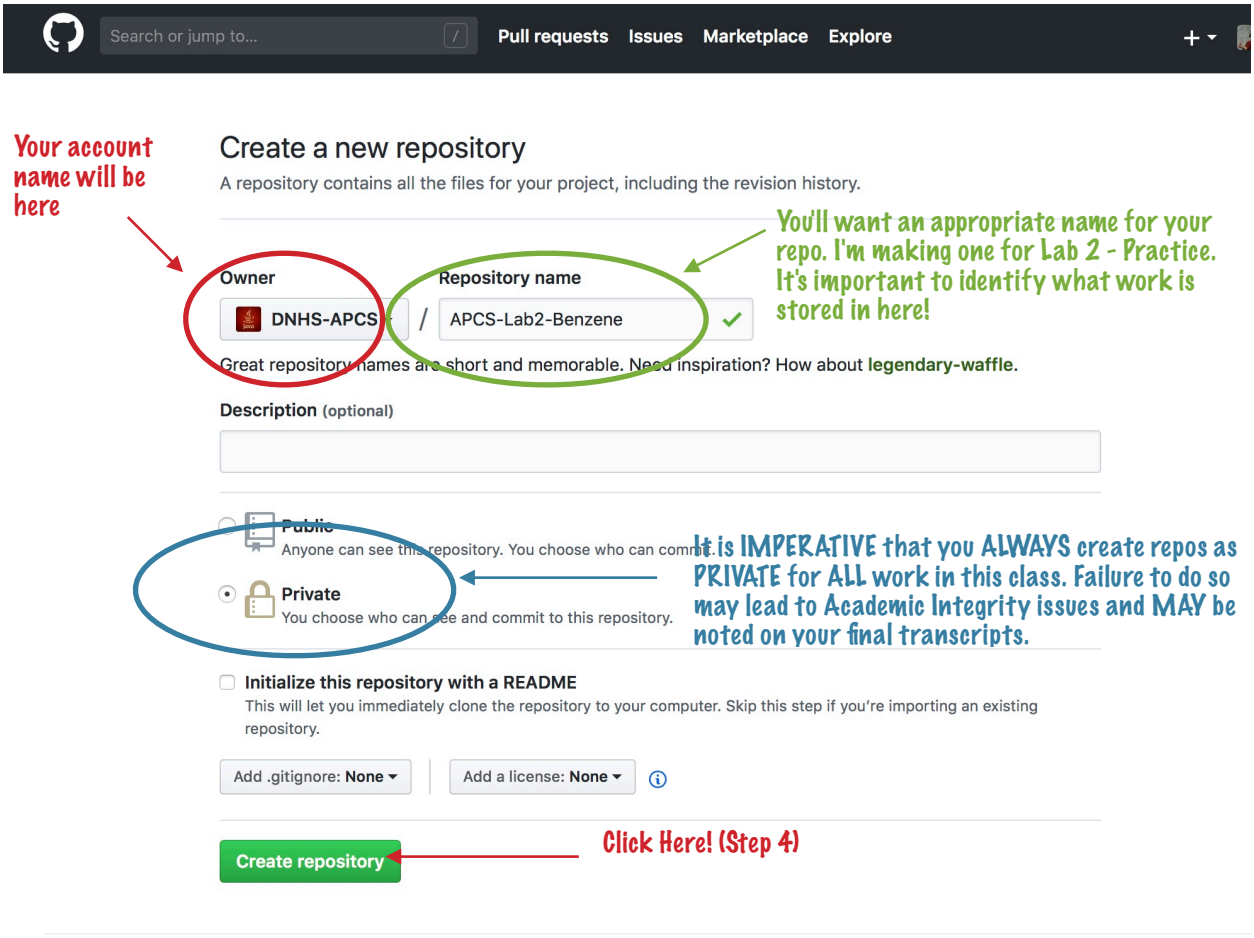
- Git allows you to create a repository (a.k.a. ***repo***) where all of your files can be kept up to date, no matter where your work location (school, home, laptop on the road)!! The repo will look much like the "project folder" created by BlueJ, but it functions a bit differently. We won't go deeply into the details for now (there are SO many details!), but over the next few labs you'll want to practice with GitHub and BlueJ as it will be required by Lab 6. In this guide, you'll find the screen shots are annotated. **The color of the notes indicate the order of the steps. First - red, Second - green, Third - blue**

- GitHub is one of the most popular web-based Git repository hosting services. It provides a web-based graphical interface where you can see your repository, its files, commits, etc. It also provides forums where you can discuss features in your projects, and a wiki engine where you and your collaborators can create documentation for your project. We'll learn more about these possibilities as the year moves on.

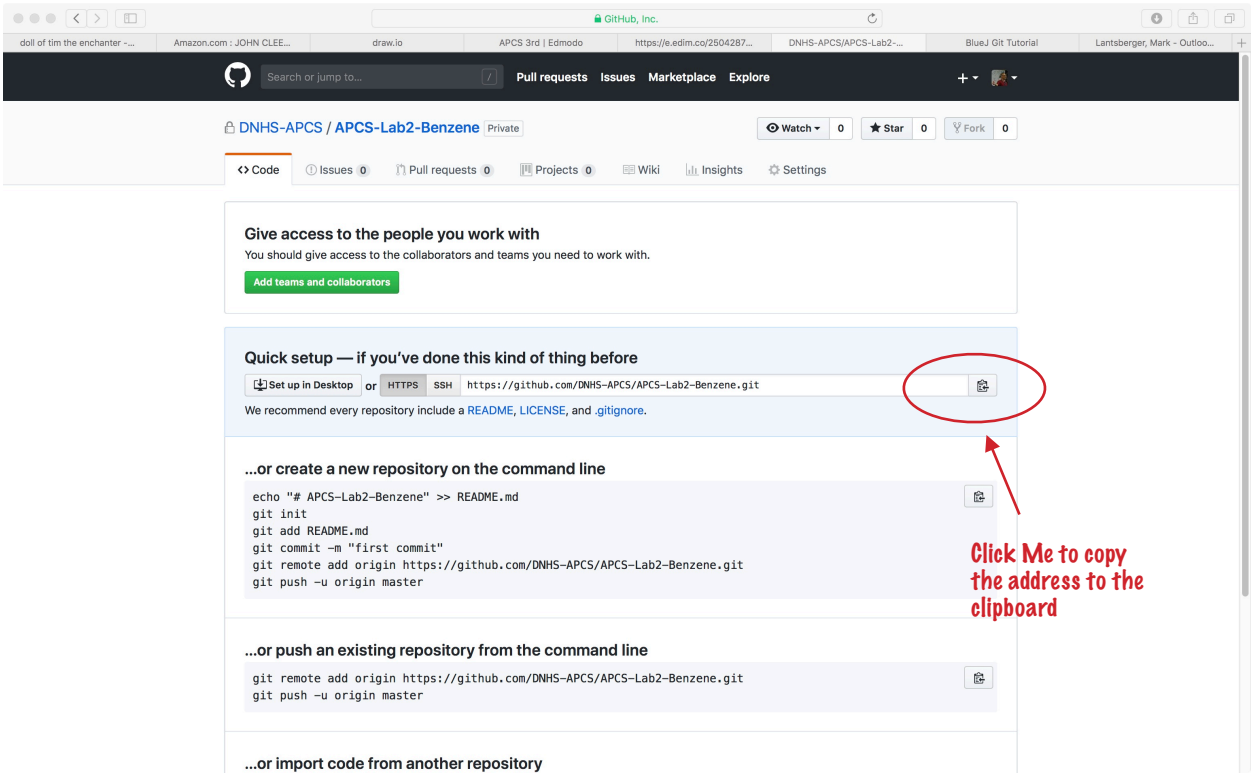
- Log on to github.com and you'll see this page (once you've created your account).



- Once you press the button above, the process to create your repo will begin. At this point, we want the repo to be empty. It's simply a storage location online where we will tell BlueJ to save our work for later retrieval. Here's the screen you'll see:



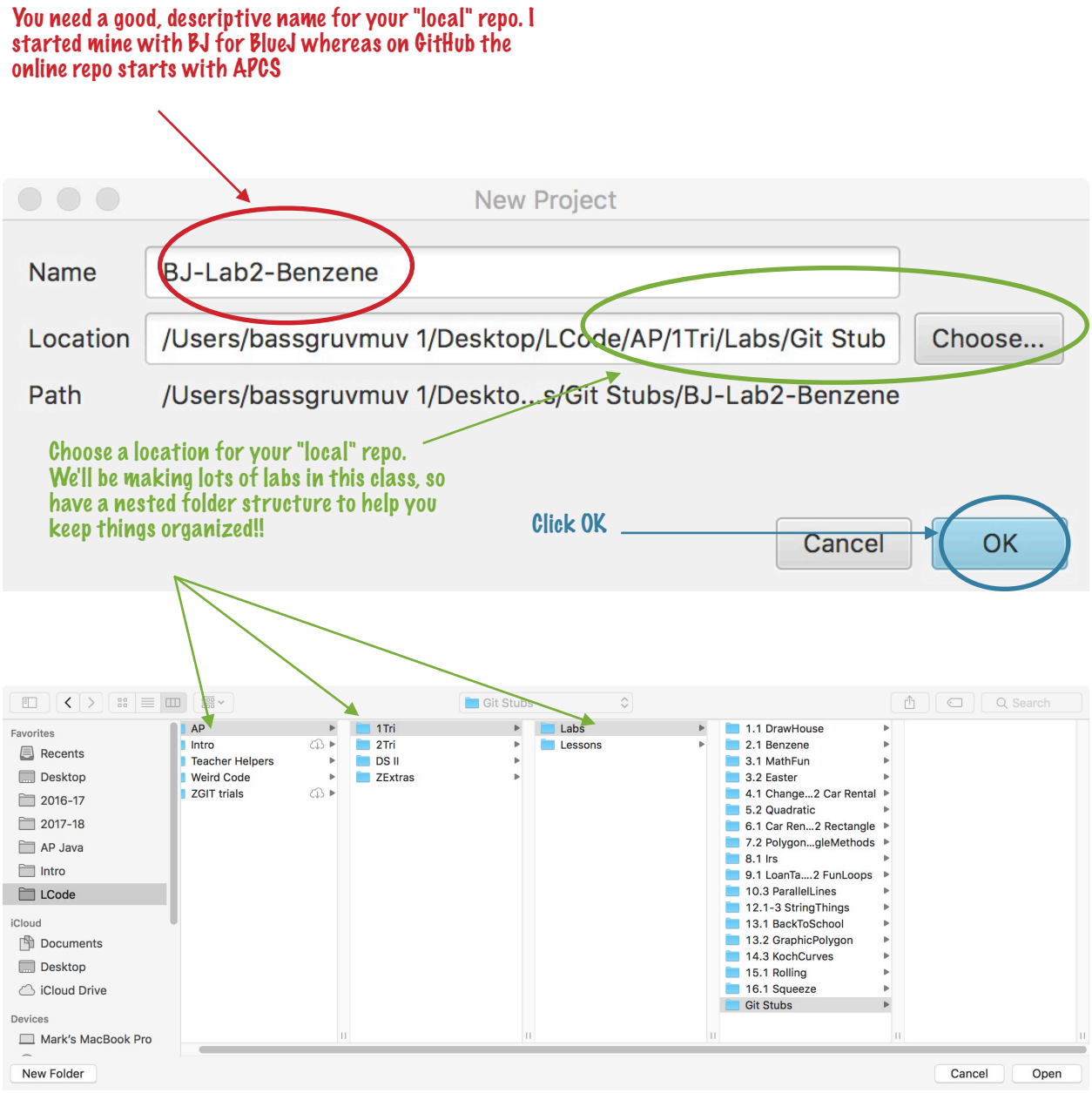
• The next page shows your completed repo. We'll leave this to go work in BlueJ for a while, but the important feature here is the address location of your repo. You should copy this to the "clipboard" for use with BlueJ.



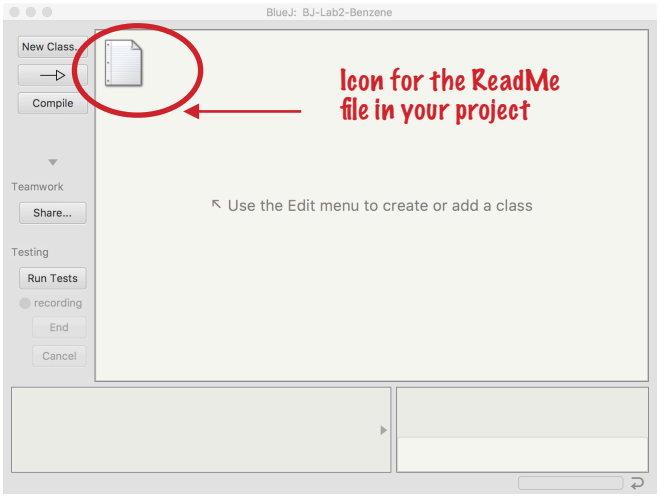
Creating a Repo with BlueJ

- Now, we should launch BlueJ and make a "local" project to begin our work. This is a place where git and the version control process can be a bit confusing, but it allows you to work in multiple locations. Each location (your school machine, your home machine, your laptop) needs to have it's own "local" repository. We use GitHub to store our changes in an online location where any of these "local" repos can update to any changes you've made elsewhere.

- We'll begin by creating a project as you've done before.



- As you've seen before, the previous step creates your project and yields the project window.



- Many students like to copy the lab specifications into the given Readme file. Just double-click on the icon and you can copy and paste into it. It won't take any graphics, but it's still handy to have the assignment and objectives close by.

README.TXT - BJ-Lab2-Benzene

README.TXT ×

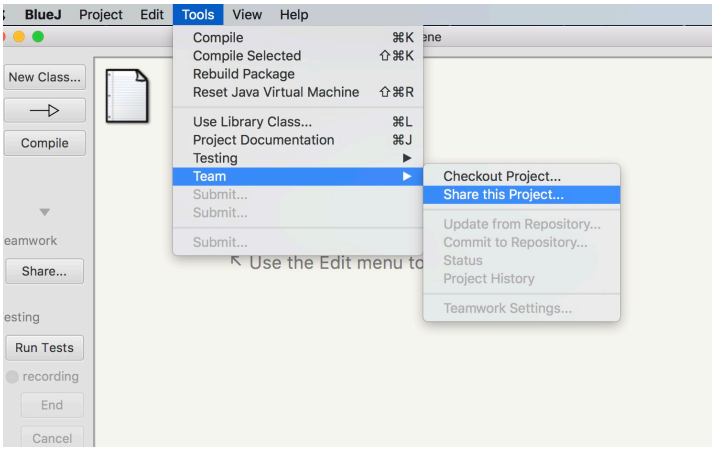
Compile | Undo | Cut | Copy | Paste | Find... | Close

```
Lab Exercise
Benzene
Background:
Benzene is a very common and useful organic compound. Its chemical formula is C6H6 and is often
drawn in abbreviated symbol form as a circle centered inside of a hexagon. Here's an example:

Assignment:
This program will use the apcslib library introduced in Lesson 1. Refer to the DrawingTool Class
Specifications as described in H.A. 1.1. Write a program that creates the benzene ring symbol consisting
of a circle centered inside a hexagon. The circle does not touch the outer hexagon, as indicated in the
example picture.



Instructions:
1. Include your name as a documentation comment and also a brief description of the program.
2. Call instructor to your workstation for scoring.

```

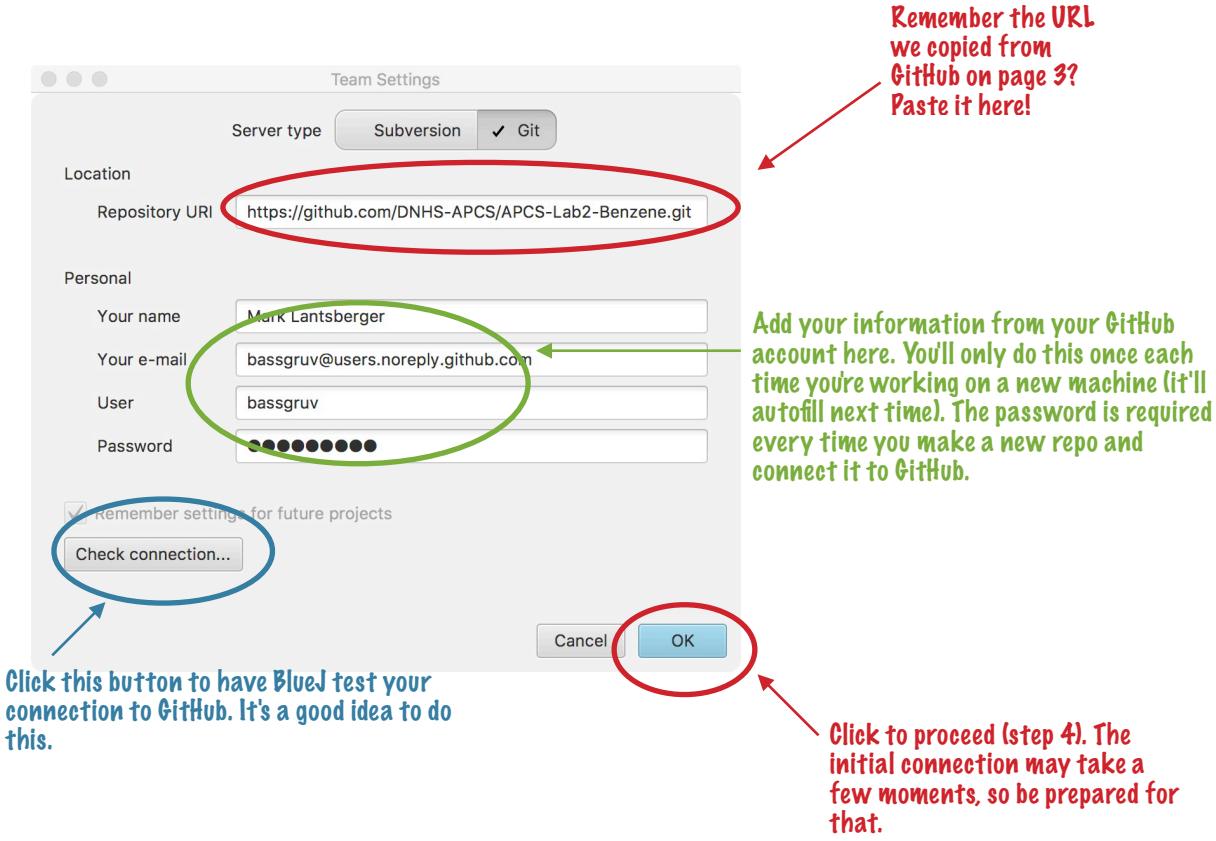
changed

- Now, we've got a nice setup to start our work. You're next step is to create the connection between your "local" repository and GitHub. This allows your code to be accessed anywhere you find yourself working!!

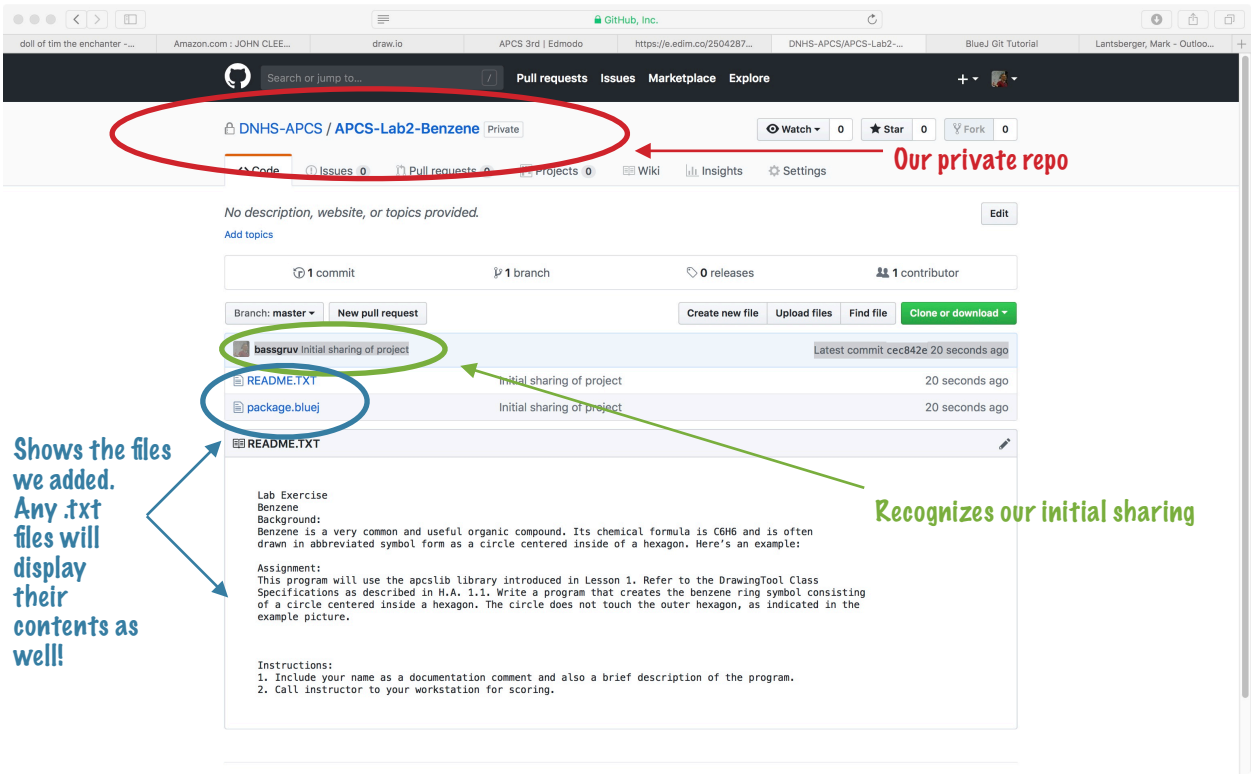
- From the Tools menu, choose Team, followed by Share this Project.



- This action brings up the Team Settings Dialog box



• This initial connection we've done actually performs a git command. A ***push***; which we'll talk about more in a moment. This command actually sent all of the files in our project to GitHub. Here is the result:

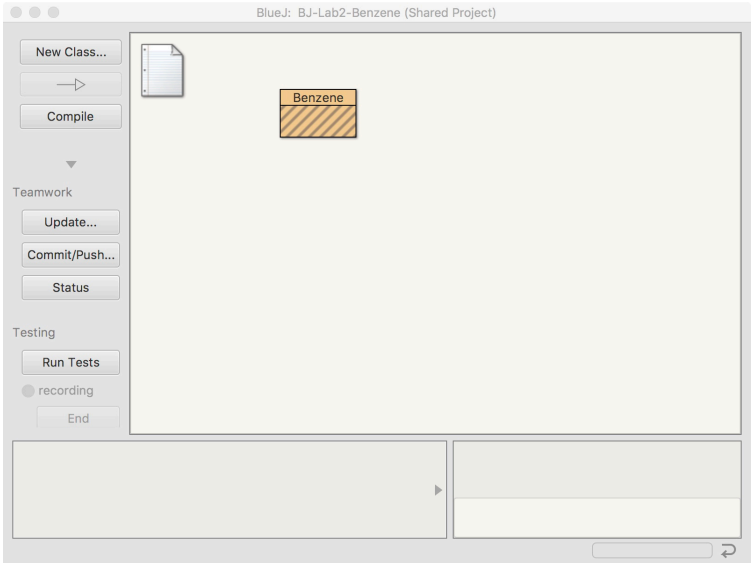


Commit and Push in BlueJ

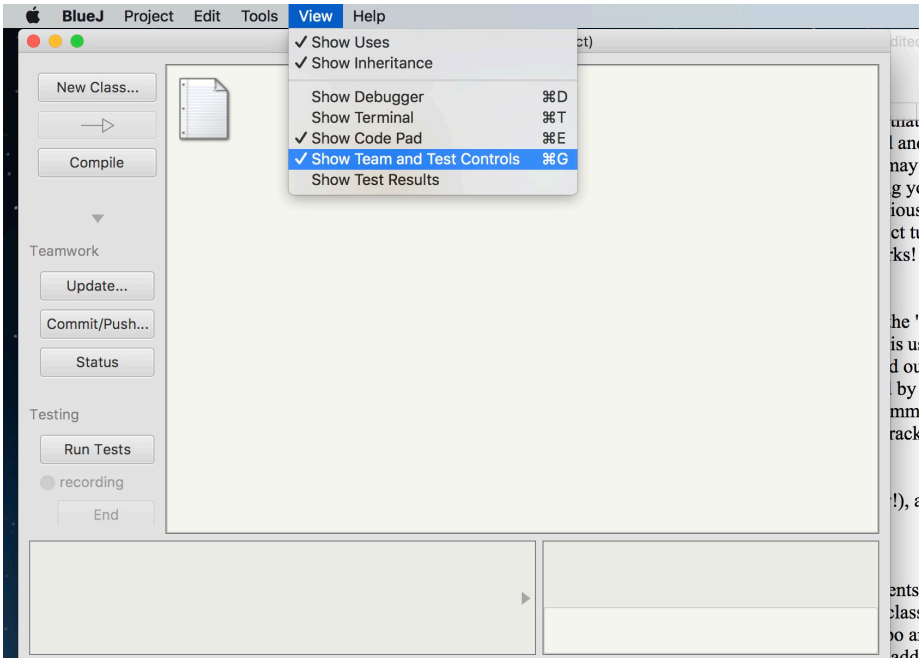
- Usually with computer documents we should "save" the document to the hard drive frequently. This avoids loss of data and grave amounts of frustration. BlueJ saves your files automatically every time you compile (it's that important). We want to use the same idea with git. As you successfully add and test parts of your code you'll need to perform what is called a ***commit***. This may seem redundant, but by having **many** "commits" along the way you're setting yourself up for success. Why? Git allows you to revert your code to a previous commit! So, if you try something new with your code and the whole project turns sour, you can revert your project back to a version you know already works!

- Something to understand is that commits are only recorded on the "local" repository you're working with. Another operation, known as a ***push***, is used to send our changes to our GitHub repository. When we just established our connection a moment ago, BlueJ automatically made a commit followed by a push. With each commit, you're required to include a message about that commit. It's important to be descriptive in these messages if you ever hope to back-track in your project on a future day/time.

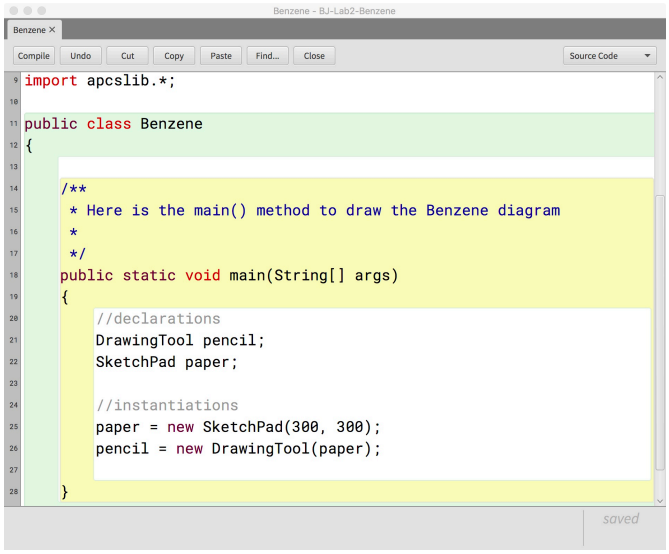
- So, I'll demonstrate by adding a class to my project (finally!), and making a commit:
  - I'll add a class to my project.



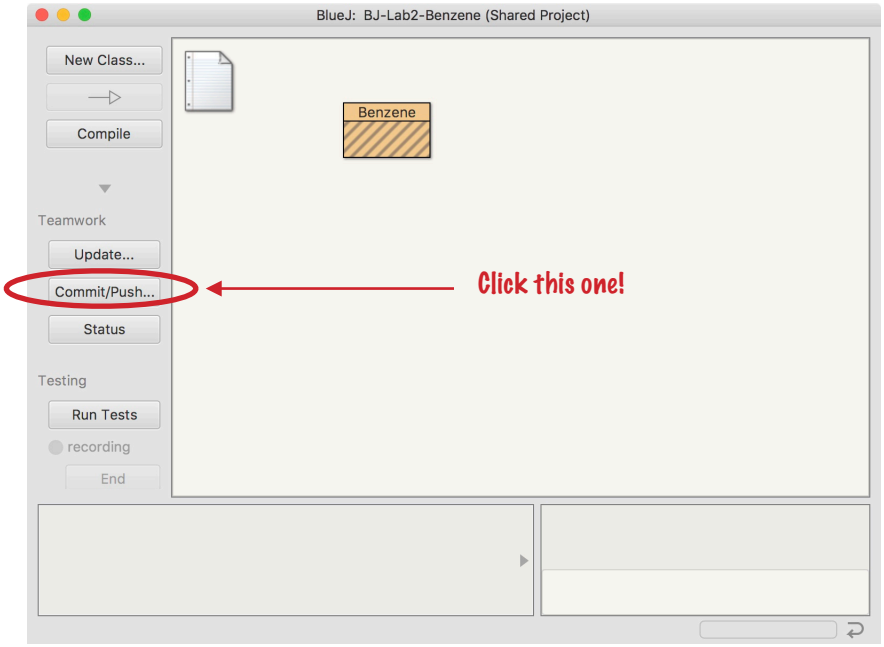
- We need to ensure the Teamwork controls are visible (mine are, but yours may not be). From the View menu, select Show Team and Test Controls.



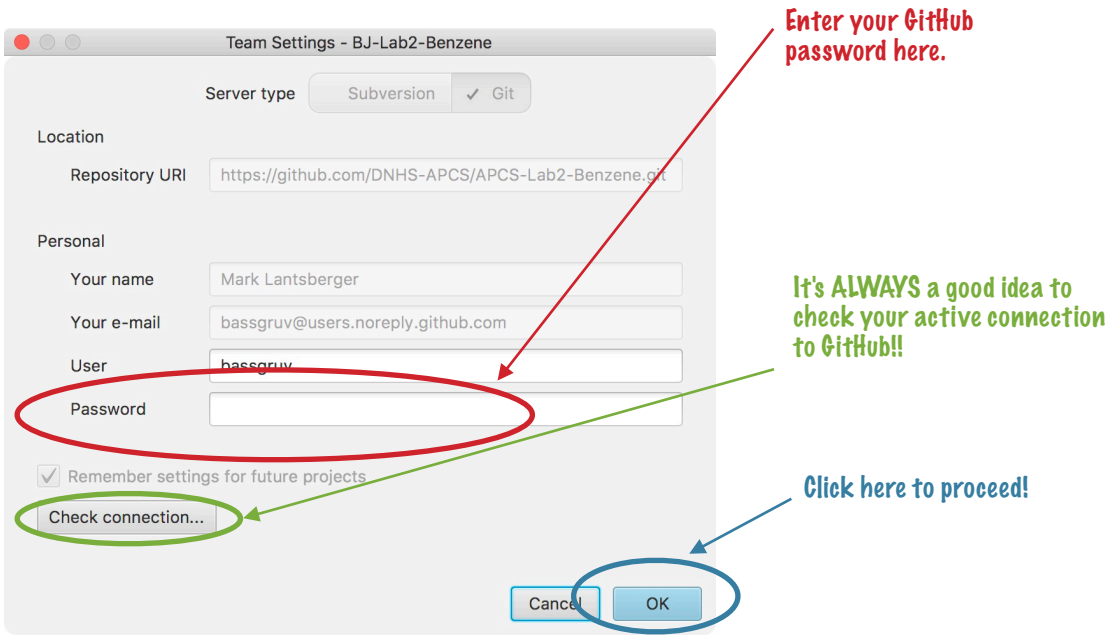
- We'll now add a bit of code.



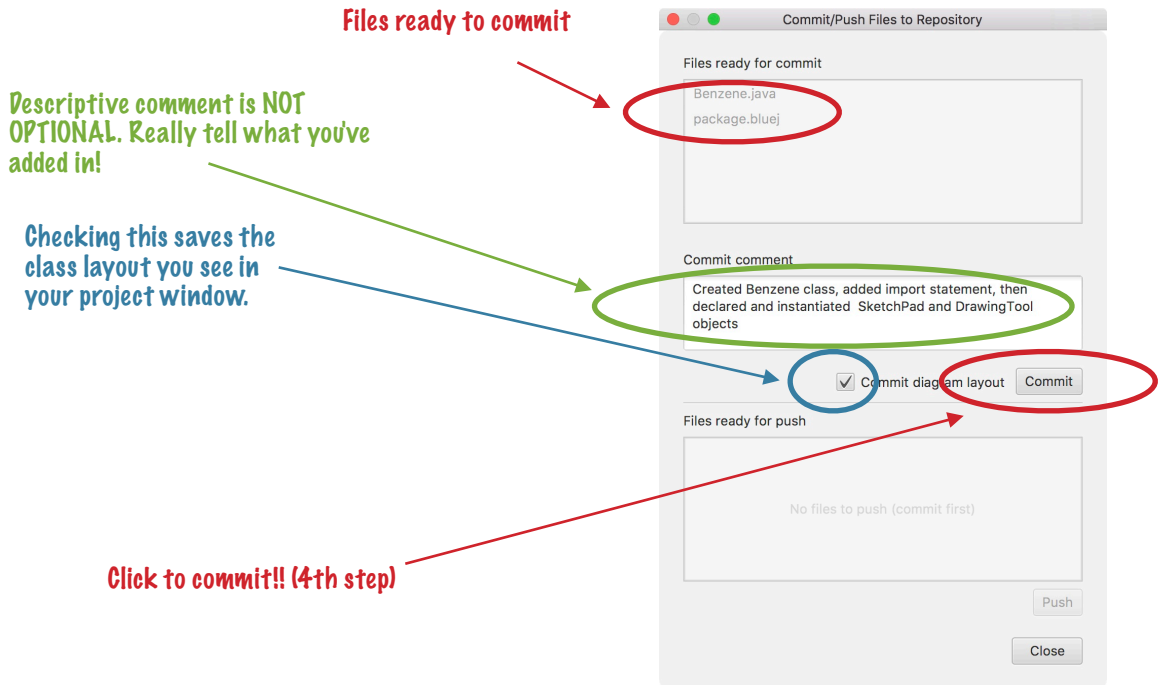
- Next, click the Commit/Push button.



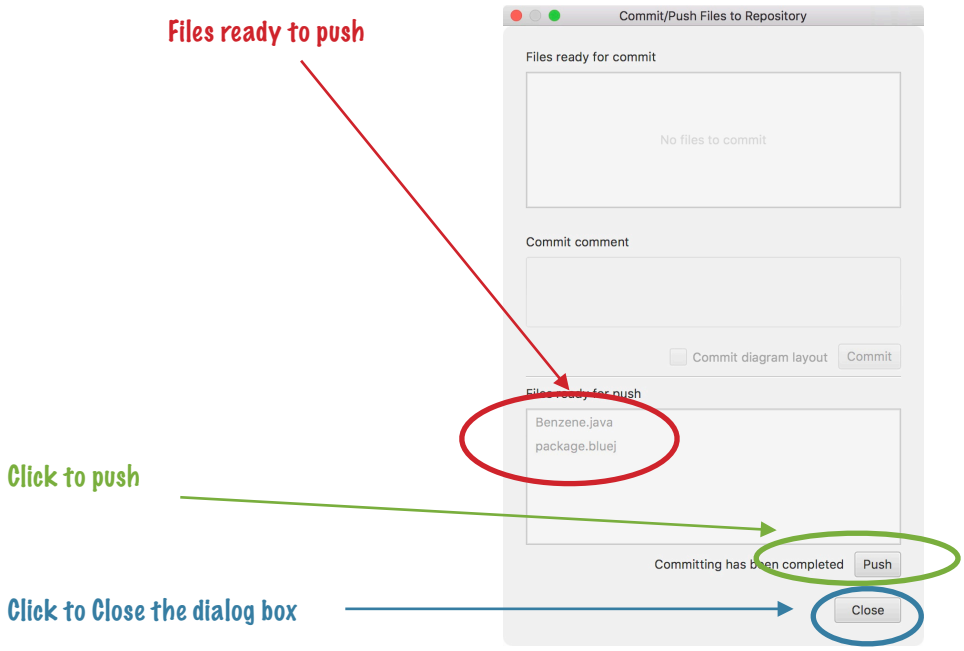
- You may be required to enter your password. This normally occurs when you continue a project at a later time after the previous steps have already been done.



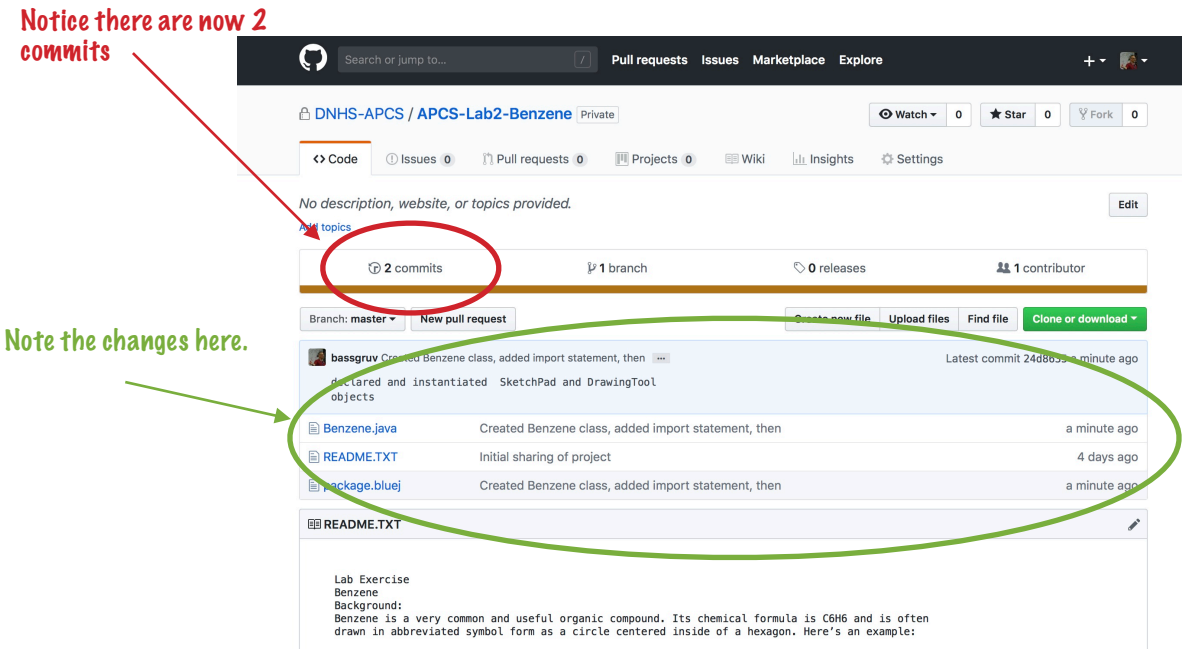
- You must add a descriptive comment before you can commit. BE descriptive, you're only helping yourself!!



- Once the commit is done, you'll see files which are available to push.

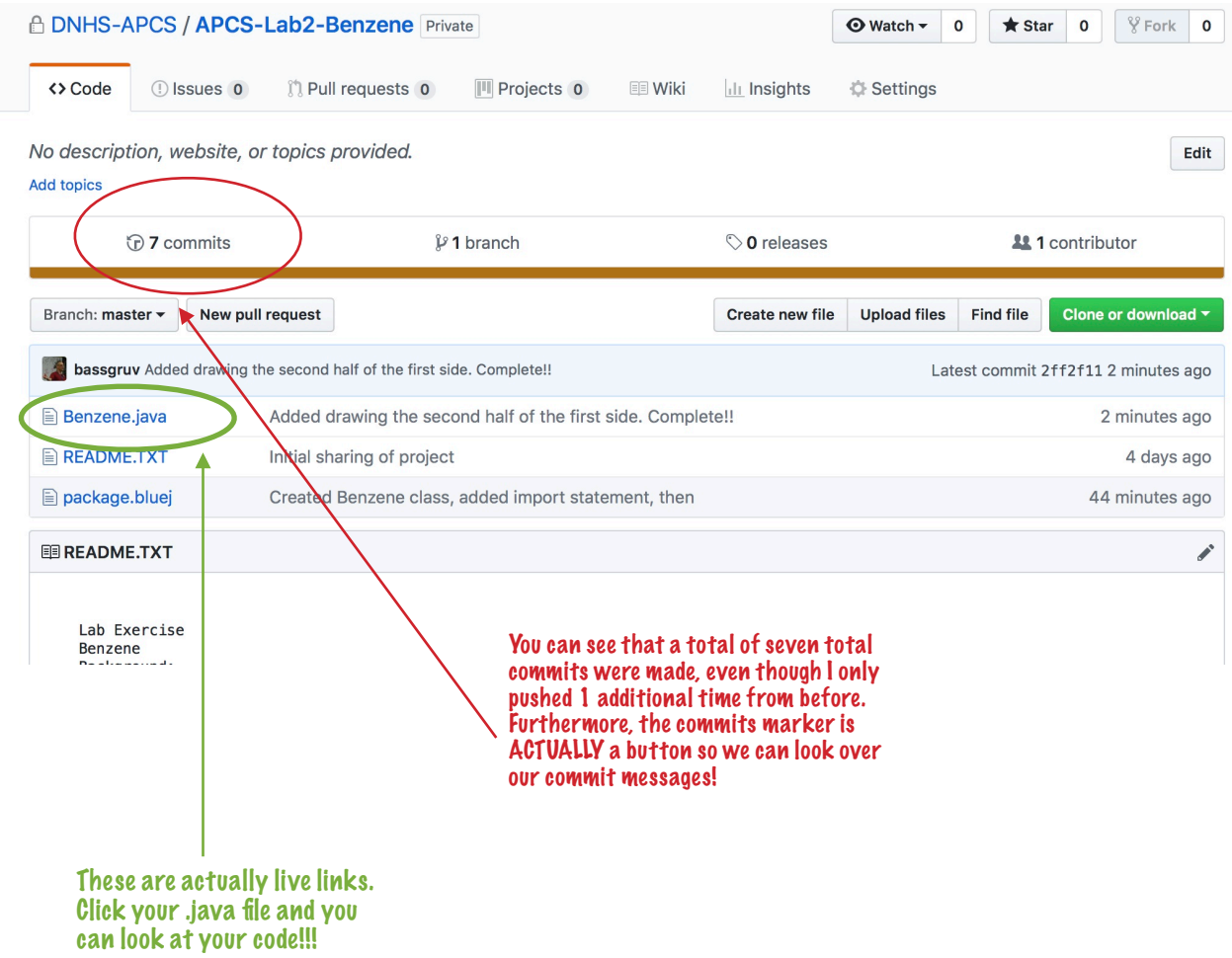


- Refresh the internet browser, and we can see the results!

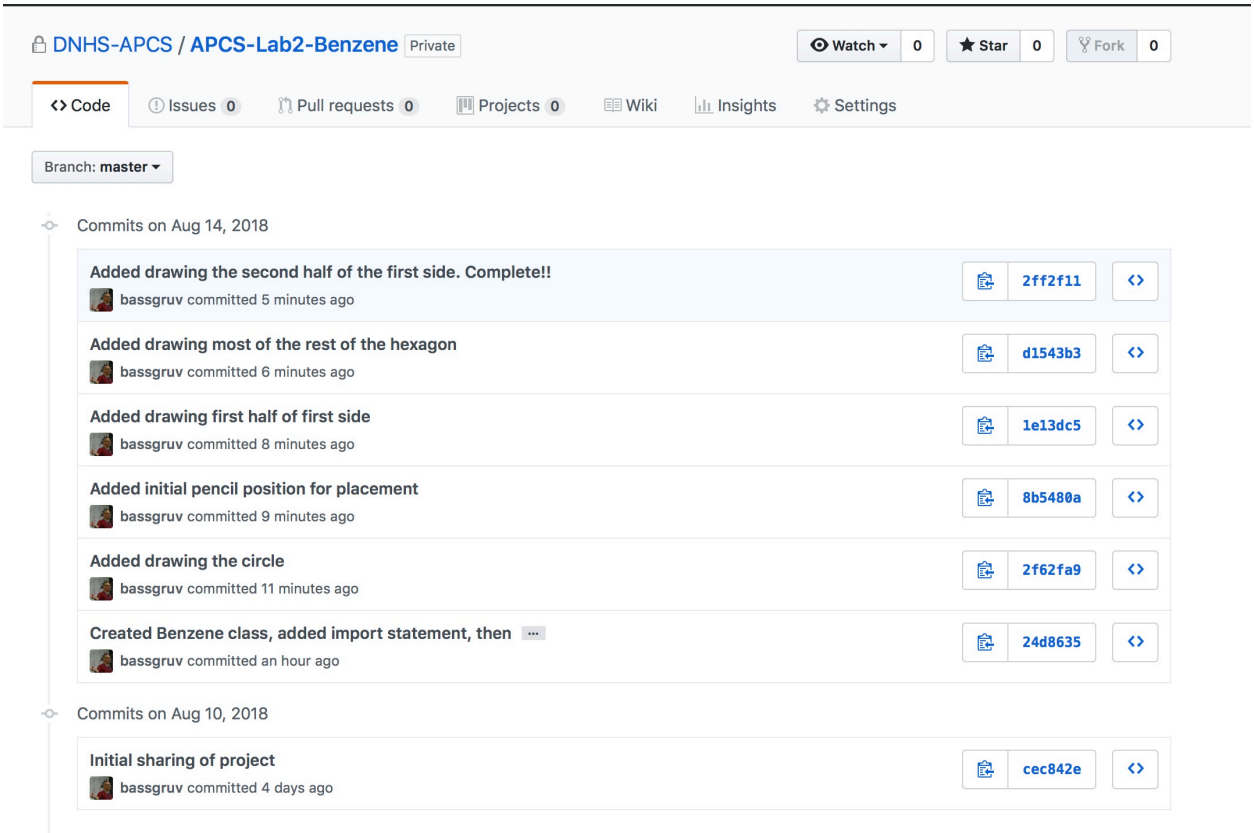


- You don't have to push every time you commit. Many students will commit often during a lab session (we call these atomic commits; they're frequent and done after small changes are made and tested) and then push only once at the end of class. Not to worry! All the commits you've done get recorded in the "local" repo and then included in that final push, so you lose nothing! To demonstrate, I'll add several "components" to Lab 2 and make a few commits along the way using the process described above (without the push). I'll only "push" to GitHub at my final commit.

- Here's the resultant repository on GitHub after those operations:



- If we click on the commit button. . . we can see all of our commits listed from newest at the top to oldest at the bottom!!!

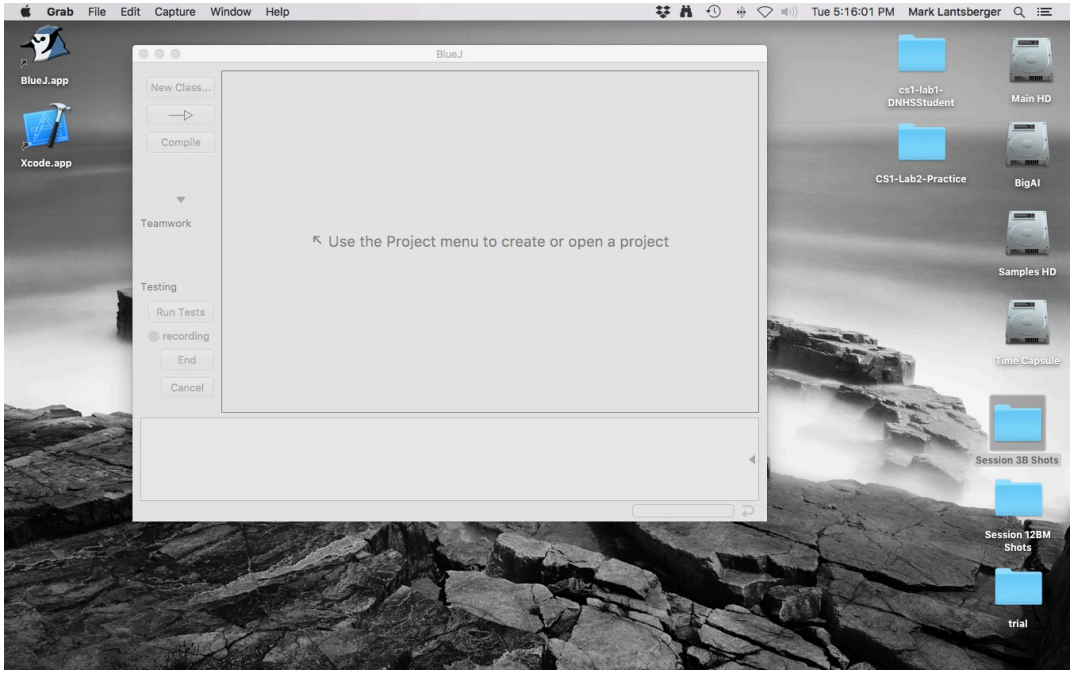


II. Resuming a Project at a New Location

Adding a Repo at Home!!

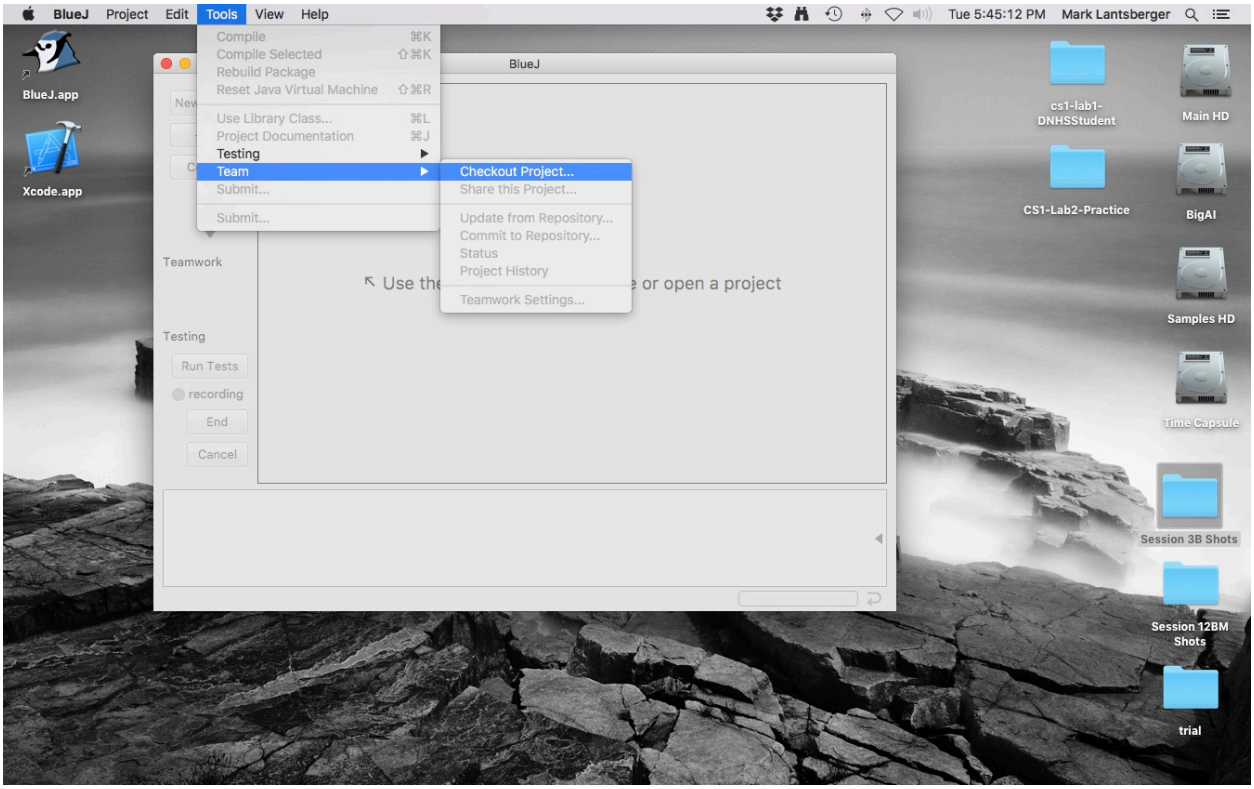
- So now you get home and you'd like to work on your code to finish it. What can you do?? Well, we can work with the repository we have on GitHub. It's important to recognize that GitHub is NOT an IDE!!!! It is NOT like BlueJ. It CANT compile or run code, it can only log and store changes you've made in a repository. What we need is to create a new (and different) "local" repository on a different computer than we've already worked with!!

- Open BlueJ and you'll see the usual "splash" screen. If you had a project open when you last quit, you'll need to close that project using the File menu. (And yes, I actually did this on a different computer. . .)

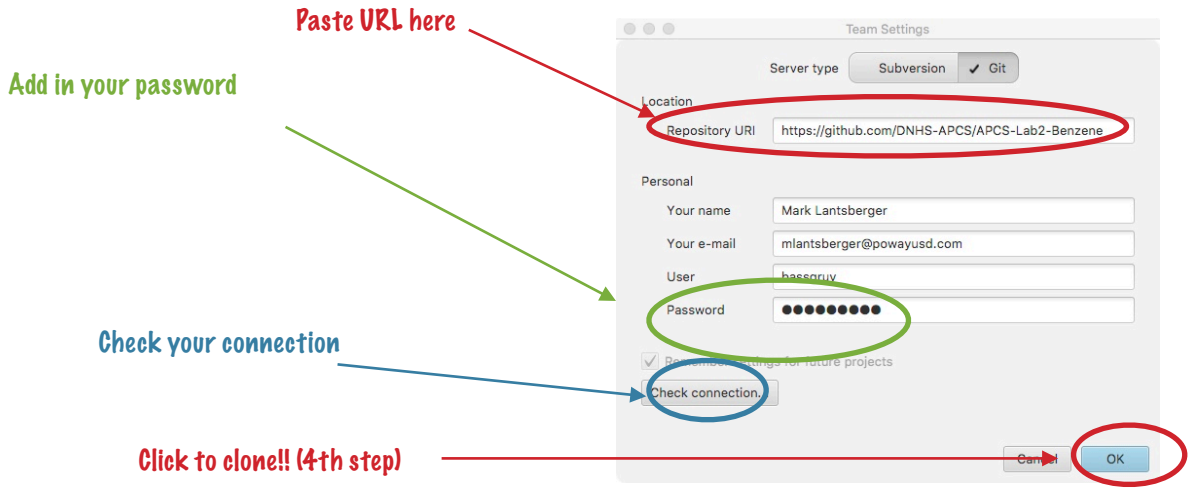


- Log on to GitHub and navigate to the repository you want to work with. Copy the URL of this location to use in a moment.

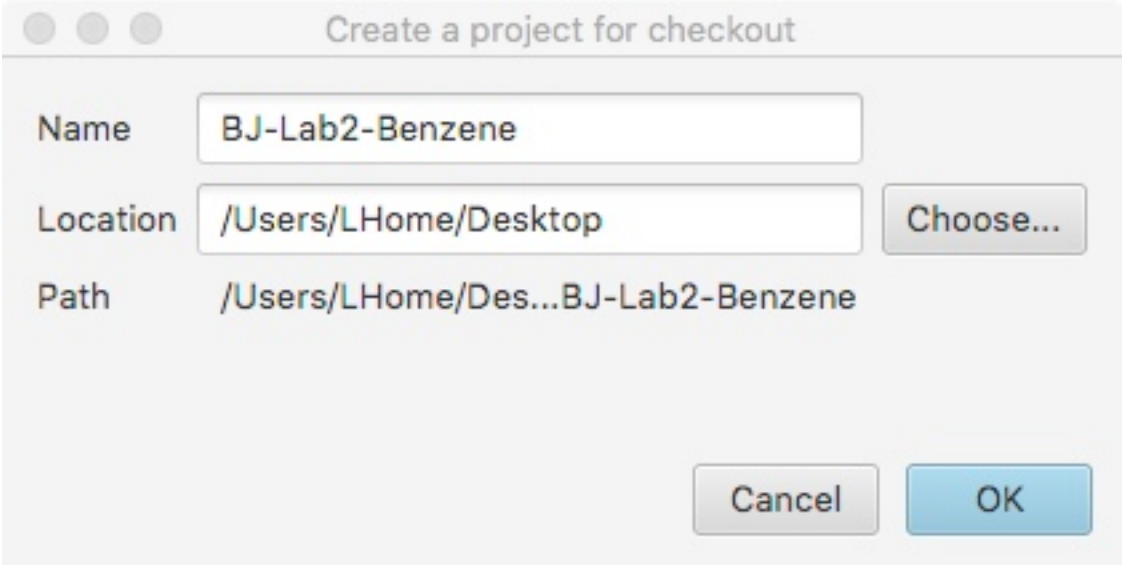
- In the Tools menu, select Team, followed by Checkout Project. What this option will do is create a ***clone*** of your repository on GitHub here "locally" on your home machine.



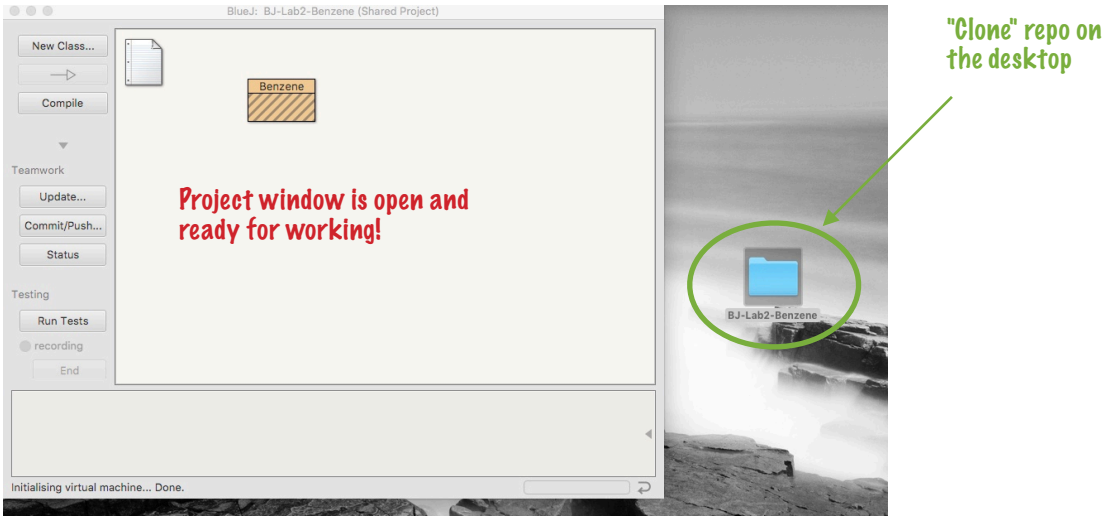
- Paste the URL from GitHub and enter your account credentials.



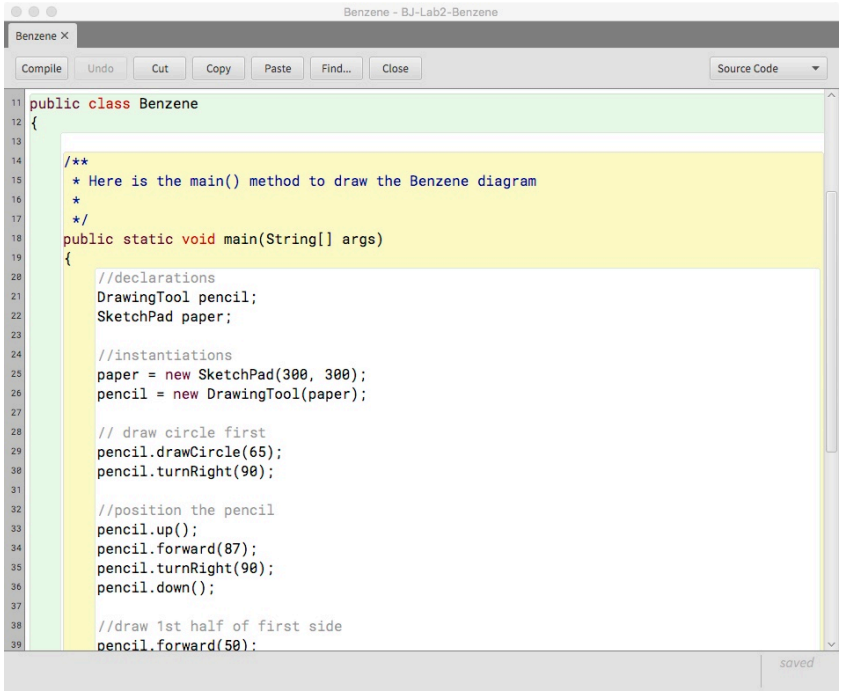
- Choose a location and a project name. Be ORGANIZED and choose a name very similar to your repo. I usually start my "locals" with BJ for BlueJ.



- It may take a moment, but your project will be downloaded and opened for you!

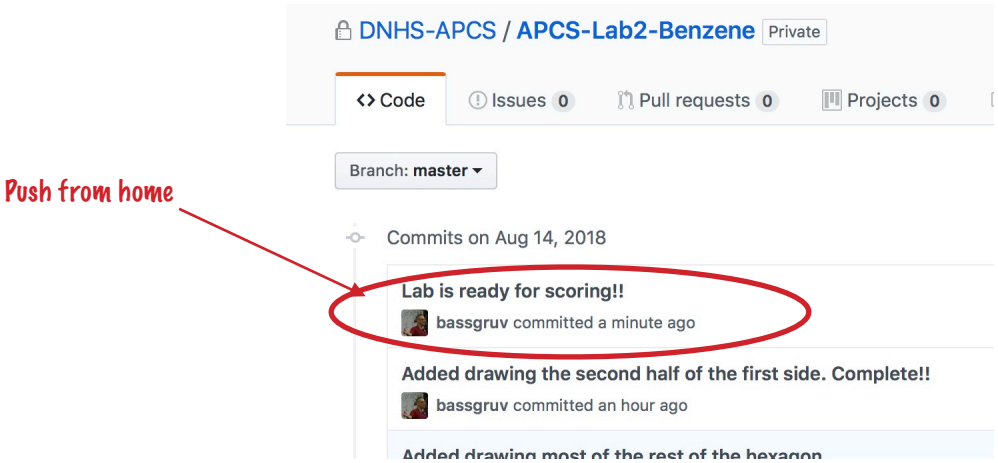


- Notice, this is the same state of the code from my final push at my "school" machine.



- Now we'll make some final edits to our project and run one last test to be certain all is working. . . One final commit and push just as we did before. . .

- And we can see on GitHub, that everything has been updated!!

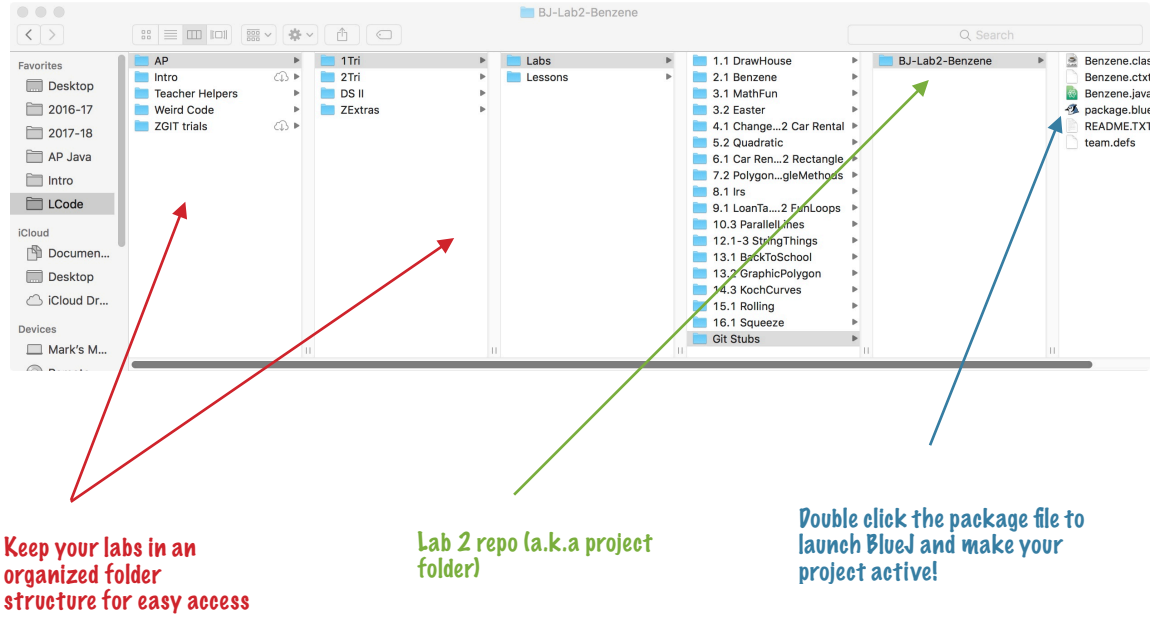


III. Resuming a Project Where You've Already Worked

Resuming work once a "local" is established

- So, now your back at school and you want to get your lab scored. But, the final version isn't on your school machine!! Not to worry, we only need to "update" your "local" repo from GitHub. Then you're ready!!!

- Launch BlueJ **from your repo** (a.k.a. your project folder)



- This repo has an old version of your code. To update this "local" repo, simply click the "Update" button (it's directly above the commit/push button). This will update your "local" repo to the code state on GitHub. This operation is known as a ***pull***.You'll need to put in your GitHub password for authentication. . . . and that's it!!! You're ready to add more code, commit, and push! Just like before.

A Recap of the GitHub Process

For A Brand New Project

- Create a new (Private) repo on GitHub.
- Create a new "local" repo in BlueJ at school, establish GitHub repo as a remote, perform initial ***commit*** and ***push***.
- Write code and test!!!
  - ***commit*** often!!! Don't forget Git comments.
  - ***push*** to GitHub at least once at the end of your work session.

Working In A New Location

- In BlueJ, make a ***clone*** of the desired GitHub repo using "Checkout".
- Write code and test!!!
  - ***commit*** often!!! Don't forget Git comments.
  - ***push*** to GitHub at least once at the end of your work session.

Resuming Work In A Previous Location

- Launch BlueJ using the package icon in your "local" repo.
- Click the "Update" button (this performs a ***pull***).
- Write code and test!!!
  - ***commit*** often!!! Don't forget Git comments.
  - ***push*** to GitHub at least once at the end of your work session.